XDP™ XDPP1100, XDPP1100A reference manual

Digital power controller

About this document

Scope and purpose

This document focuses on the XDP™ XDPP1100 hardware (HW) implementation and can be used as a reference document for firmware (FW) developers. The aim is to describe the high-level functions and provide block diagram illustrations of the implemented HW. Furthermore, it demonstrates how the XDP™ XDPP1100 interacts with the external components within the applications and show how the sensed information from the analog input pins are processed by the HW.

This document describes both the XDP™ XDPP1100 and XDPP1100A versions of the HW. References to XDPP1100 apply to both the XDPP1100 and XDPP1100A versions of the product, while references to XDPP1100A specifically refer to the XDPP1100A version of the product. The main difference between these two versions is that the XDPP1100A includes V_{DD} undervoltage lockout (UVLO) as an optional shutdown fault mechanism.

Intended audience

Power supply design and FW engineers, isolated digital brick module designers, and telecom and server power system designers.

About this product group

Target applications

- Telecom
- Datacenter and computing
- SMPS

Product family

XDP™ is a high-performance digital power controller designed to optimize the IC for the specific applications. They feature digital algorithms for high power efficiency and minimal standby power, digital parameter configuration for increased flexibility and lower development effort, and optimized bill of materials (BoM).

Table of contents

About this document ... **1**
About this product group ... **1**
Table of contents ... **2**
1 Introduction ... **8**
1.1 Applications ... 8
1.2 Control registers and PMBus commands ... 8
1.3 Naming conventions ... 9
1.3.1 Loops and phases ... 9
1.3.2 Binary number format convention ... 9
1.4 Structure of this document ... 10
2 Voltage sense ... **11**
2.1 VS module configuration ... 12
2.2 Voltage sense analog-to-digital converter ... 15
2.2.1 Analog front end and front-end compensation ... 16
2.2.2 Tracking ADC ... 16
2.3 Voltage sense processor ... 18
2.3.1 Output voltage processing ... 18
2.3.2 Rectification voltage processing ... 19
2.3.2.1 V_{RECT} timing for single PWM signal ... 20
2.3.2.2 V_{RECT} watchdog timer ... 21
2.3.2.3 Deglitcher ... 22
2.3.2.4 V_{RECT} timing for two PWM signals ... 22
2.3.2.5 V_{IN} transient response ... 22
2.3.2.6 V_{RECT} start-up programming ... 23
2.3.2.7 Same cycle response ... 24
2.3.2.8 V_{RECT} delay counters ... 25
2.3.3 V_{IN} processing ... 26
2.4 VS registers ... 26
3 Current sense (IS) ... **36**
3.1 Current sense module configuration ... 37
3.2 Current sense analog to digital converter ... 39
3.2.1 Current sense analog front end ... 40
3.2.2 Current estimator ... 41
3.2.2.1 PWM state ... 42
3.2.2.2 Slope estimator ... 47
3.2.2.3 Error tracking ... 51
3.2.2.4 Trace inductance of the PCB current sensing ... 52
3.3 Current sense processor ... 53
3.3.1 ADC codes to amps ... 53
3.3.2 Peak current limiting ... 54
3.3.3 Negative current limiting ... 54
3.3.4 Short-circuit protection ... 54
3.3.5 Error tracking fault detection ... 55
3.4 Current sense registers ... 56
3.5 Current sense PMBus commands ... 64
4 Telemetry sense ... **66**
4.1 Telemetry sense ADC ... 66

Table of contents

4.2 TS input mux and sequencing 67
4.3 Telemetry sense current DAC 69
4.4 Telemetry sense processor 69
4.4.1 Sequencer 70
4.4.2 Gain and offset correction 70
4.4.3 V_{IN} computation 71
4.4.4 Internal temperature (ITSEN) computation 71
4.4.5 X-valent measurement 72
4.5 Telemetry sense registers 73
5 Voltage control 80
5.1 PMBus commands to HW parameters 81
5.2 Ramp generator 81
5.3 Interrupts 83
5.4 Multi-segment droop (load-line) 83
5.4.1 Droop voltage computation 84
5.4.2 Droop voltage filtering 86
5.5 Output summation and clamping 87
5.6 Voltage control registers 88
5.7 Voltage control PMBus commands 95
6 Compensator 99
6.1 Compensation filter 99
6.1.1 Pre-filter 100
6.1.2 PID term computation 103
6.1.3 Post-filter and summation 112
6.1.4 Input/output clamping of the compensation filter 113
6.1.5 Output override – forced duty cycle 113
6.1.6 Coefficient scaling 113
6.1.7 Freeze, reset accumulator 114
6.2 Input voltage feed-forward 114
6.2.1 Output voltage target computation 115
6.2.2 Input voltage source select and computation 115
6.2.3 Override and adjustment options for FF 116
6.3 Control mode selection – peak current mode 118
6.4 Open sense fault detection 119
6.5 Compensation filter registers 120
7 Digital pulse width modulator 127
7.1 PWM ramp generator 128
7.1.1 PWM ramp modulation schemes 129
7.1.2 PWM ramp synchronization 131
7.2 PWM pulse generator 133
7.2.1 Pulse generator enable 133
7.2.2 Ramp selection (loop/phase) 134
7.2.3 Edge selection (t1/t2) 134
7.2.4 Dead-time programming 136
7.2.5 Force high and force low 138
7.3 Feedback control modes 138
7.3.1 Voltage mode control 138
7.3.2 Peak current mode control 140
7.3.3 Maximum and minimum pulse width enforcement 141

7.3.4 Forced duty cycle T1, T2 142
7.4 Burst mode 143
7.5 Fast-transient response 144
7.5.1 Fast-transient response – load step 144
7.5.2 Fast-transient response – load release 146
7.6 PWM interrupts 147
7.7 PWM registers 148
7.8 PWM PMBus commands 188
8 Telemetry 193
8.1 Output voltage (V_{OUT}) telemetry 193
8.2 Input voltage (V_{IN}) telemetry 194
8.3 Output current (I_{OUT}) telemetry 196
8.4 Input current (I_{IN}) telemetry 197
8.5 Input CE 197
8.6 Output and input power telemetry 198
8.7 Temperature telemetry 198
8.8 Duty-cycle telemetry 200
8.9 Switching frequency (F_{switch}) telemetry 201
8.10 General-purpose ADC telemetry 201
8.11 Telemetry interrupts 203
8.12 Telemetry high/low watermark detect 204
8.13 Telemetry registers 205
8.14 PMBus 232
9 Fault handler 237
9.1 Loop faults 237
9.1.1 Output voltage faults 238
9.1.2 Input voltage faults 240
9.1.3 Output current faults 241
9.1.4 Input current faults 243
9.1.5 Power warnings 243
9.1.6 Temperature faults 244
9.1.7 Current sharing fault 245
9.1.8 V_{OUT_MAX}, V_{OUT_MIN} warning 245
9.1.9 Sync fault 245
9.1.10 V_{DD} undervoltage lock out fault (XDPP1100A only) 246
9.1.11 T_{ON_MAX}, T_{OFF_MAX} faults 246
9.1.12 Loop fault latching 246
9.2 Common faults 247
9.2.1 Current sense tracking fault 248
9.2.2 Peak current limit fault 248
9.2.3 SCP fault 248
9.2.4 Flux balance fault 248
9.2.5 Open VS fault protection 248
9.2.6 Common fault latching 249
9.3 Fault interrupts (IRQ) 250
9.4 Faults priority encoding 251
9.5 Fault shutdown 252
9.6 Fault pin mapping 253
9.7 Fault registers 253

9.8 Fault PMBus commands 280
10 Current sharing (I_{SHARE}) 298
10.1 Current sharing circuit 298
10.2 Current sharing PI filter 300
10.3 Current sharing FW override 301
10.4 Current sharing pin, DAC and ADC configuration 302
10.5 R_{ishare} FW patch 302
10.6 Current sharing registers 303
10.7 Current sharing PMBus commands 306
11 Current balance (I_{BAL}) 307
11.1 Current balance circuit 307
11.2 Current balance PI filter 308
11.3 Current balance FW override 309
11.4 Current balance registers 310
12 Flux balance (F_{BAL}) 311
12.1 Flux balance circuit 311
12.2 Flux balance PI filter 313
12.3 Flux balance FW override 314
12.4 Flux balance DCM operation 314
12.5 Flux balance fault detection 315
12.6 Flux balance registers 316
13 Fan support 320
13.1 Fan PWM output 320
13.1.1 Duty-cycle mode 321
13.1.2 Current mode 321
13.2 Fan speed input 322
13.3 Fan registers 323
13.4 Fan PMBus commands 325
14 IO muxing 327
14.1 Multipurpose digital IO muxing 327
14.2 Digital input priority 329
14.3 GPIO deglitch 329
14.4 Digital IO buffer programming 330
14.5 IO muxing registers 330
15 Central processing unit subsystem 340
15.1 CPUS bus matrix 341
15.2 Cortex®-M0 CPU 341
15.2.1 CPU interrupt sources 342
15.2.2 Cortex®-M0 memory map 344
15.2.3 Remapping feature 345
15.3 Clock and system controller 347
15.3.1 System controller unit 347
15.3.1.1 SCU registers 347
15.3.2 Clock generator unit 357
15.3.2.1 Clock dividers 359
15.3.2.2 Clock gating 360
15.3.2.3 Primary-source clock gating 361
15.3.2.4 CGU registers 362

15.3.3 Reset generator unit ... 395
15.3.3.1 Reset sources ... 396
15.3.3.2 Software power-down ... 397
15.3.3.3 Software reset ... 398
15.3.3.4 RGU registers ... 398
15.4 Memory ... 406
15.4.1 Read-only memory ... 406
15.4.2 Random-access memory ... 407
15.4.3 One-time programmable memory ... 407
15.4.3.1 OTP memory map ... 408
15.4.3.2 OTP configuration interface ... 408
15.4.3.3 Indirect OTP access ... 409
15.4.3.4 OTP timing configuration ... 411
15.4.3.5 OTP direct access ... 412
15.4.3.6 OTP module registers ... 414
15.5 Memory management unit ... 420
15.5.1 MMU registers ... 421
15.6 DMA controller ... 519
15.6.1 DMA block diagram ... 520
15.6.2 DMA memory map ... 521
15.6.3 DMA channel assignment ... 521
15.6.4 DMA registers ... 522
15.7 General-purpose input output (GPIO) module ... 531
15.7.1 GPIO registers ... 532
15.8 WDT module ... 536
15.8.1 Watchdog block diagram ... 537
15.8.2 Watchdog registers ... 537
15.9 Dual-timer (D_{TIMER}) modules ... 540
15.9.1 Dual-timer block diagram ... 541
15.9.2 Dual-timer registers ... 541
15.10 PMBus module ... 546
15.10.1 PMBus clock scheme ... 548
15.10.2 Interrupt generation ... 548
15.10.3 PMBus operation ... 549
15.10.3.1 Configuration ... 549
15.10.3.2 Write transaction ... 550
15.10.3.3 Read byte transaction ... 551
15.10.3.4 PMBus ARA command ... 554
15.10.4 PMBus registers ... 555
15.11 I^2C module ... 573
15.11.1 I^2C operating modes ... 573
15.11.1.1 Status information ... 573
15.11.1.2 Master transmit ... 574
15.11.1.3 Master receive ... 577
15.11.1.4 Slave transmit ... 579
15.11.1.5 Slave receive ... 579
15.11.2 I^2C registers ... 580
15.12 CRC module ... 585
15.12.1 CRC registers ... 586

Table of contents

15.13 UART 587
15.13.1 UART block diagram 588
15.13.2 UART registers 588
15.14 Debugger port 603
15.14.1 Serial wire debugger interface 604
16 Related resources 607
Abbreviations or acronyms 608
Revision history 612
Disclaimer 614

1 Introduction

This chapter first introduces the applications and topologies where XDPP1100 can be used. Next there is a brief overview of the PMBus and register programming through the XDP1100 graphical user interface (GUI). General naming conventions are then summarized, including loop and phase definitions, symbols and abbreviations, signal naming and binary number formatting. The final section describes the structure of the rest of this document.

1.1 Applications

The XDPP1100 is a versatile digital controller that can be employed in various applications, including isolated and non-isolated DC-DC brick converters, intermediate bus converters and telecom power supplies, as well as point-of-load converters. The controller provides flexibility through the FW, and the same HW can be configured for various topologies such as:

- Active clamp forward (ACF)
- Full-bridge (FB) primary with full-wave or center-tap (CT) secondary
- Half-bridge (HB) primary with full-wave or CT secondary
- Buck
- Boost
- Buck-boost
- Unregulated LLC

For more details, refer to the XDPP1100 datasheet and XDPP1100 digital power supply controller application note.

1.2 Control registers and PMBus commands

The XDPP1100 controller is configured by application-specific parameter settings loaded into control registers. Access to the control register map can be achieved over I^2C and PMBus. Module manufacturers can use I^2C to set up controller features and parameters. PMBus commands enable the end user to customize system applications. Access to the register map is supported by the XDPP1100 GUI. A full listing of the control registers and their descriptions can be found in the XDPP1100 register descriptions document. Registers associated with a HW function discussed in this document are also described at the end of the relevant chapter.

For typical applications, the control registers are pre-programmed at the factory and stored in the on-chip non-volatile memory (NVM), which is then downloaded to the control registers during initialization of the controller as it powers up. Control registers can be reprogrammed in the field via the serial communication (I^2C) bus and stored in the NVM. The XDPP1100 controllers support multiple reprogramming cycles, and this is easily accomplished with the XDPP1100 GUI.

In addition to supporting multiple reprogramming cycles, the XDPP1100 controllers also support storing multiple configurations in NVM, where the initialization setting is selected from one of these stored configurations depending on the value of an external resistor connect to the XADDR pin. This capability is referred to as “multi-config”. This allows a single configuration file to support multiple applications, which is useful when multiple controllers are used in a system but require different configurations because they support multiple types of output rails.

With multi-config, the XDPP1100 controllers are capable of storing up to 16 configurations, and these 16 configurations may be reprogrammed once if needed. The controller identifies the proper configuration to load based on information stored in the configuration; this means an indicator bit identifying a multi-config should be used, with a pointer to the location of the multi-config space in the NVM.

1.3 Naming conventions

Within this document, certain naming conventions are used as described in the sections below.

1.3.1 Loops and phases

The XDPP1100 controller supports regulation of up to two output voltages. The two control loops are referred to as “Loop 0” and “Loop 1”.

Loop 0 is associated with voltage sense (VS) input pair VSEN/VREF as well as HW blocks Voltage Control 0, PID 0 and typically PWM ramp0. PMBus commands relevant to Loop 0 are found on PMBus PAGE 0.
Loop 1 is associated with VS input pair BVSEN/BVREF as well as HW blocks Voltage Control 1, PID 1 and typically PWM ramp1. PMBus commands relevant to Loop 1 are found on PMBus PAGE 1.

The XDPP1100 controller supports current sense for up to two phases through input pairs ISEN/IREF and BISEN/BIREF. In a single-loop, dual-phase (interleave) system, these pairs may be referred to as Phase 1 (ISEN/IREF) and Phase 2 (BISEN/BIREF) throughout this and other XDPP1100 documentation.

1.3.2 Binary number format convention

Throughout this document in figures and text and in other XDPP1100-related documentation, binary numbers are often referred to as having the form Ux.y for unsigned numbers or Sx.y for signed numbers. The definitions of these references are given below.

Ux.y indicates an unsigned number, with “x” indicating the bit position of the MSB and “y” indicating the bit position of the LSB, each relative to the binary point. Sx.y indicates a signed number in two’s compliment format with “x” and “y” defined as described above for the unsigned case. Figure 1 illustrates the x and y meanings.

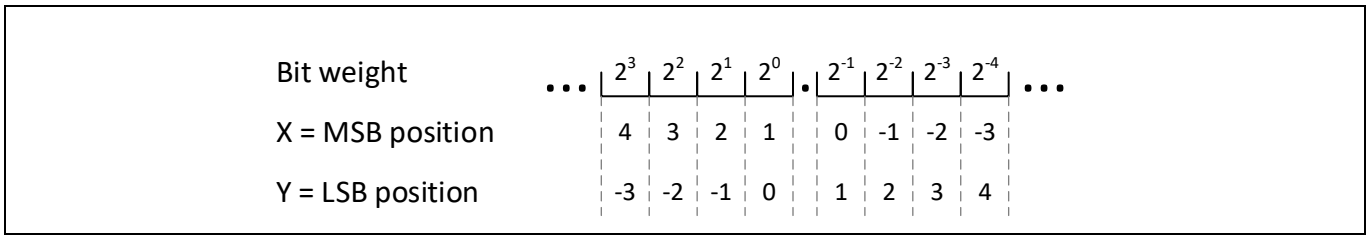

Figure 1 Binary number format x, y definitions

Some unsigned and signed number examples are given below.

- **U8.3** is an 11-bit unsigned number with 8 bits to the left of the binary point (MSB weight 2^7) and 3 bits to the right of the binary point (LSB weight 2^{-3})
- **U7.0** is a 7-bit unsigned number with 7 bits to the left of the binary point (MSB weight 2^6) and 0 bits to the right of the binary point (LSB weight 2^0)
- **U-4.10** is a 6-bit unsigned number with all bits to the right of the binary point. The MSB is offset by 4 bits to the right of the binary point (MSB weight 2^{-5}) and the LSB is 10 bits to the right of the binary point (LSB weight 2^{-10})

- **S9.3** is a 12-bit signed number with 9 bits to the left of the binary point (MSB weight 2^{8}) and 3 bits to the right of the binary point (LSB weight 2^{-3})
- **S8.-2** is a 6-bit signed number with all bits to the left of the binary point. The MSB is 8 bits to the left of the binary point (MSB weight 2^{7}) and the LSB is offset by 2 bits to the left of the binary point (LSB weight 2^{2})

1.4 Structure of this document

The following 14 chapters describe the implemented HW of XDPP1100. Each chapter presents an independent part of the HW functionality. At the end of each chapter, relevant registers and PMBus commands related to the described functionality are summarized. Throughout every chapter, the recommended register programming as well as PMBus commands are provided. Below, the chapter contents are summarized.

The most important analog functions are discussed within Section 2, Section 3 and Section 4, which describe voltage, current and telemetry sense, respectively. These chapters specify how XDPP1100 interfaces with the external environment. System feedback-related functionalities are described in Section 5, Section 6, and section 7, focusing on the following topics:

- Section 5 illustrates how the control voltage is set for the feedback system as well as the droop voltage computation.
- Section 6 describes the compensation filter and its relevant settings as well as feed-forward (FF) and control mode selection functionalities.
- Section 7 focuses on the implementation of the digital pulse width modulation (PWM) and the programming of voltage and peak current mode controls (VMC and PCMC), including fast transient response.

Various submodules for HW computed telemetry are explained in Section 8, while Section 9 describes the fault detection and reporting. Current sharing as well as current and flux balancing are all part of the same block and are discussed in Section 10, Section 11, and Section 12, respectively. Fan support is implemented as a combination of HW and FW, and it is explained in Section 13. Section 14 focuses on the programming of multiple digital IO pins and Section 15 describes the CPU subsystem in detail.

2 Voltage sense

This chapter describes the VS module and its submodules, voltage sense ADC (VSADC) and voltage sense processor (VSP) in detail. In addition, the user-programmable settings for configuring the features of the sensed voltages are described and relevant registers provided.

The XDPP1100 controller has three following analog input pin pairs for voltage sensing:

- VSEN/VREF, for V_{OUT} sensing
- VRSEN/VRREF, for V_{RECT} sensing or primary-side V_{IN} sensing
- BVSEN_BVRSEN/BVREF_BVRREF, for V_{OUT} sensing in dual-loop operation or V_{RECT} sensing for interleaved topology

Typical connection of VSEN/VREF and VRSEN/VRREF in a FB converter with CT rectifier for V_{OUT} and V_{RECT} sensing is shown in Figure 2, where the sensed voltage connections are illustrated in red. Both voltages are sensed through a resistive divider, which introduces gain and needs to be considered in the computations, as will be discussed further.

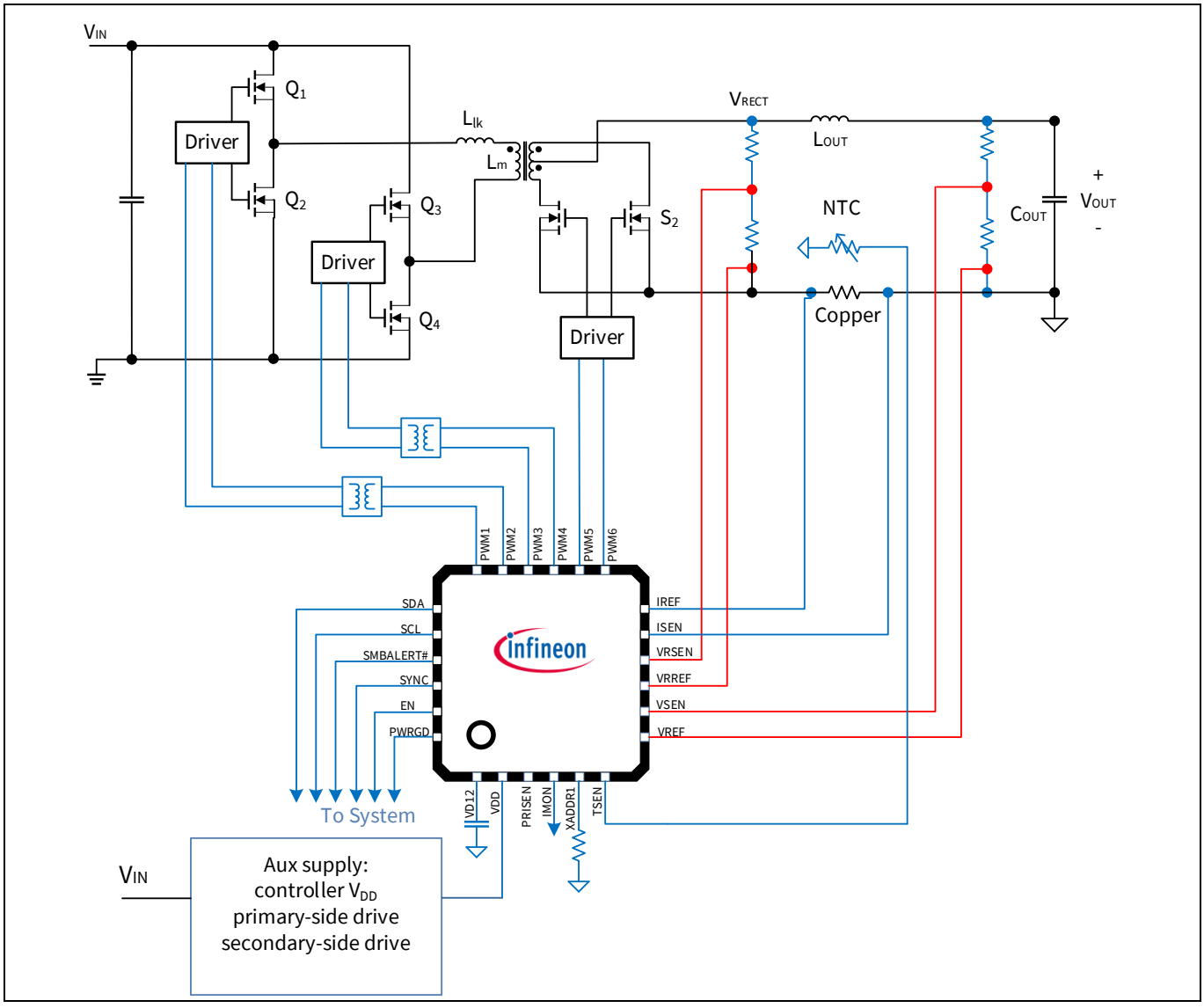

Figure 2 V_{OUT} and V_{RECT} sensing in FBCT converter with VMC

2.1 VS module configuration

The XDPP1100 controller contains three VS modules – VS0, VS1, and VS2. Each of them is connected to the above-mentioned analog input pin pairs as follows:

- Input pin pair VSEN/VREF is connected to VS0
- Input pin pair VRSEN/VRREF is connected to VS1
- Input pin pair BVSEN_BVRSEN/BVREF_BVRREF is connected to the module VS2

Depending on the application requirements these input pins can be configured to sense the relevant voltages and provide the necessary information for further computing. As an example, the following topologies require different voltage information for different purposes:

- Interleaved single-loop topology (Loop 0)
- Dual-loop topology (Loop 0 and Loop 1)

VS configuration for interleaved single-loop topology

This topology has two phases (Phase 1 and Phase 2) and one control loop. Simplified voltage module configuration for this topology is shown in Figure 3. The tasks of different VS modules can be summarized as follows:

- VS0 senses and processes V_{OUT} through the input pins VSEN/VREF and computes the error voltage for Loop 0 compensator (PID0) as well as providing the digitalized V_{OUT} for telemetry and fault processing, as described in detail in Section 2.3.1
- VS1 senses and processes the V_{RECT} of Phase 1 or the input voltage through the input pins VRSEN/VRREF. It computes their digital representation for V_{IN} telemetry, fault processing, Phase 1 current sensing and flux balancing as well as input voltage FF computation. V_{RECT} processing is discussed in Section 2.3.2
- VS2 senses and processes the V_{RECT} of Phase 2 through the input pins BVRSEN/BVRREF and computes the digitalized V_{RECT} for Phase 2 current sensing and flux balancing as well as input voltage FF computation

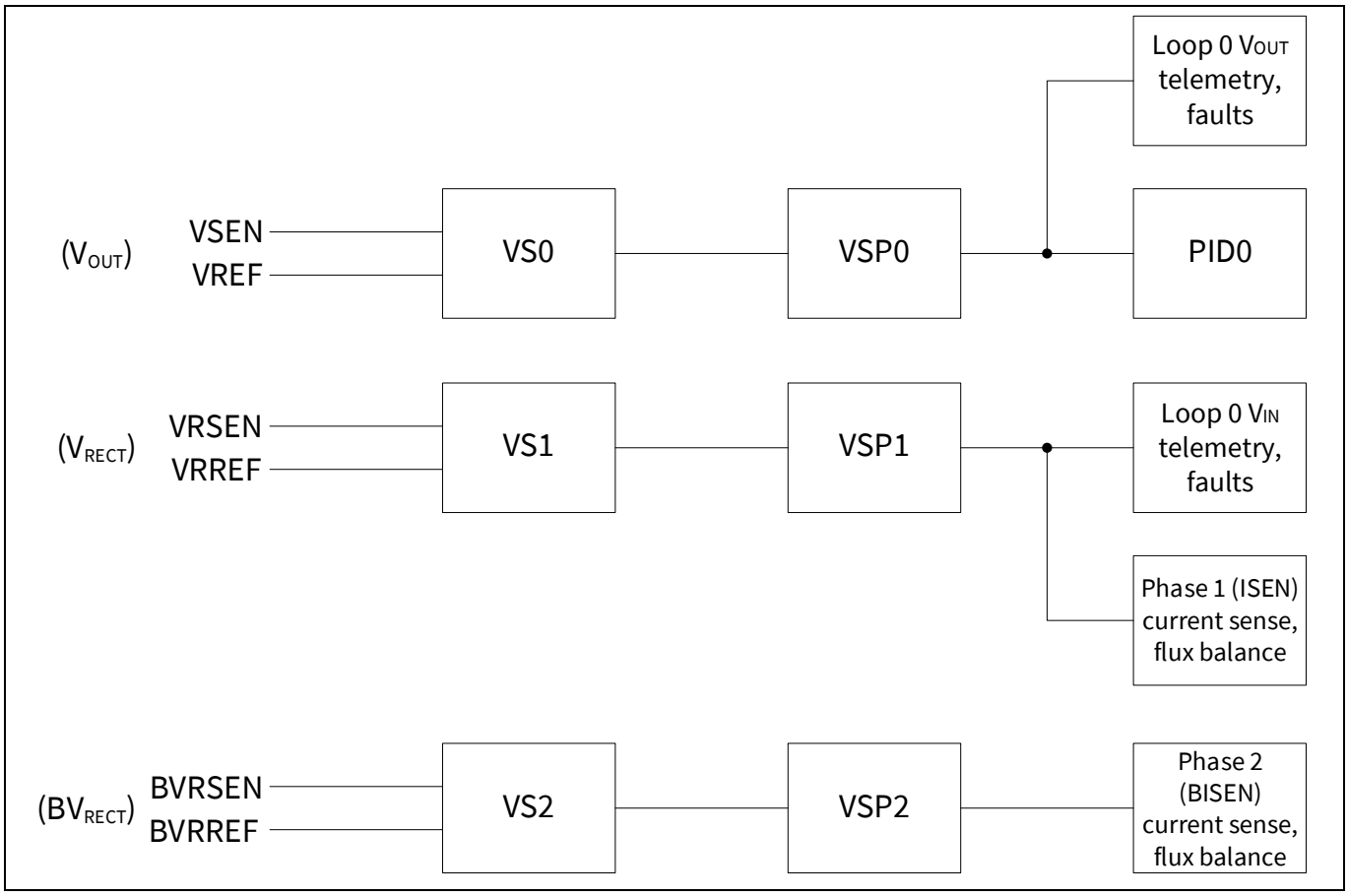

Figure 3 VS module configuration for single-loop, dual-phase topology

VS configuration for dual-loop topology

This topology has two control loops and the VS module VS0 is configured in a similar way as the interleaved topology. The simplified voltage module configuration is presented in Figure 4. The tasks of different VS modules for this topology can be summarized as:

- VS0 senses and processes Loop 0 output voltage through the input pins VSEN/VREF and computes the error voltage for Loop 0 compensator (PID0) as well as providing the digitalized V_{OUT} for telemetry and fault processing, as described in detail in Section 2.3.1
- VS1 senses and processes Loop 0 V_{RECT} or the input voltage through the input pins VRSEN/VRREF and computes their digital representation for V_{IN} telemetry, fault processing and Phase 2 current sensing and flux balancing. V_{RECT} processing is discussed in Section 2.3.2
- VS2 senses and processes Loop 1 output voltage through the input pins BVSEN/BVREF and computes the error voltage for Loop 1 compensator (PID1) as well as providing the digitalized V_{OUT} for telemetry and fault processing, as described in detail in Section 2.3.1

In the dual-loop configuration, if the V_{OUT} of Loop 0 is the input voltage of Loop 1, the digitalized voltage from VS0 can be also used for Loop 1 V_{IN} telemetry, fault processing and Phase 2 current sensing. An example of such configuration is post-buck topology, discussed briefly in Section 6.2.2

Regarding the VS module VS1, it should be noted that if both loops (Loop 0 and Loop 1) share the input voltage, the voltage information provided by the sense pins VRSEN/VRREF can be used also for Loop 1 V_{IN} telemetry, fault processing, and current sensing. However, this information cannot be used for Loop 1 flux balance and so dual-loop FB with flux balance on both loops is not supported.

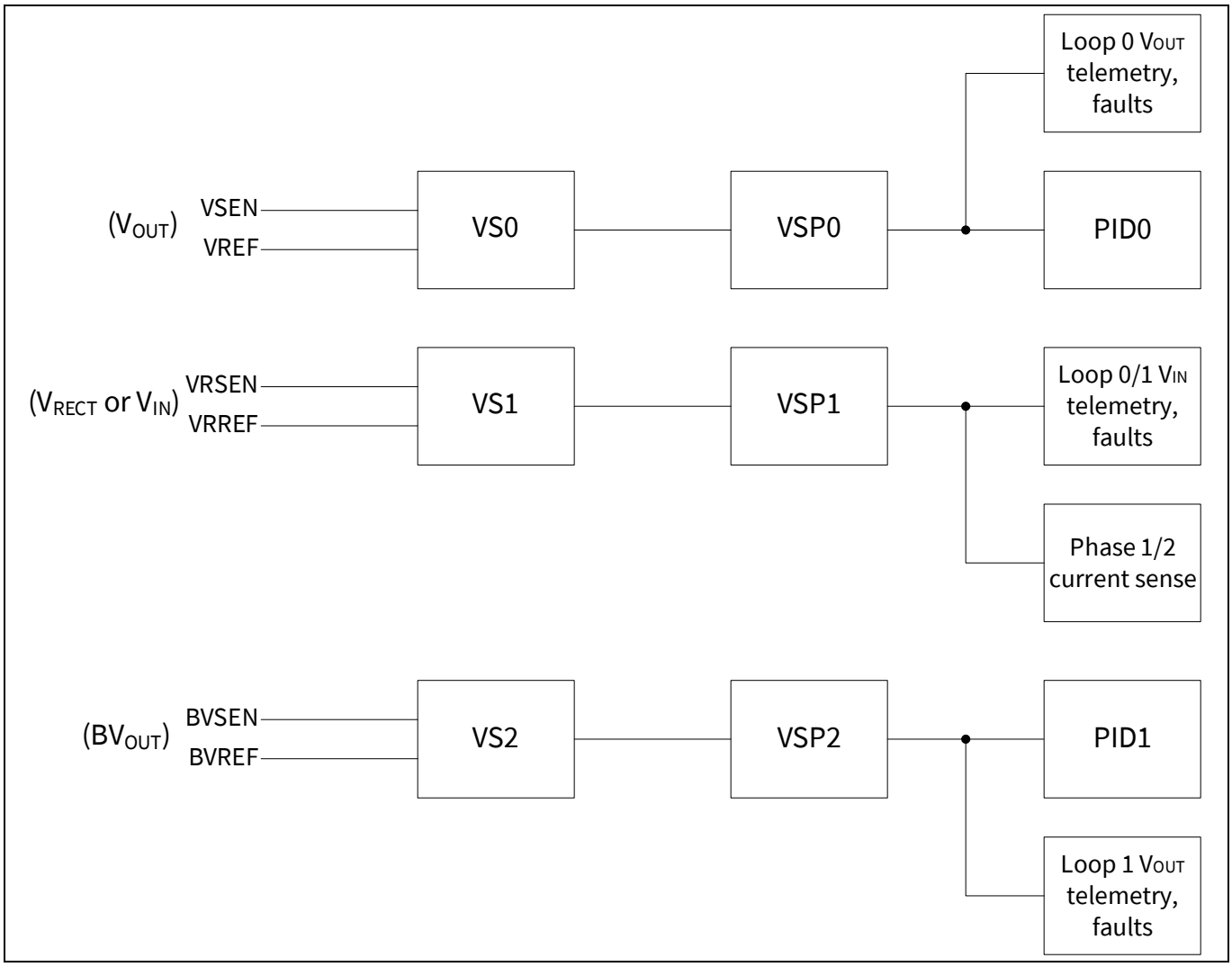

Figure 4 VS module configuration for dual-loop topology

VS module block diagram for processing V_{OUT} and V_{RECT}

A simplified block diagram of the VS module is shown in Figure 5, while the VS module is processing the output voltage. Every VS module (VS0, VS1, and VS2) consists of four submodules:

- Analog front end (AFE)
- Front end offset compensation (FEC)
- Tracking ADC
- VSP

Submodules AFE, FEC, and ADC are collectively referred to as VSADC. The fourth module, VSP, receives the VSADC output as its input and it provides a digital representation of either V_{OUT}, V_{RECT}, or V_{IN} depending on which voltage is being sensed.

Figure 6 shows the VS module block diagram while V_{RECT} is being processed. The dotted lines illustrate the additional features related to the VSP in this mode of operation. These additional features are discussed in Section 2.3.2. The main difference between the V_{OUT} and V_{RECT} modes is that in V_{RECT} mode, the tracking ADC only tracks the input when the input voltage is reflected to the secondary-side V_{RECT}.

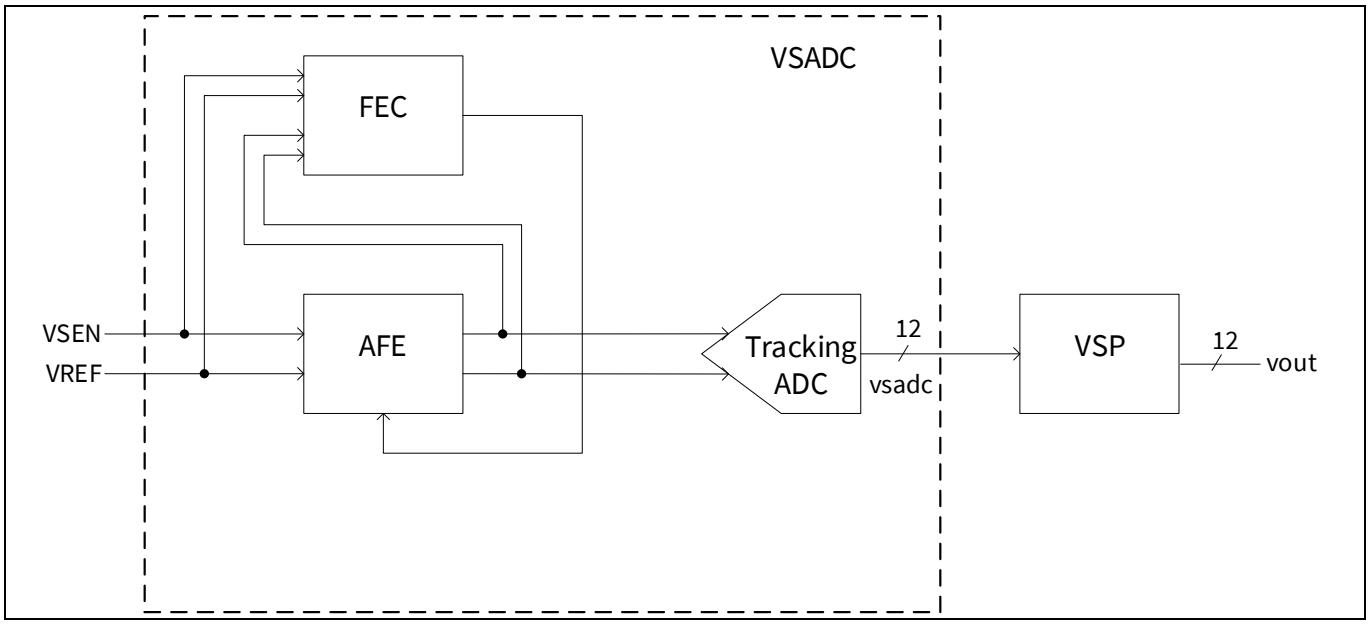

Figure 5 VS module block diagram while processing V_{OUT}

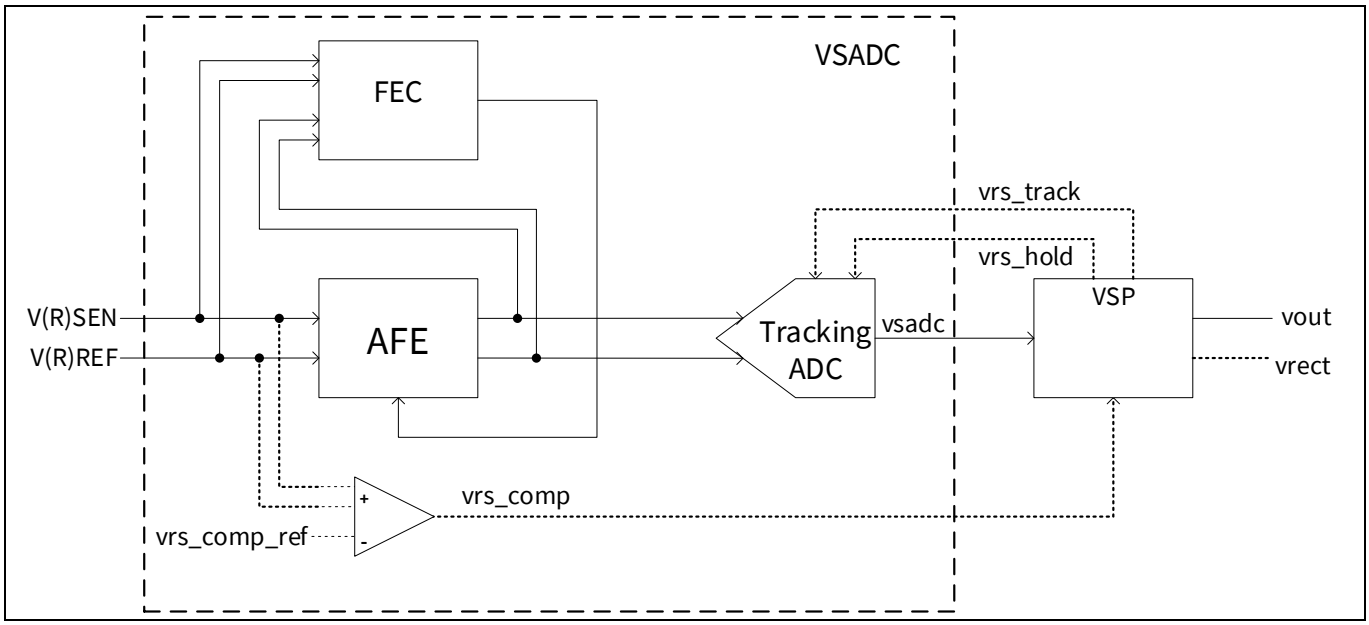

Figure 6 VS module block diagram while processing V_{RECT}

2.2 Voltage sense analog-to-digital converter

This section discusses the VSADC and its relevant submodules in more detail. It receives the sensed voltage as its input and its output is the digitized version of the same voltage. This ADC consists of three submodules:

- AFE
- FEC
- Tracking ADC

A simplified VSADC block diagram is shown in Figure 7.

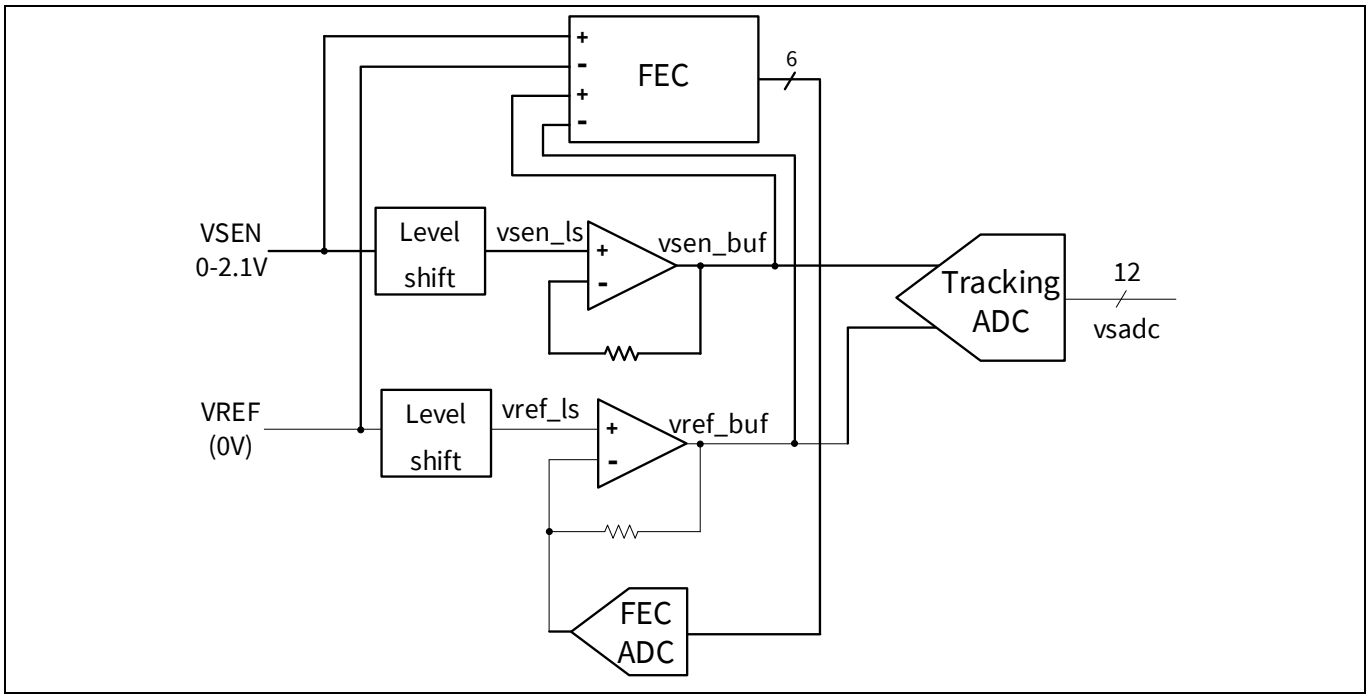

Figure 7 VSADC block diagram

2.2.1 Analog front end and front-end compensation

AFE is composed of:

- A pair of level shifters connected to the sense and reference input pins (e.g., VSEN/VREF in the case of VS0)
- Unity gain buffers

The level shifters allow a wide input voltage range, 0.0 V to 2.1 V, as well as providing a high input impedance. The unity gain buffers provide additional drive strength to the tracking ADC input stage.

The main objective of the FEC is to reduce the effects of:

- Temperature
- Stress
- Lifetime-induced offset drift in the AFE

It compares the differential voltages at the input and output of the AFE and compensates for the difference at the reference path unity gain buffer via a 6-bit DAC. For the FEC module, it is strongly recommended to use the factory settings, selected automatically when programming via the XDPP1100 GUI (although the FEC module provides some programmability).

2.2.2 Tracking ADC

The last VSADC submodule, the tracking ADC, consists of:

- Summing amplifier
- Comparator
- Tracking integrator with programmable step size control
- DAC

A simplified block diagram of the tracking ADC is shown in Figure 8. The summing amplifier amplifies the difference between the buffered differential sensed voltage and the DAC output voltage.

The output of the summing amplifier drives the comparator, whose output determines the size and direction of the next step for the tracking integrator. The tracking integrator output represents the ADC sensed voltage. It is also used to drive the DAC, whose output is summed in the summing amplifier.

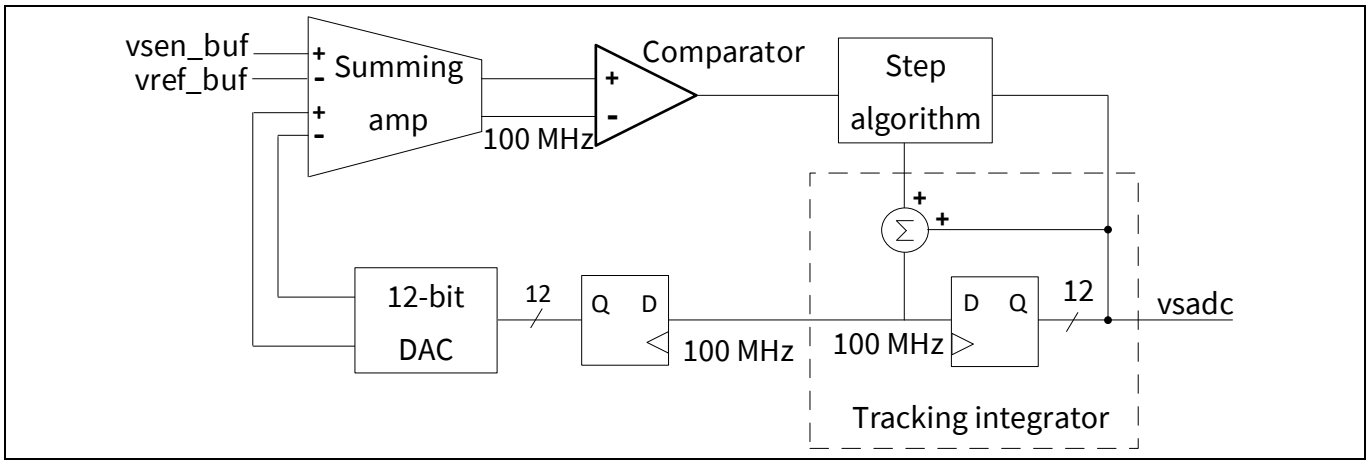

Figure 8 Tracking ADC block diagram

Tracking ADC resolution is 12 bits with an input-referred least significant bit (LSB) weight of 1.25 mV. Note that the AFE input range is restricted to maximum of 2.1 V, meaning only the lower 11 bits of the ADC output are actually used.

The ADC tracking loop integrator step size is programmable via registers:

- **vsX_step_en** enables user-definable step size control
- **vsX_step** sets a fixed integrator step size

The X in the register name denotes 0, 1, or 2, depending which VS module is being utilized. For the highest accuracy, a fixed step size of 1 is recommended. If automatic step size is programmed by setting **vsX_step_en** = 0, the XDPP1100 automatically selects the step size between 1 and 7 based on the perceived distance from the input voltage.

Table 1 shows the recommended settings for step size control for each VS module.

Table 1 Recommended settings for step size control of VS modules

VS module	Pin	Sensing	Recommended settings
VS0	VSEN	V_{OUT}	vs0_step_en = 1, vs0_step = 1
VS1	VRSEN	V_{RECT}	vs1_step_en = 0, vs1_step = 1
VS1	VRSEN	V_{IN}	vs1_step_en = 1, vs1_step = 1
VS2	BVSEN_BVRSEN	V_{OUT}, V_{IN}	vs2_step_en = 1, vs2_step = 1
VS2	BVSEN_BVRSEN	V_{RECT}	vs2_step_en = 0, vs2_step = 1

2.3 Voltage sense processor

The VSP receives the tracking ADC output (the digitized sensed voltage) as its input. Depending on the assignment of the specific module, it processes the incoming data in order to produce:

- Output and error voltages
- Rectification voltage
- Primary-side input voltage

The VS module connected to the input pins VSEN/VREF is always assigned to the output voltage. However, for the modules VS1 and VS2, the user can control the selection of V_{OUT} or V_{RECT} via registers **vsp1_vrs_sel** and **vsp2_vrs_sel**.

In the following subsections detailed processing of V_{OUT}, V_{RECT}, and V_{IN} is described in more detail as well as the relevant user-definable parameters.

2.3.1 Output voltage processing

For output voltage processing, the VS module is configured for V_{OUT} via register **vsp1_vrs_sel** or **vsp2_vrs_sel**, depending on which input pins are used to sense the voltage. A simplified block diagram of the V_{OUT} processing function and error computing is shown in Figure 9.

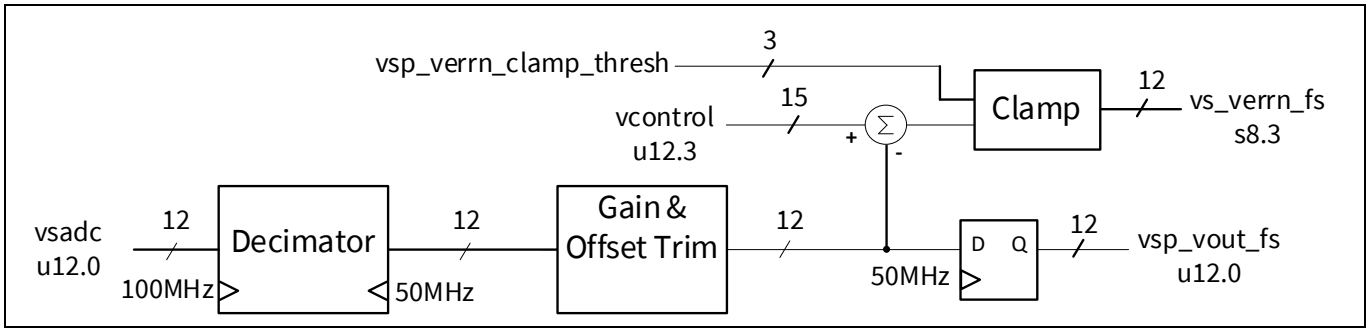

Figure 9 Simplified block diagram of the output voltage processing

The tracking ADC output, VSADC, is received at the conversion rate of 100 MHz. It is decimated (downsampled) to 50 MHz, and thereafter digital gain and offset trim is applied. The resulting registered output represents the differential voltage applied to the sense and reference input pins. This is the digital representation of the output voltage, and it is sent downstream for further processing by the following modules:

- Telemetry
- Fault
- Current sense

In addition to V_{OUT}, the VSP also computes the error voltage, V_{errn}, used by the compensation filter. The error voltage is defined in Equation (2.1)

$$v_{errn} = target\ voltage - measured\ voltage\ , \tag{2.1}$$

where:

- Measured voltage is the gain and offset corrected ADC output
- Target voltage comes from the VCM

It should be noted that the VCM output, $V_{control}$, has higher resolution than the V_{OUT}, thus resulting in an error voltage with the resolution of 156.25 µV.

Programmable clamp (register **vsp_verrn_clamp_thresh**) is applied to limit the maximum error seen by the compensation filter. A higher clamp threshold provides faster initial response to an abrupt change in the target or output voltage, however, with the expense of increased overshoot or undershoot. A lower clamp threshold provides moderate response to abrupt changes with less overshoot or undershoot. Because abrupt changes to the target or output are not common under normal operating conditions, a low-to-medium clamp threshold generally provides the best overall response.

2.3.2 Rectification voltage processing

For rectification voltage processing, the VS module is configured for V_{RECT} via register **vsp1_vrs_sel** or **vsp2_vrs_sel**, depending on which input pins are used to sense the voltage. When V_{RECT} is chosen, the VSP block diagram of the processing function is shown in Figure 10. Compared to the V_{OUT}, V_{RECT} processing has many more additional features.

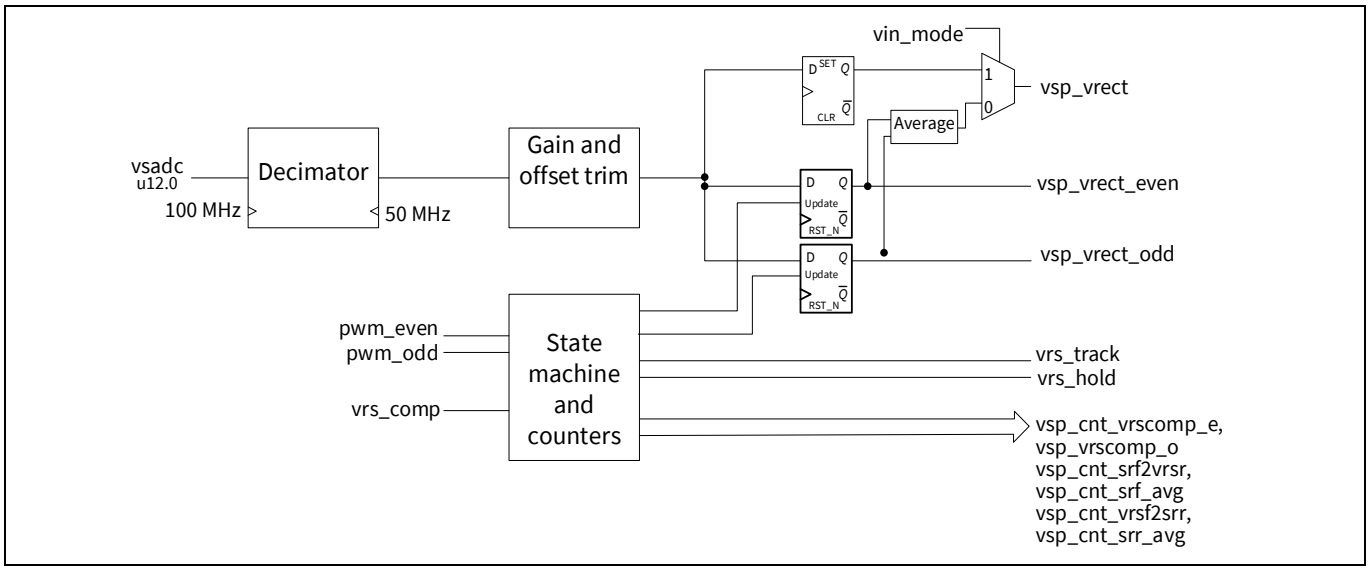

Figure 10 Simplified block diagram of the rectified voltage processing

The V_{RECT} VSP provides three outputs for further functions:

- Measured V_{RECT} of the even half-cycle (**vsp1_vsp_vrect_even** for VRSEN or **vsp2_vsp_vrect_even** for BVRSEN)
- Measured V_{RECT} of the odd half-cycle (**vsp1_vsp_vrect_odd** for VRSEN or **vsp2_vsp_vrect_odd** for BVRSEN)
- Average of the measured even and odd half-cycle V_{RECT} (**vsp1_vsp_vrect** for VRSEN or **vsp2_vsp_vrect** for BVRSEN)

The even and odd V_{RECT} outputs are utilized by flux balancing in the case of FB topology. For other functions, such as telemetry or faults, this separation is unnecessary, and they apply the average output. It should be noted that non-bridge topologies do not have even/odd half-cycles and the output is sampled only on the **vsp_vrect_even** output and thus the average function only passes through even.

The state machine controls the ADC sample timing based on the detected edges of:

- PWM
- V_{RECT}

Therefore, while operating in V_{RECT} mode, the tracking ADC only tracks the input when the input voltage is reflected to the secondary-side V_{RECT}. This occurs when the primary-side PWM signal is high or “on”. Note that the identification of the PWM and SR FET PWM outputs is through registers:

- **ce_on_mask0**
- **ce_on_mask1**
- **ce_off_mask0**
- **ce_off_mask1**

These settings are discussed in Section 7 for PWM.

2.3.2.1 V_{RECT} timing for single PWM signal

The rectified voltage is pulsating, and therefore its measurement is not as straightforward as in the case of V_{OUT}. The measurement cycle timing is initiated when the primary-side PWM signal goes high. At this point, a timer is started and the XDPP1100 waits for a high transition on the VRSEN or BVSEN_BVRSEN input pin. The rising edge of the rectified voltage is detected via a comparator, vrs_comp. It is clocked at 200 MHz and has a programmable threshold via register **vrs_cmp_ref_sel**. When V_{RECT} exceeds this threshold, the comparator enters its hold phase of operation.

Once the transition is detected, the tracking ADC’s DAC is preloaded to its value during the previous cycle before the PWM signal transitioned low. The tracking ADC then waits for a user-programmable time (register **vrs_track_start_thr**) measured from the PWM rising edge. Subsequent to the timer completion, the tracking ADC begins tracking the input voltage and the tracking continues until falling-edge PWM is detected. The current-tracking DAC code is then saved for the next cycle. The complete V_{RECT} measurement cycle timing is shown in Figure 11.

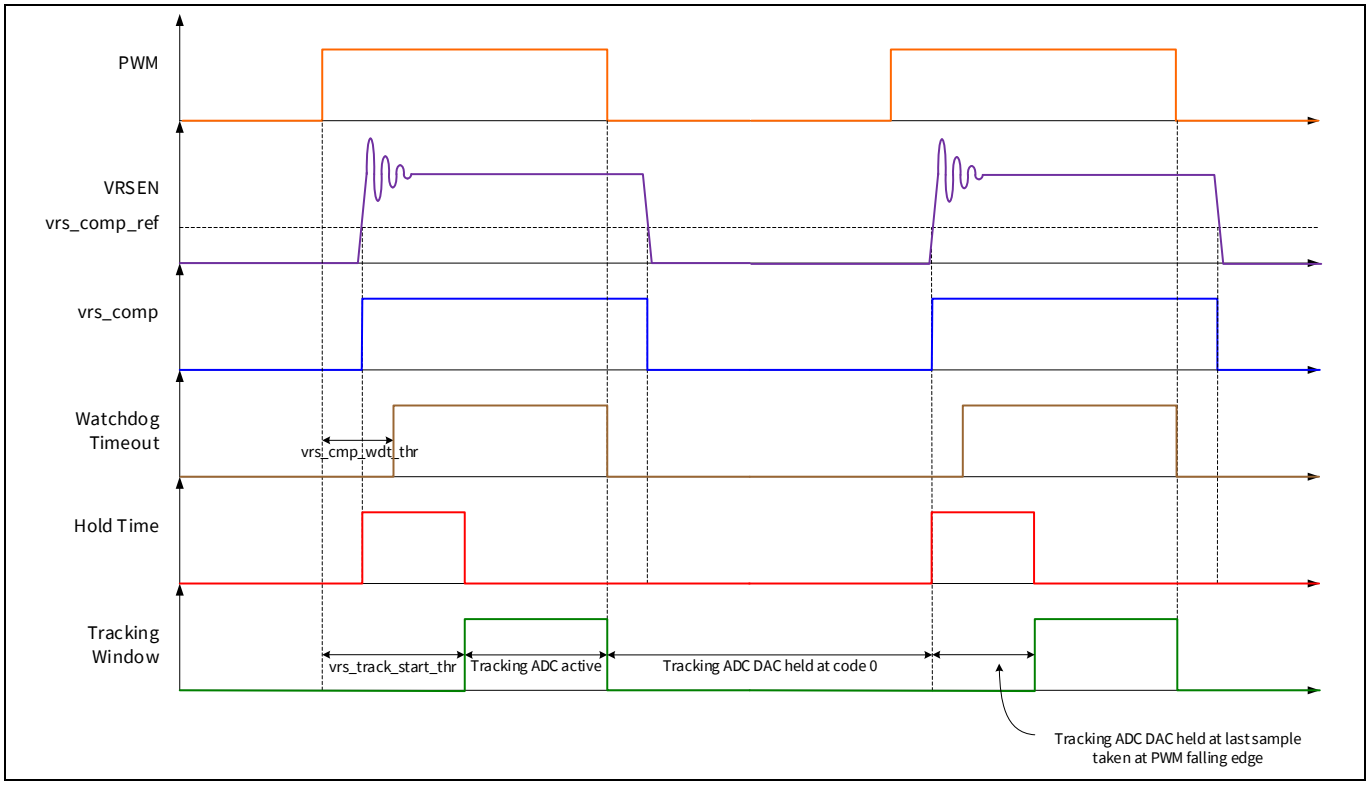

Figure 11 Timing of the V_{RECT} measurement cycle

2.3.2.2 V_{RECT} watchdog timer

For correct V_{RECT} measurement, the input voltage needs to have certain minimum value to trip the vrs_comp comparator. Therefore, a watchdog timer (WDT) is started on the PWM rising edge. This timer monitors the quality of the V_{RECT} signal. The user can define a timeout threshold for the comparator via register **vrs_cmp_wdt_thr**. If no transition is observed at the comparator output after this time, the tracking ADC proceeds by:

- Preloading the previous tracking ADC DAC value
- Waiting for the track start timer to complete

Figure 12 shows the VRECT measurement cycle timing for the case where the vrs_comp transition is not detected.

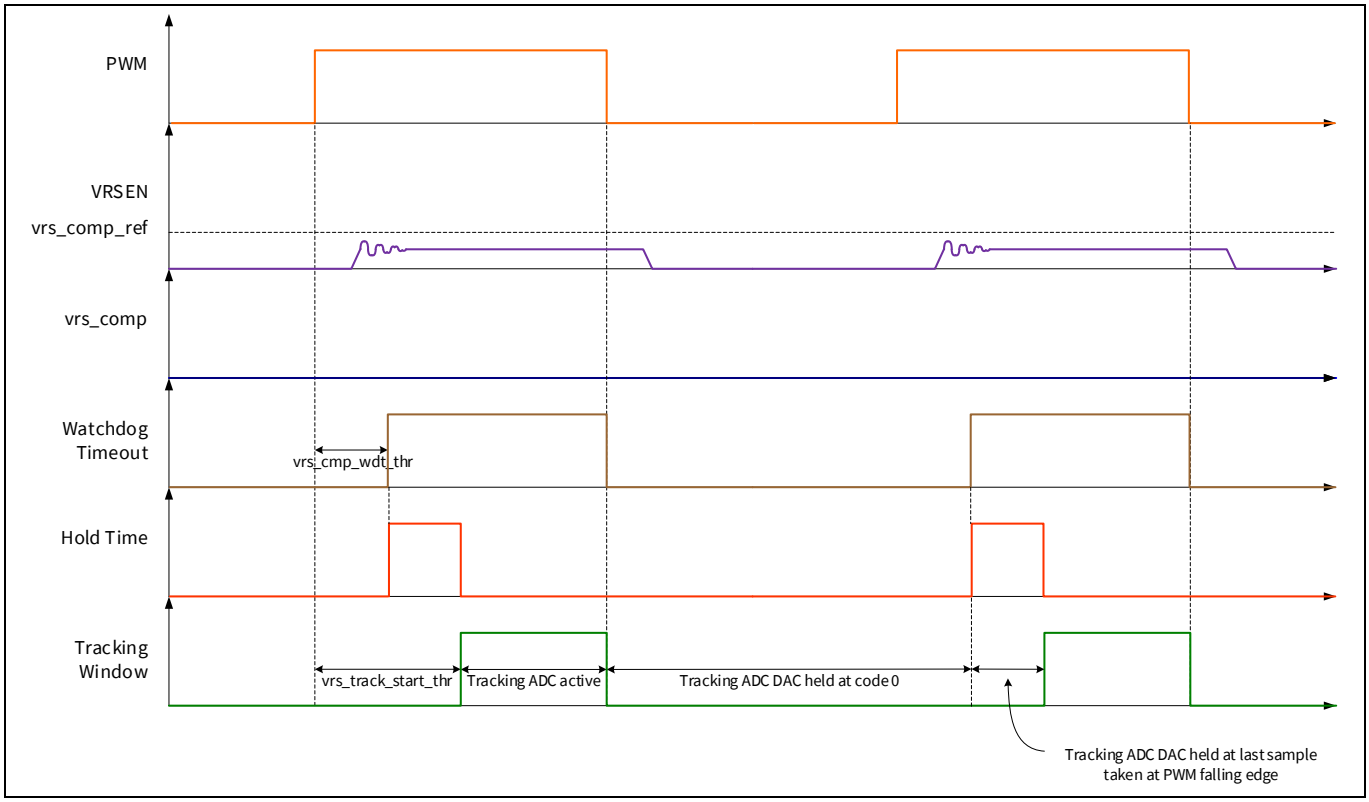

Figure 12 V_{RECT} watchdog timeout

The following timer settings need to be considered:

- Set the WDT count, **vrs_cmp_wdt_thr**, greater than the latest expected arrival time of the V_{RECT} pulse at VRSEN as measured from the PWM output
- Set the tracking start timer count, **vrs_track_start_thr**, to a value at minimum 250 ns greater than the WDT to ensure adequate time for the AFE to settle after the V_{RECT} pulse

If significant ringing is observed on the VRSEN input pin at the V_{RECT} pulse transition, the tracking start timer should be set longer than the expected length of this ringing. Since both timers are started from the PWM rising edge, the tracking start timer should always be larger than the WDT.

2.3.2.3 Deglitcher

The rising and falling edges of the measured V_{RECT} signal can contain noise that might trip the comparator prematurely. A deglitcher is added at the comparator output to prevent spurious signals from triggering the hold phase within the V_{RECT} measurement cycle.

The minimum pulse width (PW) of the deglitcher is programmable via register **vrs_min_pw**. If this deglitch pulse is wider than the noise ringing pulse width, the false triggering is avoidable. It should be noted that if a non-zero PW is programmed, the comparator output is delayed by the same amount, and the tracking start timer should be increased accordingly via register **vrs_track_start_thr**.

2.3.2.4 V_{RECT} timing for two PWM signals

Previously, the V_{RECT} timing was discussed in the case of a single PWM signal (Figure 11 and Figure 12), utilized typically in the ACF converter topology. However, in the bridge topologies, two PWM signals operate on opposite cycles, referred to here as even and odd. XDPP1100 measures and stores the PWM signals separately, as shown in Figure 13.

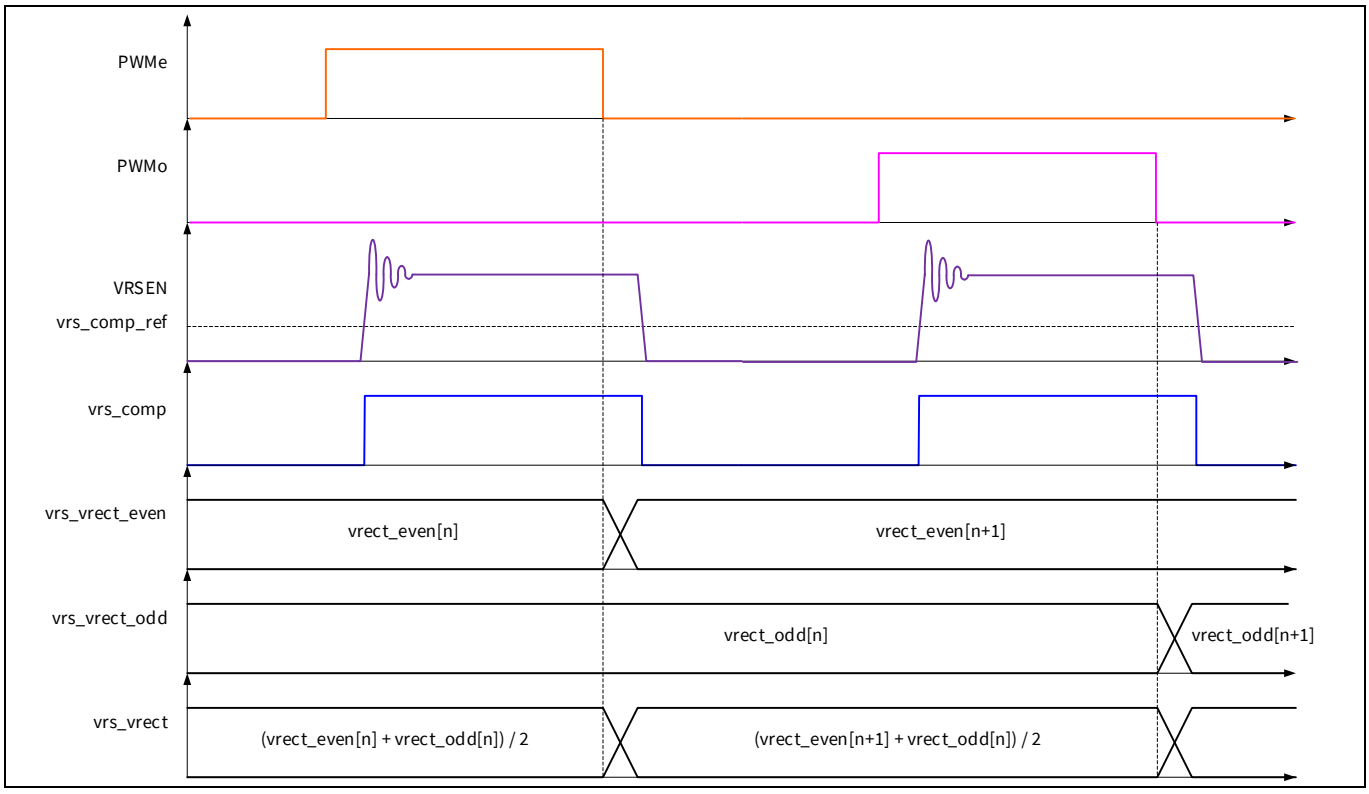

Figure 13 VSP V_{RECT} even and odd cycle timing

As previously discussed, the even and odd V_{RECT} outputs are utilized by flux (volt-second) balancing function, whereas the computed average is used by functions such as telemetry and FF.

2.3.2.5 V_{IN} transient response

The response to input voltage transient is illustrated in Figure 14. If the transient is initiated prior to entering the tracking window, the tracking ADC initially increases its step size to reach the moving input voltage. Then it adjusts the step size downward, if necessary, to maintain the tracking. It might require several switching cycles to complete the tracking to the settled input voltage. For V_{RECT} sense, automatic step size is recommended.

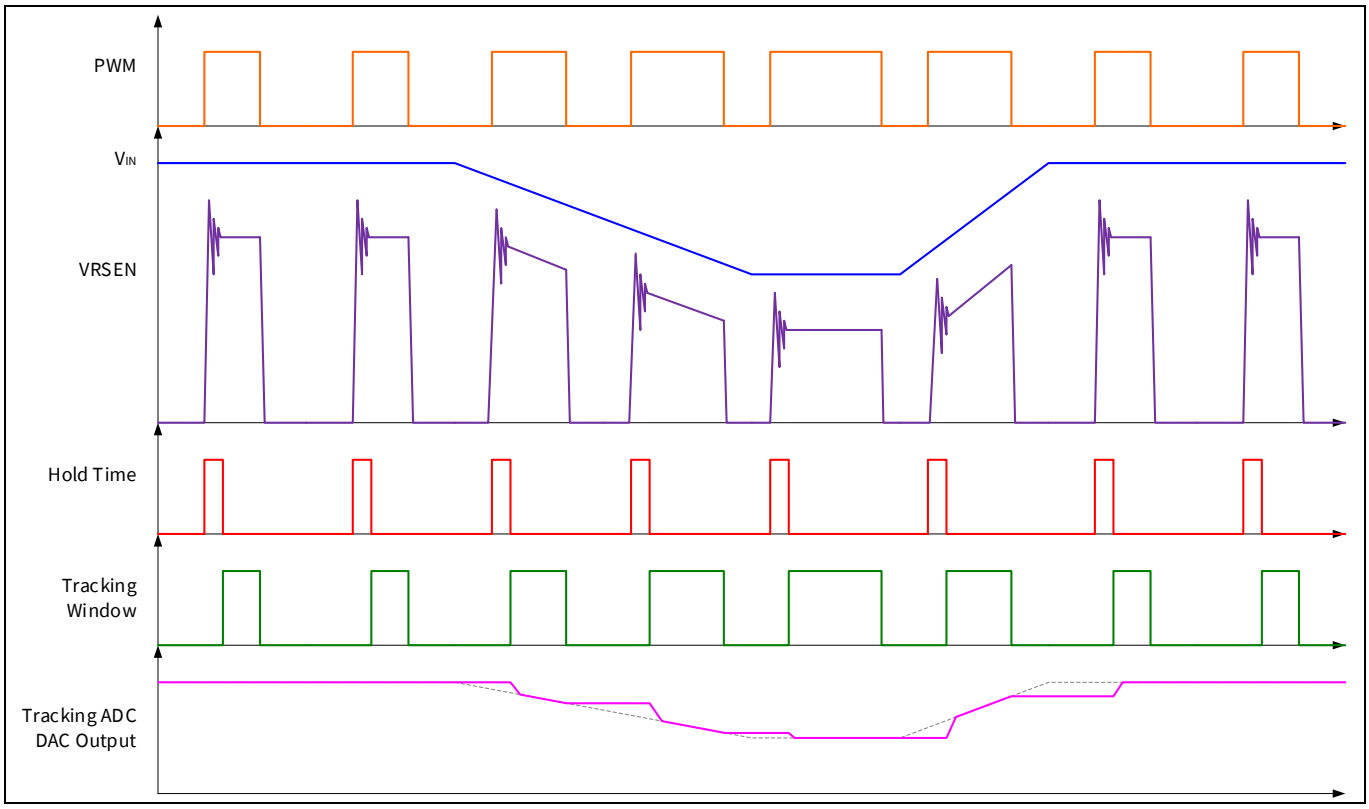

Figure 14 V_{RECT} sense transient

2.3.2.6 V_{RECT} start-up programming

At start-up, prior to the first PWM pulse, there is no V_{RECT} pulse on the secondary to be measured. Therefore, to enable start-up, user-programmable initial voltage (register **vrs_voltage_init**) is provided for:

- FF
- Telemetry
- Fault conditions

Without pre-bias (i.e., output ramps from 0 V), the initial PWM pulses are narrower than the tracking start timer threshold. During this time, the V_{RECT} tracking ADC continues to output the initial voltage set by the user. Once the PWM pulses obtain a width greater than the tracking start timer threshold, the ADC begins to track toward the actual V_{RECT} voltage.

The initial voltage to be set in the register **vrs_voltage_init** depends on the topology. For HB it can be calculated as in Equation (2.2).

$$vrs_voltage_init = \left(\frac{vin_init(V)}{0.04V}\right) * MFR_VRECT_SCALE * MFR_TRANSFORMER_SCALE \quad (2.2)$$

For FB and ACF as in Equation (2.3)

$$vrs_voltage_init = \left(\frac{vin_init(V)}{0.02V}\right) * MFR_VRECT_SCALE * MFR_TRANSFORMER_SCALE \quad (2.3)$$

The terms utilized in the equations are defined as follows:

- Vin_init is the initial input voltage (i.e., 48 V for a 36 V to 72 V system), and it must be greater than the PMBus command VIN_ON setting for start-up to occur
- MFR_VRECT_SCALE is a PMBus command defining the resistor divider ratio between V_{RECT} and the VRSEN input pin
- MFR_TRANSFORMER_SCALE is a PMBus command defining the transformer turns ratio

Figure 15 shows the start-up response of the V_{RECT} measurement.

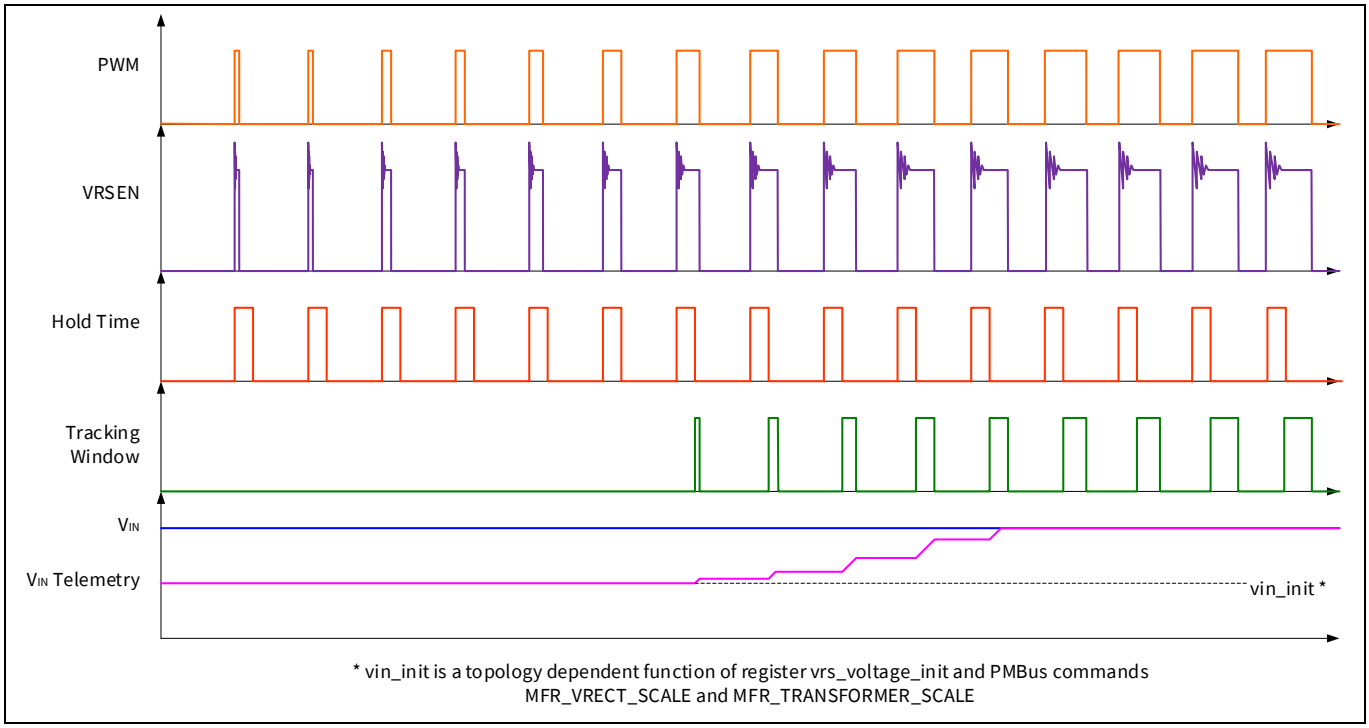

Figure 15 V_{RECT} sense start-up

2.3.2.7 Same cycle response

As a default feature, the sensed rectified voltage is sampled once per cycle at the PWM falling edge. Thus, the current V_{RECT} value does not contribute to the PWM turn-off via FF function. An alternate “same cycle” mode is available and user programmable by:

- Enabling the register **vrs_same_cycle_en**
- Defining the time before the live samples appear on the V_{RECT} output (register **vrs_meas_start_thr**) after entering the tracking window

The same cycle mode is illustrated in Figure 16, where the bottom two waveforms compare the V_{RECT} output with and without same cycle mode. For the best input voltage transient response, it is recommended to enable the same cycle mode.

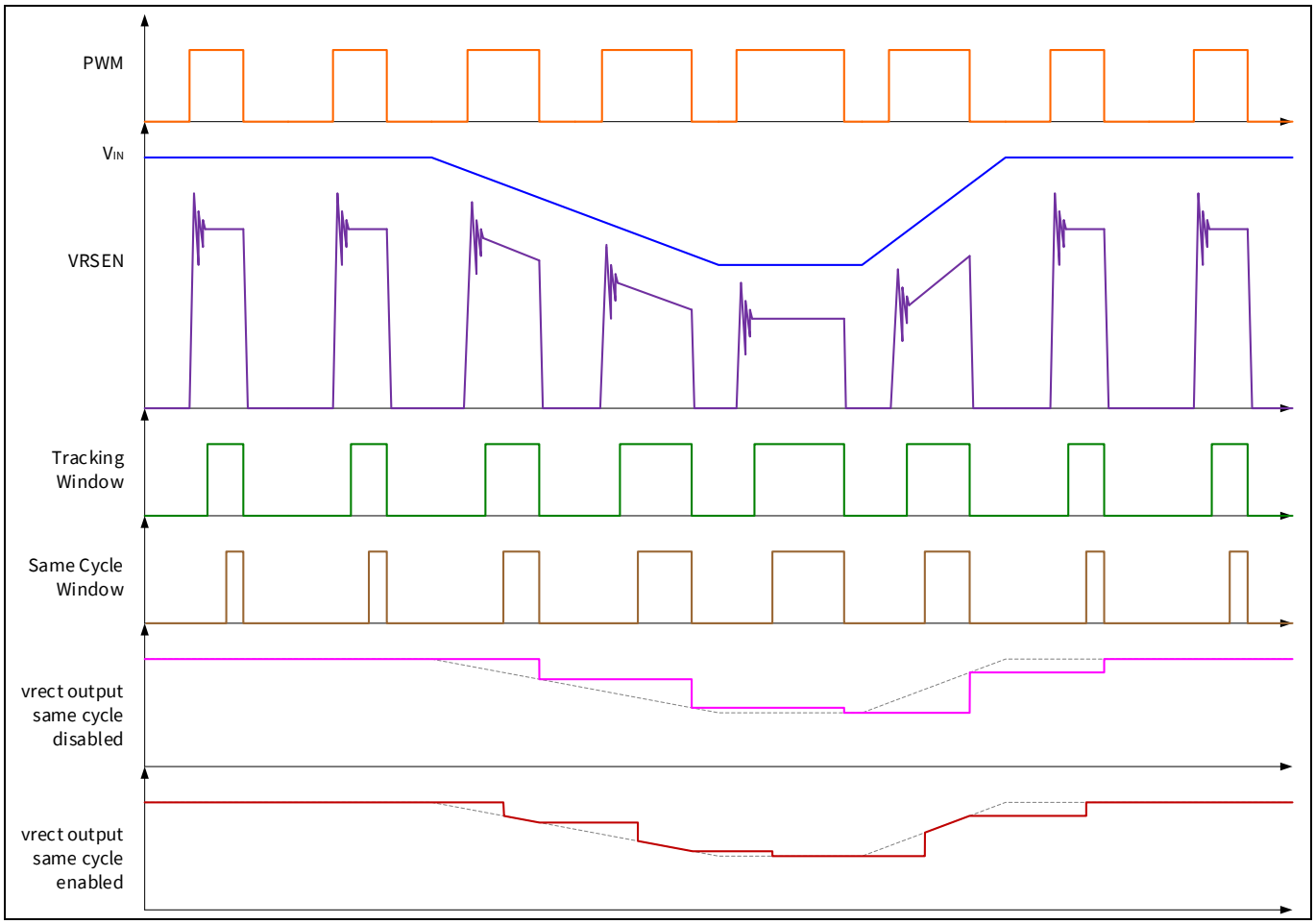

Figure 16 Same cycle mode

2.3.2.8 V_{RECT} delay counters

The V_{RECT} processing function includes a set of counters. They measure the delay between:

- Falling (rising) PWM edge driving the synchronous rectification (SR) FETs
- Rising (falling) edge of the rectification voltage

The rising (falling) edge of V_{RECT} is detected by the vrs_comp comparator at the VRSEN input pin. These timers obtain an indication of the actual (versus programmed) dead time. It should be noted that:

- The counters operate on a 5 ns clock period
- The vrs_comp comparator is clocked on a 5 ns clock period leading to an overall counter accuracy result of ±10 ns

In addition to the delay counters, a third counter measures the rectification voltage pulse width as detected by the vrs_comp comparator on the VRSEN input pin. This measurement result is used by the flux (volt-second) balance function. See Section 12 for additional information.

Two versions of the delay count results are available for each edge of the rectification voltage:

- Block averaged result
- Non-averaged result

Only the non-averaged result is available for the pulse width measurement, though it is separated into even and odd measurements for bridge topologies. Figure 17 shows the timing of the measured waveforms.

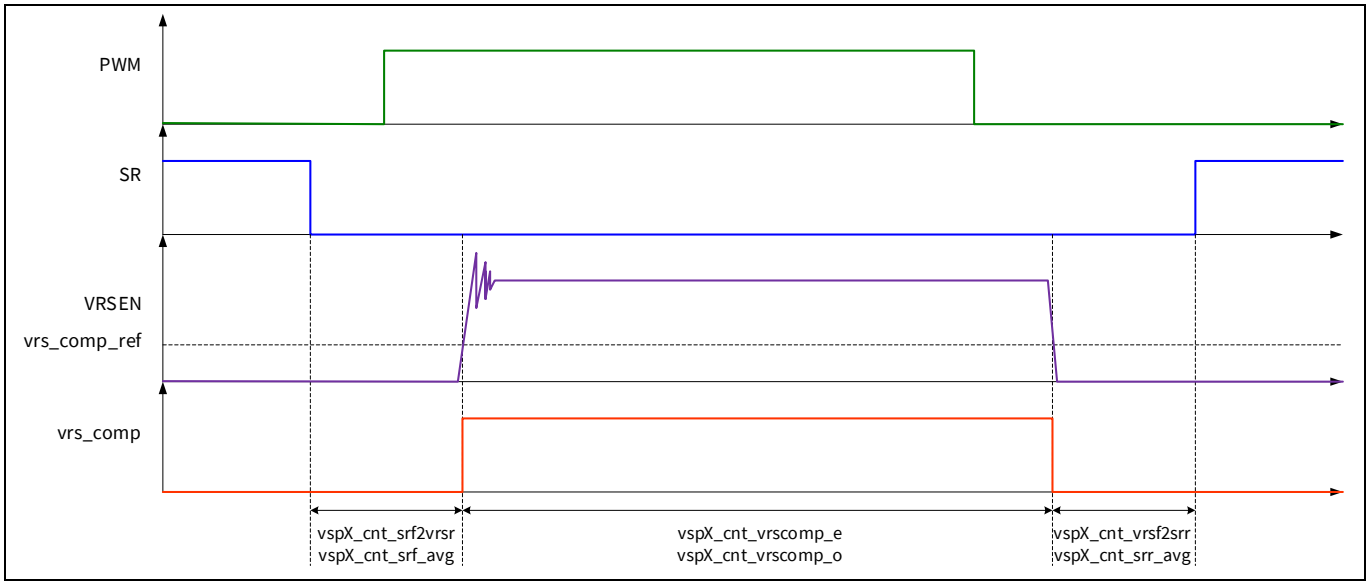

Figure 17 Dead time counters

If the counter detects the incoming waveform edges in the incorrect order, it will indicate this through the following status registers:

- For VS module 1: **vsp1_vrsr_b4_srf** indicates VRSEN detected VRS comp rising edge before SR FET falling edge, and **vsp1_srr_b4_vrsf** indicates VRSEN detected SR FET rising edge before VRS comp falling edge
- For VS module 2: **vsp2_vrsr_b4_srf** indicates BVRSEN detected VRS comp rising edge before SR FET falling edge, and **vsp2_srr_b4_vrsf** indicates BVRSEN detected SR FET rising edge before VRS comp falling edge

2.3.3 V_{IN} processing

The VS modules VS1 and VS2 with sense pins VRSEN and BVRSEN, respectively, can be configured to measure the primary-side input voltage, V_{IN}, directly. This is obtained by:

- Selecting the input voltage source (VRSEN) or (BVRSEN) with register **tlm_vin_src_sel**
- Selecting general-purpose ADC mode (0) via register **vsp1_vrs_sel** for VRSEN, or V_{OUT} sense mode (0) via **vsp2_vrs_sel** for BVRSEN

When the VS module is configured to sense the input voltage from the primary side, the vsp_vrect output is continuous, not pulsating as in the case of sensing V_{RECT}.

2.4 VS registers

All VS-related registers and their descriptions are provided in Table 2.

Table 2 VS-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
analog	vs0_step	RW	7000_0400h	[2:0]	VS0 (VSEN) ADC tracking loop integrator step size when automatic step size disabled. Recommended setting is 1 for highest resolution
analog	vs0_step_en	RW	7000_0400h	[3]	VS0 (VSEN) tracking loop step size control. 0: Automatic step size 1: Use vs0_step[2:0] (recommended for VSEN)
analog	vs1_step	RW	7000_0400h	[6:4]	VS1 (VRSEN) tracking loop step size when automatic step size disabled. Recommended setting is 1 for highest resolution
analog	vs1_step_en	RW	7000_0400h	[7]	VS1 (VRSEN) tracking loop step size control. 0: Automatic step size (recommended for V_{RECT} sense) 1: Use vs1_step[2:0] (recommended for V_{IN} sense)
analog	vs2_step	RW	7000_0400h	[10:8]	VS2 (BVSEN_BVRSEN) tracking loop step size when automatic step size disabled. Recommended setting is 1 for highest resolution
analog	vs2_step_en	RW	7000_0400h	[11]	VS2 (BVSEN_BVRSEN) tracking loop step size control. 0: Automatic step size (recommended for V_{RECT} sense) 1: Use vs2_step[2:0] (recommended for V_{OUT}, V_{IN} sense)
vsen	vsp_verrn_clamp_thresh	RW	7000_0800h (VSEN) 7000_1000h (BVSEN)	[2:0]	V_{errn} clamp threshold at PID input where V_{errn} is defined as the control voltage minus the sensed output voltage. Ignoring droop, V_{errn} = VOUT_COMMAND*VOUT_SCALE_LOOP - V(VSEN) 0: Clamp range = -40 to 40 mV 1: Clamp range = -80 to 80 mV 2: Clamp range = -120 to 120 mV

Peripheral	Field name	Access	Address	Bits	Description
					3: Clamp range = -160 to 160 mV 4: Clamp range = -200 to 200 mV 5: Clamp range = -240 to 240 mV 6: Clamp range = -280 to 280 mV 7: Clamp range = -318.75 to 318.75 mV
vsen	vsp_en_fe_comp_lp	RW	7000_0800h (VSEN) 7000_0C00h (VRSEN) 7000_1000h (BVSEN)	[6]	VS FEC enable. The FEC circuit is used to reduce long-term offset drift at the input to the VSADC. 0: Disabled 1: Enabled
vsen	vsp_fe_count_thr	RW	7000_0800h (VSEN) 7000_0C00h (VRSEN) 7000_1000h (BVSEN)	[16:7]	Defines variable part of VS FEC period. LSB = 10 ns, range = 0 to 10230 ns
vsen	vsp_mode_fe_1p25m	RW	7000_0800h (VSEN) 7000_0C00h (VRSEN) 7000_1000h (BVSEN)	[17]	Defines VS FEC DAC LSB weight. 0: 0.625 mV 1: 1.250 mV
vsen	vsp_count_fe_cmp	RW	7000_0800h (VSEN) 7000_0C00h (VRSEN) 7000_1000h (BVSEN)	[18]	Defines the number of consecutive VS FEC cycles with the comparator in the same polarity required to increment or decrement the compensation DAC. Cycles = vsp_count_fe_cmp + 1
vsen	vsp_adc_blank	RW	7000_0804h (VSEN) 7000_0C04h (VRSEN) 7000_1004h (BVSEN)	[0]	When high, holds the previous output from the VSADC. This may be used by FW to stall but not zero out the ADC output for further downstream processing
vsen	vsp_vout_fs	R	7000_0810h (VSEN) 7000_0C10h (VRSEN) 7000_1010h (BVSEN)	[11:0]	Gain and offset trimmed VSADC output. LSB = 1.25 mV, range = 0.0 to 2.1 V

Peripheral	Field name	Access	Address	Bits	Description
common	vrs_cmp_wdt_thr	RW	7000_3018h	[9:0]	V_{RECT} comparator watchdog timeout threshold. The WDT measures from the rising edge of the PWMs indicated in ceX_on_mask0 and ceX_on_mask1 (where X = 0, 1). If V_{RECT} has not tripped the comparator by the time the timeout threshold is reached, it is assumed V_{RECT} is below the comparator threshold and the V_{RECT} sense will enter its hold phase. This threshold should be set to a value greater than the expected time for V_{RECT} to go high but less than the tracking start threshold defined by vrs_track_start_thr. This threshold is shared by the VRSEN and BVRSEN sense paths. LSB = 10 ns, range = 0.0 to 10.23 µs
common	vrs_track_start_thr	RW	7000_3018h	[19:10]	V_{RECT} tracking start time threshold. This threshold is compared against the same timer used by vrs_cmp_wdt_thr, which is started on the rising edge of the PWMs indicated in ceX_on_mask0 and ceX_on_mask1 (where X = 0, 1). When the timer exceeds this threshold the V_{RECT} sense moves from its hold phase to its tracking phase. This threshold should be set approximately 250 ns beyond the expected time of arrival of the V_{RECT} rising edge to ensure adequate settling of the VS AFE. This threshold is shared by VRSEN and BVRSEN sense paths. LSB = 10 ns, range = 0.0 to 10.23 µs
common	vrect_div_2_sel	RW	7000_3018h	[20]	Defines the equation used for rectification voltage (V_{RECT})

Peripheral	Field name	Access	Address	Bits	Description
					measurement. 0: vrect_even + vrect_odd, used for non-bridge topologies 1: (vrect_even + vrect_odd) / 2, used for bridge topologies
common	vsp1_vrs_sel	RW	7000_3018h	[21]	VRSEN input ADC rectification VS (VRS) mode select. 0: General-purpose ADC mode 1: V_{RECT} sense (VRS) mode
common	vsp2_vrs_sel	RW	7000_3018h	[22]	BVSEN_BVRSEN ADC rectification VRS mode select. 0: V_{OUT} sense (VS) mode 1: VRS mode
common	vsp1_vrs_cnt_num_avg	RW	7000_3018h	[24:23]	Defines the number of samples in the VRSEN dead time measurement block averagers. The dead time measurements include SR FET falling edge to VRS comp rising edge and VRS comp falling edge to SR FET rising edge. 0: 4 samples 1: 8 samples 2: 16 samples 3: 32 samples
common	vsp2_vrs_cnt_num_avg	RW	7000_3018h	[26:25]	Defines the number of samples in the BVRSEN dead time measurement block averagers. The dead time measurements include SR FET falling edge to VRS comp rising edge and VRS comp falling edge to SR FET rising edge. 0: 4 samples 1: 8 samples 2: 16 samples 3: 32 samples
common	vrs_cmp_ref_sel	RW	7000_3018h	[27]	VRS comparator threshold select. This threshold is shared by the VRSEN and BVRSEN sense paths. When the rectification voltage exceeds this threshold, the VRS enters its hold phase of operation. 0: 500 mV 1: 300 mV

Peripheral	Field name	Access	Address	Bits	Description
common	vrs_delta_vsum_sel	RW	7000_3018h	[28]	In the VRS tracking phase, the ADC output is filtered by both a fast (higher bandwidth) LPF and a slow (lower bandwidth) LPF. When the difference between the two LPF outputs is greater than the threshold selected by vrs_delta_vsum_sel, the fast filter output is selected, otherwise the slow filter output is selected. 0: 8 ADC codes 1: 16 ADC codes
common	vrs_bypass_slow_lpf	RW	7000_3018h	[29]	VRS slow filter bypass control. Shared by VRSEN and BVRSEN paths. 0: Filter not bypassed (recommended) 1: Filter bypassed
common	vrs_bypass_fast_lpf	RW	7000_3018h	[30]	VRS fast filter bypass control. Shared by VRSEN and BVRSEN paths. 0: Filter not bypassed (recommended) 1: Filter bypassed
common	vrs_voltage_init	RW	7000_301Ch	[7:0]	Initial voltage for VRSEN and BVRSEN tracking integrators when operating in VRS mode as observed at the VRSEN/BVRSEN inputs (i.e., after the sense resistor divider). This setting must be greater than the rectification voltage corresponding to VIN_ON for controller start-up when a rectification voltage is selected for the input voltage telemetry input source by tlm_vin_src_sel. LSB = 20 mV, range = 0.0 to 2.1 V
common	vrs_same_cycle_en	RW	7000_301Ch	[8]	V_{RECT} same cycle mode enable. When enabled, live V_{RECT} updates will begin vrs_meas_start_thr samples after entering tracking mode

Peripheral	Field name	Access	Address	Bits	Description
					for faster FF response. Otherwise V_{RECT} is only updated on the falling PWM edge. 0: Same cycle mode disabled 1: Same cycle mode enabled
common	vrs_min_pw	RW	7000_301Ch	[11:9]	V_{RECT} pulse width counter minimum pulse width. Pulses less than this width are considered glitches and are ignored. LSB = 20 ns, range = 0 to 140 ns
common	vrs_meas_start_thr	RW	7000_301Ch	[16:12]	When V_{RECT} same cycle measurement is enabled by vrs_same_cycle_en, this parameter defines the number of samples after vrs_track_start_thr before live updates are enabled. LSB = 40 ns, range = 0 to 1240 ns
common	vsp1_vrs_vrect_even	R	7000_3034h	[11:0]	Measured VRSEN rectification voltage on the even half-cycle. LSB = 1.25 mV, range = 0.0 to 2.1 V
common	vsp1_vrs_vrect_odd	R	7000_3038h	[11:0]	Measured VRSEN rectification voltage on the odd half-cycle. LSB = 1.25 mV, range = 0.0 to 2.1 V
common	vsp1_vrs_vrect	R	7000_303Ch	[11:0]	Average of measured VRSEN even and odd half-cycle rectification voltages. LSB = 1.25 mV, range = 0.0 to 2.1 V
common	vsp2_vrs_vrect_even	R	7000_3040h	[11:0]	Measured BVRSEN rectification voltage on the even half-cycle. LSB = 1.25 mV, range = 0.0 to 2.1 V
common	vsp2_vrs_vrect_odd	R	7000_3044h	[11:0]	Measured BVRSEN rectification voltage on the odd half-cycle. LSB = 1.25 mV, range = 0.0 to 2.1 V
common	vsp2_vrs_vrect	R	7000_3048h	[11:0]	Average of measured BVRSEN even and odd half-cycle rectification voltages. LSB = 1.25 mV, range = 0.0 to 2.1 V

Peripheral	Field name	Access	Address	Bits	Description
common	vsp1_cnt_srf_avg	R	7000_3060h	[7:0]	Block averaged VRSEN SR FET falling edge to VRS comp rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp1_cnt_srr_avg	R	7000_3060h	[15:8]	Block averaged VRSEN VRS comp falling edge to SR FET rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp1_cnt_srf2vrsr	R	7000_3060h	[23:16]	Non-averaged VRSEN SR FET falling edge to VRS comp rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp1_cnt_vrsf2srr	R	7000_3060h	[31:24]	Non-averaged VRSEN VRS comp falling edge to SR FET rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp1_cnt_vrscomp_e	R	7000_3064h	[10:0]	Non-averaged VRSEN VRS comp pulse width measurement result for ACF topology or the even half-cycle of bridge topologies. LSB = 5 ns, range = 0 to 10235 ns
common	vsp1_cnt_vrscomp_o	R	7000_3064h	[21:11]	Non-averaged VRSEN VRS comp pulse width measurement result for the odd half-cycle of bridge topologies. LSB = 5 ns, range = 0 to 10235 ns
common	vsp1_srr_b4_vrsf	R	7000_3064h	[22]	Status flag indicating VRSEN detected SR FET rising edge before VRS comp falling edge (i.e., no dead time on PWM falling edge) on previous cycle. 0: SR FET rising edge after VRS comp falling edge 1: SR FET rising edge before VRS comp falling edge
common	vsp1_vrsr_b4_srf	R	7000_3064h	[23]	Status flag indicating VRSEN detected VRS comp rising edge before SR FET falling edge (i.e., no dead time on PWM rising edge) on previous cycle. 0: VRS comp rising edge after

Peripheral	Field name	Access	Address	Bits	Description
					SR FET falling edge 1: VRS comp rising edge before SR FET falling edge
common	vsp2_cnt_srf_avg	R	7000_3068h	[7:0]	Block averaged BVRSEN SR FET falling edge to VRS comp rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp2_cnt_srr_avg	R	7000_3068h	[15:8]	Block averaged BVRSEN VRS comp falling edge to SR FET rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp2_cnt_srf2vrsr	R	7000_3068h	[23:16]	Non-averaged BVRSEN SR FET falling edge to VRS comp rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp2_cnt_vrsf2srr	R	7000_3068h	[31:24]	Non-averaged BVRSEN VRS comp falling edge to SR FET rising edge measurement result. LSB = 5 ns, range = 0 to 1275 ns
common	vsp2_cnt_vrscomp_e	R	7000_306Ch	[10:0]	Non-averaged BVRSEN VRS comp pulse width measurement result for ACF topology or the even half-cycle of bridge topologies. LSB = 5 ns, range = 0 to 10235 ns
common	vsp2_cnt_vrscomp_o	R	7000_306Ch	[21:11]	Non-averaged BVRSEN VRS comp pulse width measurement result for the odd half-cycle of bridge topologies. LSB = 5 ns, range = 0 to 10235 ns
common	vsp2_srr_b4_vrsf	R	7000_306Ch	[22]	Status flag indicating BVRSEN detected SR FET rising edge before VRS comp falling edge (i.e., no dead time on PWM falling edge) on previous cycle. 0: SR FET rising edge after VRS comp falling edge 1: SR FET rising edge before VRS comp falling edge
common	vsp2_vrsr_b4_srf	R	7000_306Ch	[23]	Status flag indicating BVRSEN detected VRS comp rising edge

Peripheral	Field name	Access	Address	Bits	Description
					before SR FET falling edge (i.e., no dead time on PWM rising edge) on previous cycle. 0: VRS comp rising edge after SR FET falling edge 1: VRS comp rising edge before SR FET falling edge
telem	tlm_vin_src_sel	RW	7000_3400h 7000_3800h	[30:28]	Input voltage telemetry source select. 0: VRSEN, secondary V_{RECT} sense, vrs_voltage_init prior to start-up 1: BVSEN_BVRSEN, secondary V_{RECT} sense, vrs_voltage_init prior to start-up 2: Loop 0 V_{OUT}, select on Loop 1 when Loop 1 V_{IN} provided by Loop 0 V_{OUT} (e.g., post-buck) 3: PRISEN, non-pulsed/primary V_{IN} sense via telemetry ADC 4: tlm_vin_force, forced V_{IN} via FW (e.g., FW override of HW computation) 5: VRSEN, secondary V_{RECT} sense, 0 V prior to start-up, select on Loop 1 when sharing Loop 0 V_{RECT} sense 6: VRSEN: non-pulsed/primary V_{IN} sense 7: BVSEN_BVRSEN, non-pulsed/primary V_{IN} sense

3 Current sense (IS)

This chapter describes the current sense module and its submodules in detail. In addition, the user-programmable settings for configuring different current sense features are described and relevant registers provided.

Different applications require different current information. The XDPP1100 controller supports both primary and secondary-side current sensing in order to obtain information on I_{IN} and I_{OUT}, respectively. It has two input pin pairs for current sensing:

- ISEN/IREF
- BISEN/BIREF

Generally, either input pin pair can be used for both input and output current sensing. However, there are a few application-related limitations. Below are the current sensing rules:

- In a single-loop system for VMC, either input pin pair can be used for secondary-side or primary-side current sense
- In a single-loop system for PCMC, the ISEN/IREF pin pair must be connected to the current being controlled; I_{OUT} for secondary-side PCMC and I_{IN} for primary-side PCMC
- In a dual-loop system using either VMC or PCMC, the ISEN/IREF pin pair must be used for the first loop (Loop 0) and the BISEN/BIREF pin pair must be employed for the second loop (Loop 1)

A full-bridge full-bridge (FBFB) rectifier utilizing primary-side PCMC is an example of an application where both currents I_{IN} and I_{OUT} are sensed. Typical connection of this converter is shown in Figure 18, where the current sensing is emphasized in red. The primary-side current is sensed through a current transformer and fed to the ISEN/IREF pin pair. The I_{OUT} is sensed through a shunt and fed to BISEN/BIREF pin pair. In addition to the PCMC, the primary-side current information is used for I_{IN} telemetry and overcurrent fault protection. The secondary-side current information is employed for various functionalities, e.g., I_{OUT} telemetry and fault protections, droop control and current sharing.

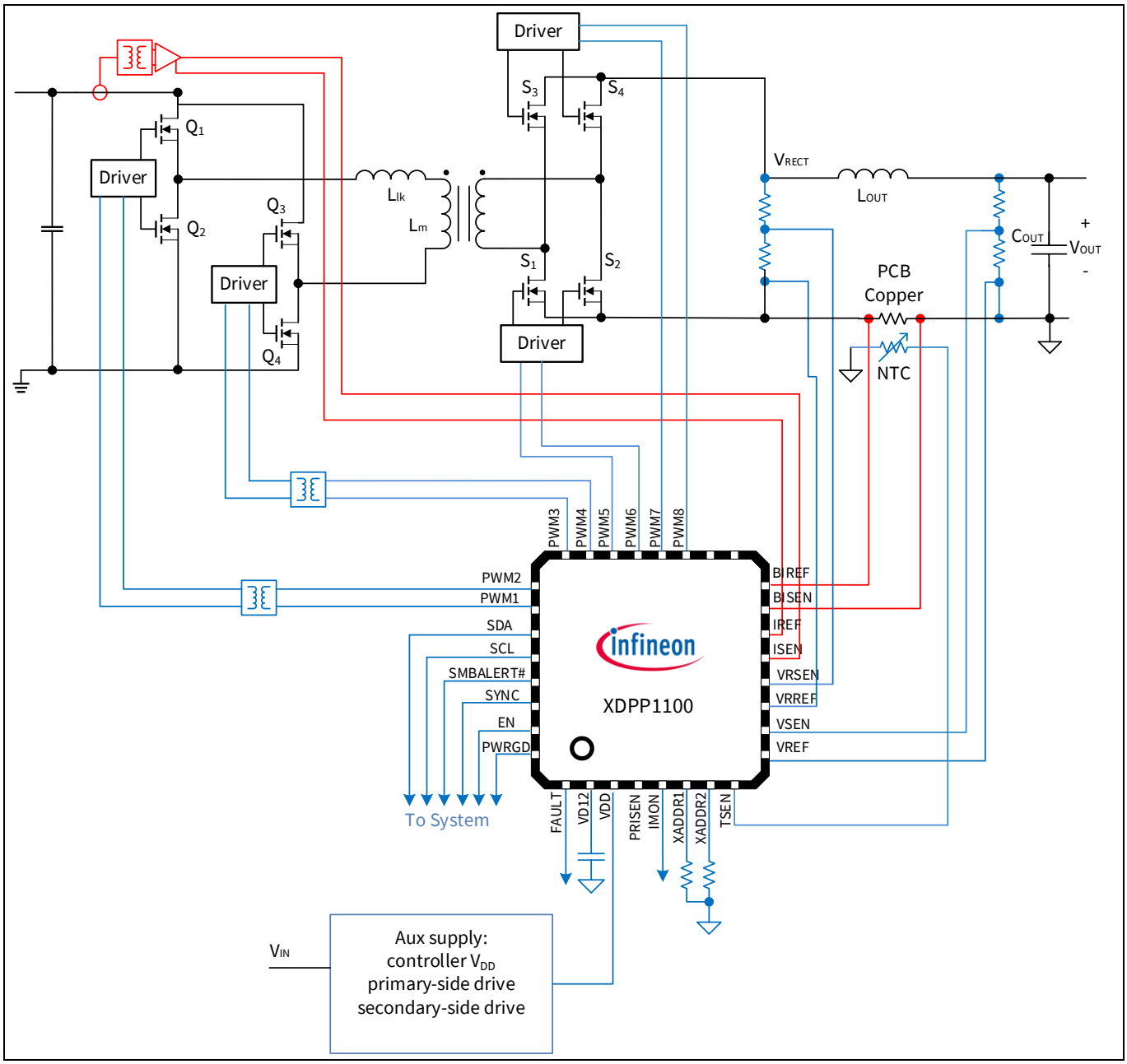

Figure 18 PCMC FBFB rectifier with primary and secondary current sensing

3.1 Current sense module configuration

The XDPP1100 controller contains two current sense modules which are connected to the analog input pins ISEN/IREF and BISEN/BIREF. A simplified block diagram of these current sense modules is shown in Figure 19. Each module consists of three submodules:

- IS AFE
- Current estimator (CE)
- Current sense processor (ISP)

Submodules IS AFE and CE form the current sense ADC (IADC).

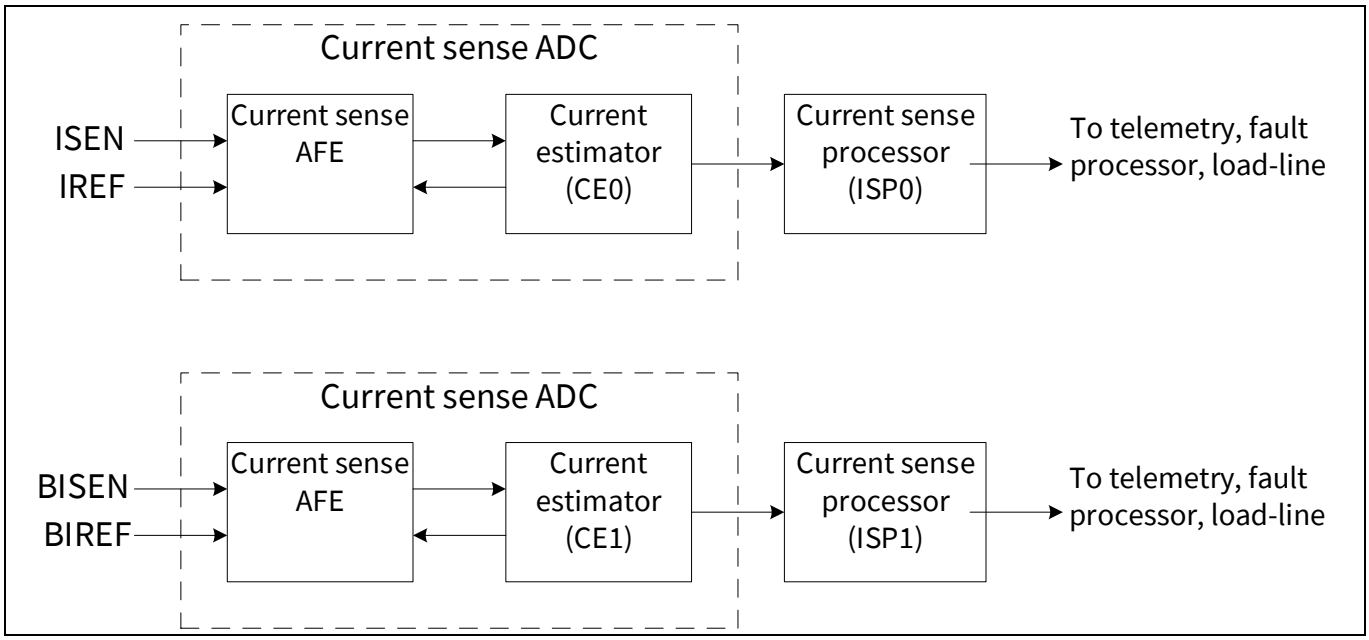

Figure 19 Simplified block diagram of the current sensing

The current sense information is provided to ISEN/IREF and BISEN/BIREF input pins as a voltage that is proportional to the sensed current. There are different methods to obtain the current information, depending on the application requirements. For the XDPP1100 controller, the supported current sensing methods are:

- Shunt resistor
- PCB trace
- Current transformer (isolated topologies)
- Integrated power stage (IPS) with integrated current sense

The main features and purposes of these methods are summarized in Table 3.

Table 3 Current sensing methods

Sensing method	Advantages	Disadvantages	Applications
Shunt resistor	• Most common solution • Simple	• BOM cost • Additional power dissipation	• All applications
PCB trace	• No additional bill of materials (BOM) cost • No extra power dissipation	• Higher temperature variation must be compensated • Need for onboard trimming for accuracy • Potential for higher noise-to-signal ratio; layout is critical	• High-current applications
Current transformer	• Simple method	• Additional BOM	• Primary current sensing in isolated converters

Sensing method	Advantages	Disadvantages	Applications
Integrated current sense of IPS	• Board space saving • Improved current sensing accuracy	• Integrated power stage only supports non-isolated buck topology	• Non-isolated buck converters

A shunt resistor is suitable for all applications, whereas the PCB trace sensing is recommended only for high-current applications. If PCB trace sensing is applied in low-current applications, there could be various drawbacks:

- Increased noise due to longer PCB trace that is needed for sufficient resistance
- High-gain amplifier stage, which might impose bandwidth (BW) limitations

Current transformer is the simplest method to transfer primary current information to the controller on the secondary side. In non-isolated topologies it is possible to sense the primary current as long as the current sense voltage is referenced to the supported reference levels (GND or 1.2 V), as will be described later in this chapter. The main requirement for using the integrated current sensing of IPS is to have 1.2 V reference voltage for current sense. An example of supported integrated power stage is Infineon's IR3555A.

Depending on the selected sensing method, the IADC resolution as well as the reference level need to be adjusted, as will be discussed in the following sections.

3.2 Current sense analog to digital converter

The IADC is a tracking ADC, and the two IADCs in the current sense module are identical. They comprise two main sub-blocks:

- AFE
- CE

A simplified block diagram of the IADC is shown in Figure 20, where the submodules are emphasized as dashed lines. This ADC has configurable LSB weight and effective conversion rate of 25 MHz. The subsections that follow describe the ADC resolution configuration depending on the sensing method utilized as well as the CE functionality in more detail.

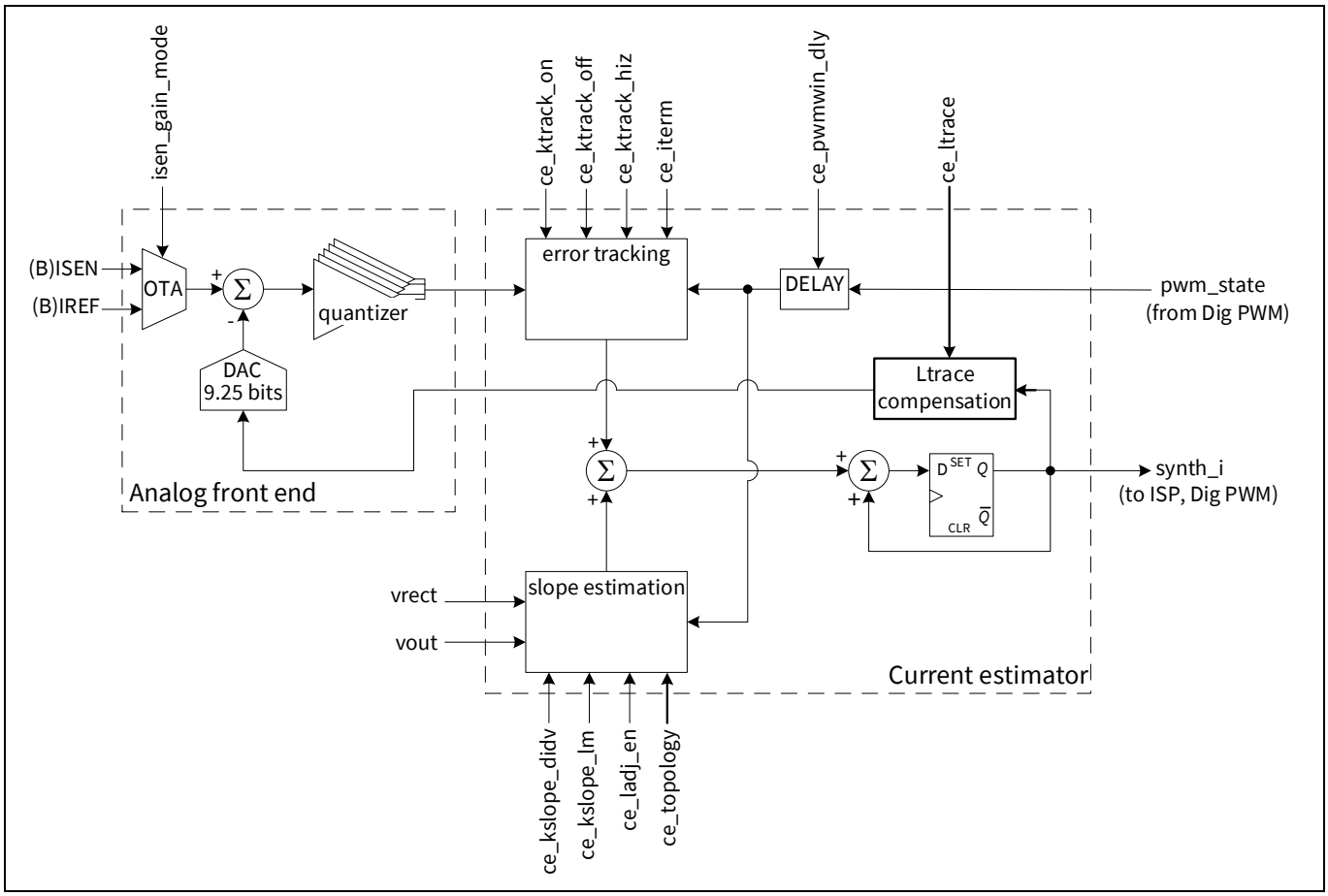

Figure 20 Simplified block diagram of the IADC

3.2.1 Current sense analog front end

The IS AFE block diagram was illustrated in Figure 20, inside the dashed box. It comprises of four sub-blocks:

- Programmable gain operational transconductance amplifier (OTA)
- 9.25-bit current DAC
- Summing node
- Quantizer

The first sub-block, programmable gain OTA, converts the current sense voltage to a current that is summed with the DAC current. The gain level for the OTA is programmable via register **isen_gain_mode** and shown in Table 4 as well as the reference level for current sense voltage. The X in the register name denotes 1 for ISEN/IREF and 2 for BISEN/BIREF.

Table 4 IS AFE gain modes

isenX_gain_mode	ADC LSB	V(IREF)	Gain mode
0	Reserved, not to be used		
1	100 µV	GND	High gain
2	1.45 mV	GND	Low gain
3	1.45 mV	1.2 V	Low gain

The high-gain mode is intended for applications where the voltage is directly sensed on the sensing element (either shunt or PCB trace). The low-gain mode is intended to be used with the external operational amplifier-based gain stage or IPS. All other applications require the reference level to be ground, except IPS. High-gain mode refers to smaller LSB size and finer resolution. This implies higher noise susceptibility, whereas low-gain mode means higher LSB size and lower noise sensitivity.

Depending on the application current level and other requirements, the proper method for the current sensing should be selected. Table 5 summarizes possible current sensing methods for different applications and the corresponding current gain modes.

Table 5 Current sense method vs. gain mode selection

Application	Current sense method	Current gain mode
High current	PCB trace	High-gain mode (1)
	PCB trace with external amplifier	Low-gain mode (2)
	Small shunt resistor with external amplifier	Low-gain mode (2)
Low and medium current	Small shunt	High-gain mode (1)
	Small shunt with external amplifier	Low-gain mode (2)
	Larger shunt resistor	Low-gain mode (2)
Primary-side current sensing	Current transformer	Low-gain mode (2)
Non-isolated buck with IPS	Integrated current sense	Low-gain mode (3)

The second sub-block, the 9.25-bit DAC, has 640 quantization levels. However, some of the bits are used for offset correction and the effective bits are less, depending on the OTA gain setting as shown in Table 4. The input voltage range per LSB weight is shown in Table 6.

Table 6 Input voltage range as a function of LSB weight

ADC LSB	Min.	Max.	Units
100 µV	–22	22	mV
1.45 mV	–280	395	mV

The last two sub-blocks have the following functionality:

- The summing node subtracts the DAC current from the OTA output current and converts to voltage for use by the quantizer
- The quantizer consists of five comparators, which generate a 6-level (±1, ±3, ±5) error tracking signal used by the CE to correct current estimation

3.2.2 Current estimator

The CE provides a digital reconstruction of the secondary-side inductor current or the primary-side current, depending on the current sense input pin configuration. The shape of this current is obtained through slope estimation function. For that purpose, the following application-related information is required:

- Output inductor value
- Primary-side magnetizing inductance
- Measured values of V_{OUT} and V_{RECT}
- Tracking error feedback from the AFE

The reconstructed current from the CE drives the AFE DAC in order to center the quantizer at the currently estimated current value. It is also processed downstream by the ISP to be used by the telemetry and fault processing functions. In addition, this current is applied in PCMC to compare against the reference current to determine the PWM turn-off time.

A simplified block diagram of the CE was shown in Figure 20, where it is emphasized with dashed lines. It has four main functions:

- PWM state delay function
- Slope estimator
- Error tracking function
- Parasitic inductance L_{trace}

The following subsections describe the PWM state and slope estimator, as well as error tracking and parasitic inductance estimation in more detail, emphasizing the user-definable parameters. In this section, various CE-related register names begin as **ceX_**, where the X is:

- X = 0 for ISEN/IREF input pins
- X = 1 for BISEN/BIREF input pins

Thus, depending on which current sensing pins are used, the programming needs to be performed accordingly.

3.2.2.1 PWM state

The XDPP1100 controller continuously predicts the current phase and ripple based on the state of the PWM pulse. Therefore, the PWM state is critical to both slope estimation and error tracking, because it is used to determine the equation for voltage across the inductor in order to estimate the inductor current slope.

The PWM has three states with respect to the inductor current cycle for a buck-derived isolated topology, as illustrated in Figure 21. These states are:

- **On-state:** inductor current rising slope when the PWM FET is on
- **Off-state:** inductor current falling slope when the PWM FET is off and the SR FET is on
- High impedance (HIZ) state: inductor current slope when both switches are off; in case of positive current, the slope is falling toward the zero and for negative current the slope is increasing toward the zero

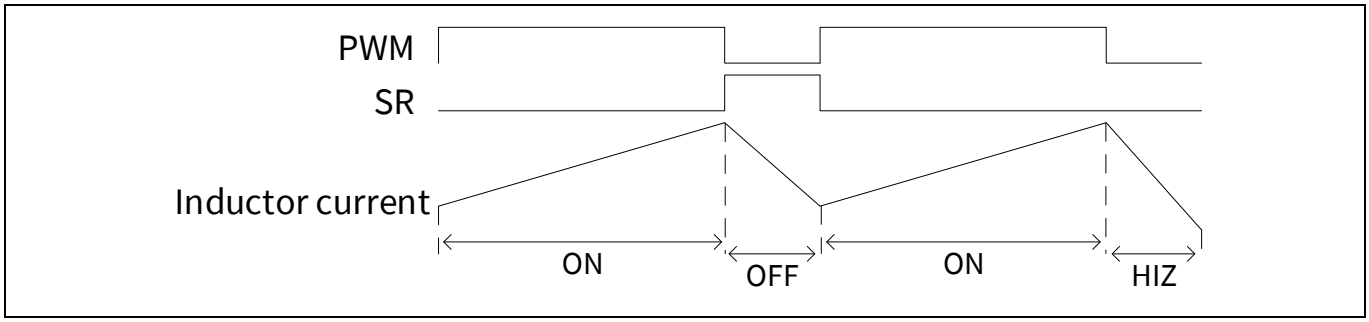

Figure 21 PWM state definition with respect to the inductor current cycle

In the actual system, there is delay between the internal PWM state and the actual PWM state observed from the output inductor. In order to better align the internal state to the sensed current waveform, the internal PWM can be delayed. This is possible via register **ce_pwmwin_dly**. While adjusting this parameter, all possible delay sources (e.g., controller output latency, isolator latency, driver latency) within the system should be considered. It should be noted that an exact unit-by-unit delay match is not required, as the error tracking corrects minor timing mismatches at the PWM state transitions.

Depending on the application, there are different number of FETs that define the on- and off-times in a certain topology. Each FET is controlled by a PWM signal, and for current reconstruction purposes the FETs contributing to the PWM state are relevant for the state programming. The PWM states are programmed for ISEN and BISEN input pins via the following registers:

- For defining ISEN on-time: **ce0_on_mask0[11:0]** and **ce0_on_mask1[11:0]**
- For defining ISEN off-time: **ce0_off_mask0[11:0]** and **ce0_off_mask1[11:0]**
- For defining BISEN on-time: **ce1_on_mask0[11:0]** and **ce1_on_mask1[11:0]**
- For defining BISEN off-time: **ce1_off_mask0[11:0]** and **ce1_off_mask1[11:0]**

Each register consists of 12 bits, where each bit field corresponds to a specific PWM output. For instance, in the above registers, bit [n] corresponds to the pin PWM n + 1 so that:

- Bit [0] corresponds to pin PWM1
- Bit [11] corresponds to pin PWM12

The bit corresponding to the specific PWM output that drives a FET that defines the on- or off-time needs to be set to 1 in the relevant register. In bridge topologies there are two on-/off-states per switching cycle; therefore, mask0 defines the first on-/off-state and mask1 the second. In non-bridge topologies (e.g., ACF, buck) mask1 is typically set to the same value as mask0. The PWM state programming is described below in more detail through various topology examples.

The XDPP1100 GUI can be used to automatically program the on-/off-states, based on topology selection and PWM mapping.

PWM state programming examples

The PWM state programming is described in detail for different topologies. The following topologies are considered:

- Buck (a)
- ACF (b)
- HBCT (c)
- Half-bridge current-doubler (HBCD) (d)
- FBFB (e)

For each application, a simplified topology diagram is provided, highlighting the relevant FETs that define the on- and off-times. The detailed register programming is described in a table, where:

- The first column defines the relevant switch
- The second column is the selected PWM output that drives the FET
- The third column describes which bit fields in a certain register need to be set to 1
- The last column provides the resulting register value

a) Buck topology

The simplified buck converter topology is shown in Figure 22, where the relevant FETs are Q1, defining the inductor current on-time, and Q2, defining the inductor current off-time. Table 7 describes the relevant programming for this topology.

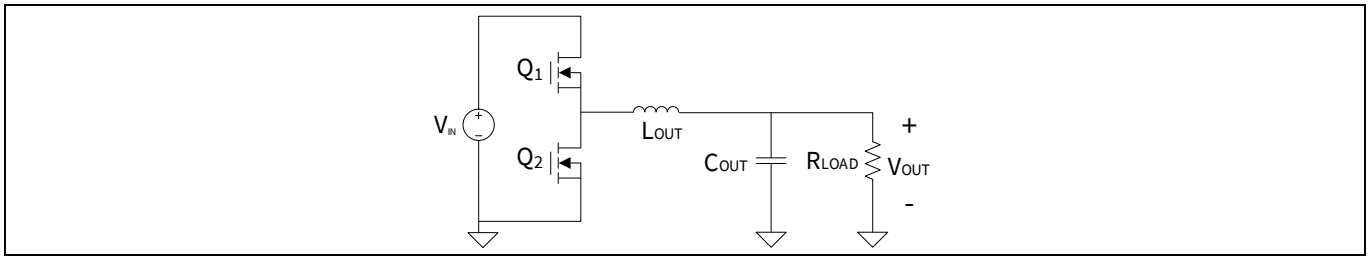

Figure 22 Buck topology FET naming

Table 7 Buck topology PWM state programming example

FET	PWM output	Register programming	Register value
Q1	PWM1	Set bit field values corresponding PWM1 to 1 and other bit fields to 0 in **ceX_on_mask0** and **ceX_on_mask1**	**ceX_on_mask0** = 001h **ceX_on_mask1** = 001h
Q2	PWM3	Set bit field values corresponding PWM3 to 1 and other bit fields to 0 in **ceX_off_mask0** and **ceX_off_mask1**	**ceX_off_mask0** = 004h **ceX_off_mask1** = 004h

b) ACF topology

The ACF topology is shown in Figure 23. For this topology, the relevant FETs are Q1, defining the inductor current on-time, and the synchronous rectification switch SR2, defining the inductor current off-time. The other two FETs also require a PWM control signal but are not relevant for specifying the PWM state for current reconstruction. Table 8 describes the relevant programming for this topology.

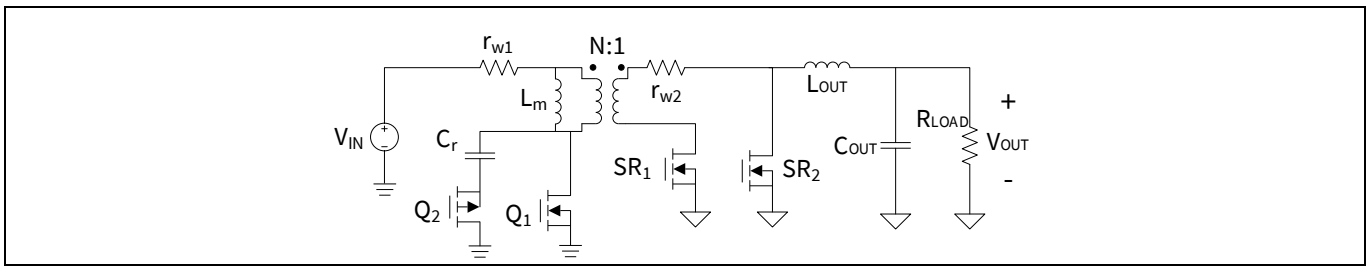

Figure 23 ACF topology FET naming

Table 8 ACF PWM state programming example

FET	PWM output	Register programming	Register value
Q1	PWM2	Set bit field values corresponding PWM2 to 1 and other bit fields to 0 in **ceX_on_mask0** and **ceX_on_mask1**	**ceX_on_mask0** = 002h **ceX_on_mask1** = 002h
SR2	PWM5	Set bit field values corresponding PWM5 to 1 and other bit fields to 0 in **ceX_off_mask0** and **ceX_off_mask1**	**ceX_off_mask0** = 010h **ceX_off_mask1** = 010h

c) HBCT topology

The HBCT topology is shown in Figure 24. In bridge topologies there are two on- and off-states per switching cycle, and thus two FETs contribute on the on-and off-time. For HBCT, Q1 defines the on-time for the first half-switching cycle and Q2 for the second half. The off-time for the first half of the switching cycle is defined by SR2 and the second by SR1. Table 9 describes the relevant programming for this topology.

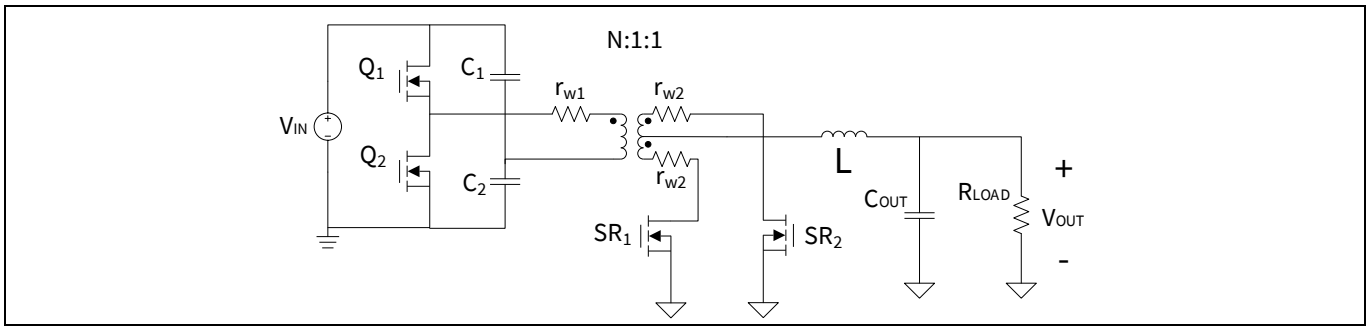

Figure 24 HBCT topology FET naming

Table 9 HBCT PWM state programming example

FET	PWM output	Register programming	Register value
Q1	PWM1	Set bit field values corresponding PWM1 to 1 and other bit fields to 0 in **ceX_on_mask0**	**ceX_on_mask0** = 001h
Q2	PWM2	Set bit field values corresponding PWM2 to 1 and other bit fields to 0 in **ceX_on_mask1**	**ceX_on_mask1** = 002h
SR1	PWM3	Set bit field values corresponding PWM3 to 1 and other bit fields to 0 in **ceX_off_mask1**	**ceX_off_mask1** = 004h
SR2	PWM4	Set bit field values corresponding PWM4 to 1 and other bit fields to 0 in **ceX_off_mask0**	**ceX_off_mask0** = 008h

d) HBCD topology

The HBCD topology adds an inductor to the secondary in order to double the maximum available current. The simplified topology is shown in Figure 25, where Q1 defines the on-time for inductor L1, and Q2 defines the inductor L2 on-time. The off-time for L1 is defined by SR1, and correspondingly the off-time for L2 is specified by SR2. Table 10 describes the relevant programming for this topology.

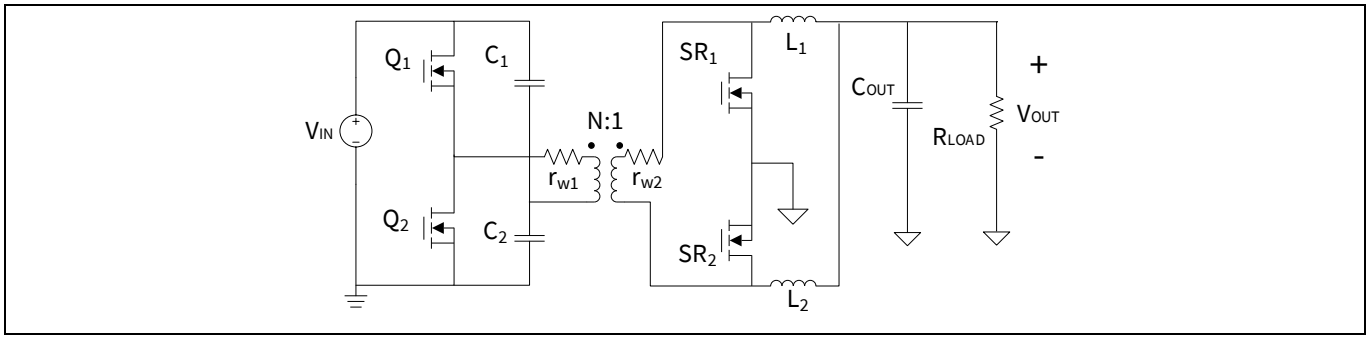

Figure 25 HBCD FET naming

Table 10 HBCD PWM state programming example

FET	PWM output	Register programming	Register value
Q1	PWM1	Set bit field values corresponding PWM1 to 1 and other bit fields to 0 in **ceX_on_mask0**	**ceX_on_mask0** = 001h
Q2	PWM2	Set bit field values corresponding PWM2 to 1 and other bit fields to 0 in **ceX_on_mask1**	**ceX_on_mask1** = 002h
SR1	PWM3	Set bit field values corresponding PWM3 to 1 and other bit fields to 0 in **ceX_off_mask0**	**ceX_off_mask0** = 004h
SR2	PWM4	Set bit field values corresponding PWM4 to 1 and other bit fields to 0 in **ceX_off_mask1**	**ceX_off_mask1** = 008h

e) FBFB topology

The last example is the FBFB rectifier, and its simplified topology is shown in Figure 26. The FETs Q1 and Q3 define the inductor on-time for the first half-switching cycle and Q2 and Q4 define the second switching cycle on-time. The synchronous rectification switches SR2 and SR4 define the inductor off-time for the first half-switching cycle and correspondingly SR1 and SR3 define the second half. Table 11 describes the relevant programming for this topology.

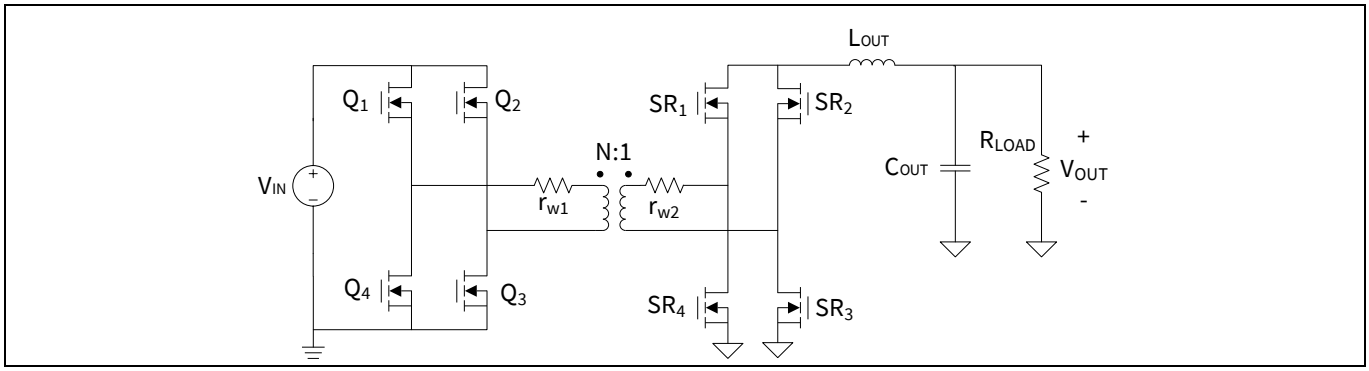

Figure 26 FBFB topology FET naming

Table 11 FBFB topology PWM state programming example

FET	PWM output	Register programming	Register value
Q1	PWM1	Set bit field values corresponding PWM1 and PWM3 to 1 and other bit fields to 0 in **ceX_on_mask0**	**ceX_on_mask0** = 005h
Q2	PWM2	Set bit field values corresponding PWM2 and PWM4 to 1 and other bit fields to 0 in **ceX_on_mask1**	**ceX_on_mask1** = 00Ah
Q3	PWM3	Set bit field values corresponding PWM1 and PWM3 to 1 and other bit fields to 0 in **ceX_on_mask0**	**ceX_on_mask0** = 005h
Q4	PWM4	Set bit field values corresponding PWM2 and PWM4 to 1 and other bit fields to 0 in **ceX_on_mask1**	**ceX_on_mask1** = 00Ah
SR1	PWM9	Set bit field values corresponding PWM9 and PWM11 to 1 and other bit fields to 0 in **ceX_off_mask1**	**ceX_off_mask1** = 500h

FET	PWM output	Register programming	Register value
SR2	PWM10	Set bit field values corresponding PWM10 and PWM12 to 1 and other bit fields to 0 in **ceX_off_mask0**	**ceX_off_mask0** = A00h
SR3	PWM11	Set bit field values corresponding PWM9 and PWM11 to 1 and other bit fields to 0 in **ceX_off_mask1**	**ceX_off_mask1** = 500h
SR4	PWM12	Set bit field values corresponding PWM10 and PWM12 to 1 and other bit fields to 0 in **ceX_off_mask0**	**ceX_off_mask0** = A00h

3.2.2.2 Slope estimator

The slope estimation provides the current shape based on the PWM timing and relevant system parameters. It is based on the general inductor equation given in (3.1).

$$V_L = L\frac{di}{dt} \tag{3.1}$$

From this equation, the expression for the current is obtained as given in (3.2) for the slope estimator.

$$dc = \frac{di}{APC} = V\frac{dt}{L*APC} \tag{3.2}$$

Where:

- Dc is “delta code” referenced to the IADC LSB (current in amps = ADC codes * APC)
- APC is “amps per code”, and it maps the IADC LSB to amps, as defined by PMBus parameter MFR_IOUT_APC and discussed in more detail in Section 3.3.1
- Inductor voltage consists of the measured voltages V_{OUT}, V_{IN}, and/or V_{RECT}
- Normalized current slope $\frac{dt}{L*APC}$ is defined by the parameter **ceX_kslope_didv**

The definition for the current slope depends on whether the current is sensed on the primary or secondary side, and on the topology. The relevant parameters for slope specification are described below, after inductor voltage definition.

Inductor voltage

The voltage across the inductor is obtained based on the measured voltages V_{OUT}, V_{IN}, and/or V_{RECT}, depending on the topology and the PWM state. The XDPP1100 controller supports inductor voltage equations for buck-derived, as well as boost and buck-boost topologies. The topology is selected via register **ceX_topology**. The supported topologies and their corresponding inductor voltage equations for different PWM states are shown in Table 12.

Table 12 Topology selection for CE

ce_topology	Topologies	V_{ON} equation	V_{OFF} equation	V_{HIZ} equation
0	Buck, ACF, HB, FB	V_{RECT} - V_{OUT}	-V_{OUT}	-V_{OUT} - V_{body}
1	Boost	V_{RECT}	V_{RECT} - V_{OUT}	V_{RECT} - V_{OUT} - V_{body}
2,3	Buck-boost	V_{RECT}	-V_{OUT}	-V_{OUT} - V_{body}

The voltage V_{body} refers to the voltage across the body diode and it is hardcoded to 0.5 V. The measured voltages V_{OUT} and V_{RECT} can be selected via the following registers:

- **ceX_vout_sel** for V_{OUT} selection
- **ceX_vrect_sel** for V_{RECT} selection

Table 13 and Table 14 provide the different voltage source options for V_{OUT} and V_{RECT}, where X = 0 refers to ISEN/IREF input pins and 1 refers to BISEN/BIREF. The “CDR” in Table 13 indicates current doubler topology on the secondary.

Table 13 CE V_{OUT} source

ceX_vout_sel	CE0 (ISEN) associated V_{OUT}	CE1 (BISEN) associated V_{OUT}
0	Loop 0 V_{OUT}	Loop 0 V_{OUT}
1	Loop 0 V_{OUT} CDR	Loop 0 V_{OUT} CDR
2	Loop 0 V_{OUT}	Loop 1 V_{OUT}
3	Loop 0 V_{OUT} CDR	Loop 1 V_{OUT} CDR

Table 14 CE V_{RECT} source

ceX_vrect_sel	CE0 (ISEN) associated V_{RECT}	CE1 (BISEN) associated V_{RECT}
0	VS1 (VRSEN/VRREF)	VS1 (VRSEN/VRREF)
1	VS2 (BVRSEN/BVRREF)	VS2 (BVRSEN/BVRREF)
2	pid_ff_vrect_override	pid_ff_vrect_override
3	PRISEN	PRISEN
4	VS1 (VRSEN/VRREF)	Loop 0 V_{OUT}
5	VS2 (BVRSEN/BVRREF)	Loop 0 V_{OUT}
6	pid_ff_vrect_override	Loop 0 V_{OUT}
7	PRISEN	Loop 0 V_{OUT}

In a dual-loop system:

- The ISEN/IREF pin pair must be used to sense the current for Loop 0, whereas the BISEN/BIREF pin pair can be assigned for Loop 1, as shown in Table 13
- Loop 0 V_{OUT} is allowed to be used as input voltage for Loop 1, as shown in Table 14

Current slope for secondary-side current sense

The slope estimation for buck-derived topologies (e.g., buck, forward, ACF, HB, FB) with secondary-side current sense can be set through register **ceX_kslope_didv**. The normalized current slope is defined according to Equation (3.3), where the kslope_real is given in Equation (3.4) for secondary-side current sense.

$$ceX_kslope_didv = INT(kslope_real * 2^{13}) \quad (3.3)$$

$$kslope_{real} = 1.0\,V * \frac{10ns}{L_{OUT} * APC} \quad (3.4)$$

L_{OUT} is the output inductance in nH, and APC is the “amps per code” in amps as defined by PMBus command MFR_IOUT_APC. However, in real application the output inductance has a certain tolerance, and therefore, its contribution to **ceX_kslope_didv** has a variation. This tolerance can be corrected by:

- The error tracking part of the CE
- Setting parameter **ceX_ladj_en** to “1”, which enables the autocorrect for errors in **ceX_kslope_didv** based on the incoming current sense waveform

In addition to the inductance tolerance, the inductor value may vary with respect to the current. This variation is typically non-linear, and the XDPP1100 provides a linear correction as illustrated in Figure 27. Register **ceX_dt_l_slope** defines the L slope dependence on the current. As illustrated in the figure, the compensation is specified by:

- **First point:** Use inductance value at 0 A
- **Second point:** User definable based on the best fit curve for each application

Therefore, assuming L_0 is the inductance at zero current, the parameter **ceX_dt_l_slope** is computed according to Equation (3.5).

$$ceX_dt_I_slope = INT\left(2^{14} * \left(\left(\frac{L_0}{L_1}\right) - 1\right) * \frac{APC}{I_1}\right), \quad (3.5)$$

Where:

- L_0 = inductance at 0 A
- L_1 = inductance at user-selected I_1
- APC = amps per code from MFR_IOUT_APC

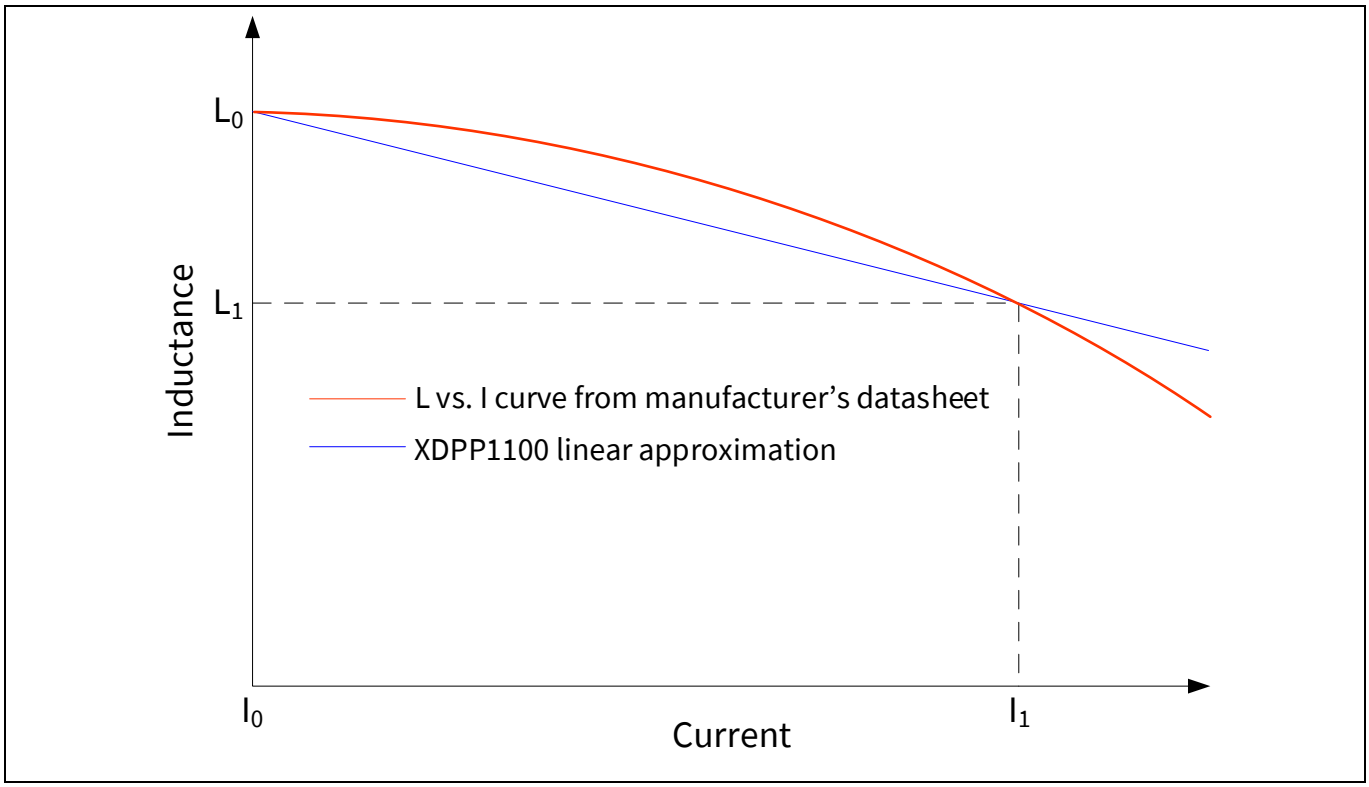

Figure 27 Inductance variation with the current

Current slope for primary-side current sense

The primary-side and secondary-side current waveforms are different, as illustrated in Figure 28. The continuous secondary-side current can be sensed during any PWM state. However, the primary-side current can be sensed only while PWM is in the on-state. This means that in the off- and HIZ states, tracking through the ADC is not possible. Therefore, the tracking should be disabled by setting both register values, **ce_ktrack_off** and **ce_ktrack_hiz**, to zero. The slope estimator correctly places the DAC at the beginning of the next on-state.

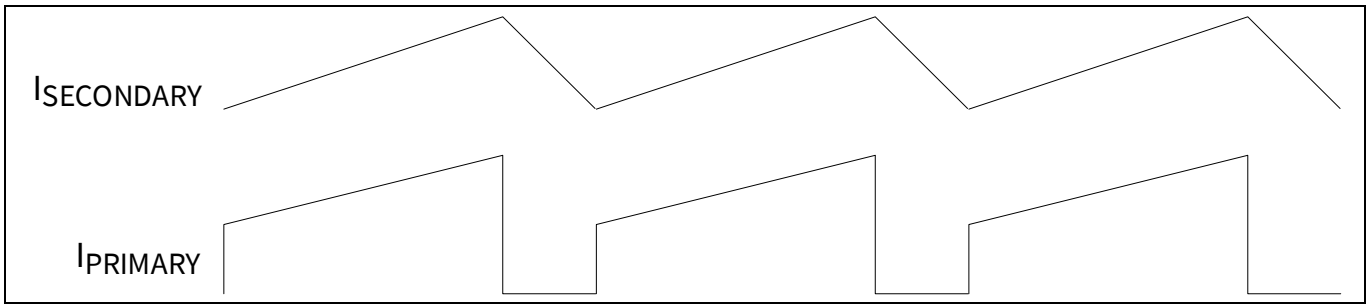

Figure 28 Primary-side and secondary-side current waveforms

While determining the slope for the slope estimator, besides L_{OUT} the magnetizing inductance must also be considered. Therefore, Equation (3.3) is modified to consider the transformer turns ratio by computing the parameter kslope_real according to Equation (3.6).

$$kslope_real = \frac{1.0V*10ns}{N_{turn}*L_{OUT}*APC} \quad (3.6)$$

The transformer magnetizing inductance contribution can be defined via register **ceX_kslope_lm** according to Equation (3.7), where the parameter kslope_lm_real is given in Equation (3.8).

$$ceX_kslope_lm = INT(kslope_lm_real * 2^{13}) \quad (3.7)$$

$$kslope_lm_real = \frac{1.0V*10ns*N_{turn}}{L_m*APC} \quad (3.8)$$

Where N_{turn} is the transformer turns ratio and L_m is the magnetizing inductance.

The register **ceX_ps_current_emu** indicates which current is being sensed, and while primary-side current is chosen, this register needs to be set to 1. In case both primary- and secondary-side current sensing is required in a single-loop topology, PMBus command FW_CONFIG_REGULATION can be used by setting bit [1] EN_PRIM_ISENSE to “1” in order to enable both the ISEN and BISEN current sense paths simultaneously. It should be noted that the registers **ce0_ps_current_emu** and **ce1_ps_current_emu** also need to be configured to indicate which input pin pair (ISEN/IREF or BISEN/BIREF) is used for primary and which is used for secondary current sensing.

Boost and buck-boost topology current sense

For applications other than buck-derived, the correct topology for the CE should be selected according to Table 12. The **ceX_topology** should be set to boost or buck-boost. In case of boost topology, the inductor current can be measured only during the on-state. Therefore, the tracking must be disabled during the off- and HIZ states. This can be set by selecting:

- **ceX_ktrack_off** = 0
- **ceX_ktrack_hiz** = 0

3.2.2.3 Error tracking

The error tracking block is essentially a gain block with independent gains for the on-, off- and HIZ states. As shown in Figure 20:

- Tracking ADC feedback is provided from the AFE quantizer output through the error tracking function
- Error tracking output is summed with the slope estimator output and fed back to the AFE DAC

The tracking gain defines the relative strength of the IADC-based correction term, which is applied to the emulated current waveform. The current sense tracking gains can be defined through registers:

- **ceX_ktrack_on**, for on-state
- **ceX_ktrack_off**, for off-state
- **ceX_ktrack_hiz**, for high impedance state

In addition, register **ceX_iterm** defines an integral coefficient which is applied across all states if set to a non-zero value. Setting this current sense tracking error integrator coefficient to 0 disables the error accumulation.

The PWM transition might cause noise on the current sense signal. Therefore, certain parameters exist that are used to blank the ADC feedback. The following registers can be used to define the transition time windows:

- **ceX_blank_ne_dly** defines the time subsequent to the PWM on-state negative edge during which the error tracking output is blanked to 0 (i.e., the gain is set to 0)
- **ceX_blank_pe_dly** defines the time after a PWM on-state positive edge during which the error tracking output is blanked to 0

3.2.2.4 Trace inductance of the PCB current sensing

The PCB trace resistance can be used for current sensing. In the ideal case the parasitic inductance would be zero, and the current sense voltage shape would be a clear sawtooth waveform. However, in practice this trace inductance is not zero and it introduces a step to the current sense waveform, as indicated by Equation (3.9).

$$V_{SEN} = R_{PCB} * I + L_{PCB} * \frac{dI}{dt} \quad (3.9)$$

Where $\frac{dI}{dt}$ is a function of V_{RECT}, V_{OUT} and L_{OUT}. Figure 29 shows the current sense voltage waveforms for the theoretical case L = 0 (V_{RPCB} in red) and L ≠ 0 (V_{SEN} in blue).

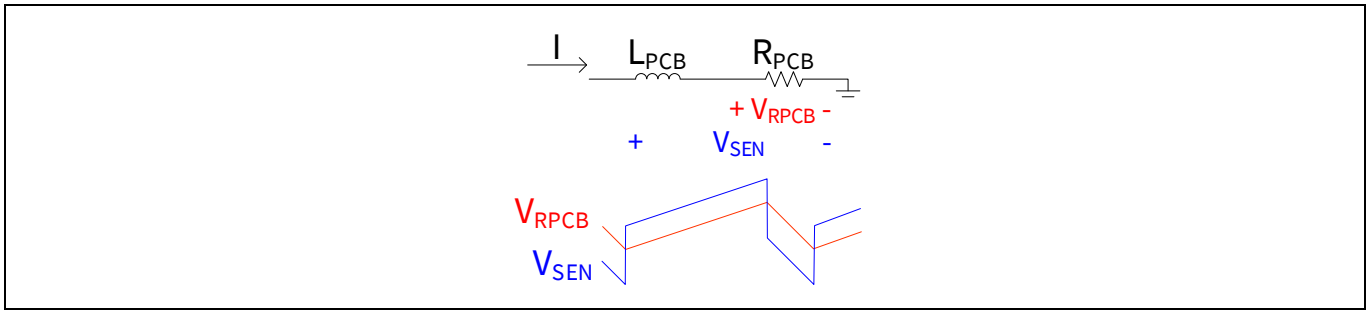

Figure 29 PCB trace inductance

Register **ce_ltrace** is provided to compensate for this step in the input to the AFE DAC. The compensation value is obtained according to Equation (3.10), where the parameter ce_ltrace_real is defined in Equation (3.11).

$$ce_ltrace = INT\left(\frac{ce_ltrace_real}{2H/\Omega sec}\right), \quad (3.10)$$

$$ce_ltrace_real = \frac{L_{PCB}(H)}{(2*R_{PCB}(\Omega)*10e{-}8sec)}, \quad (3.11)$$

3.3 Current sense processor

The main purpose of the ISP is to convert digitally reconstructed inductor current waveform from ADC codes into digital representation of amps. A simplified block diagram of the ISP is shown in Figure 30.

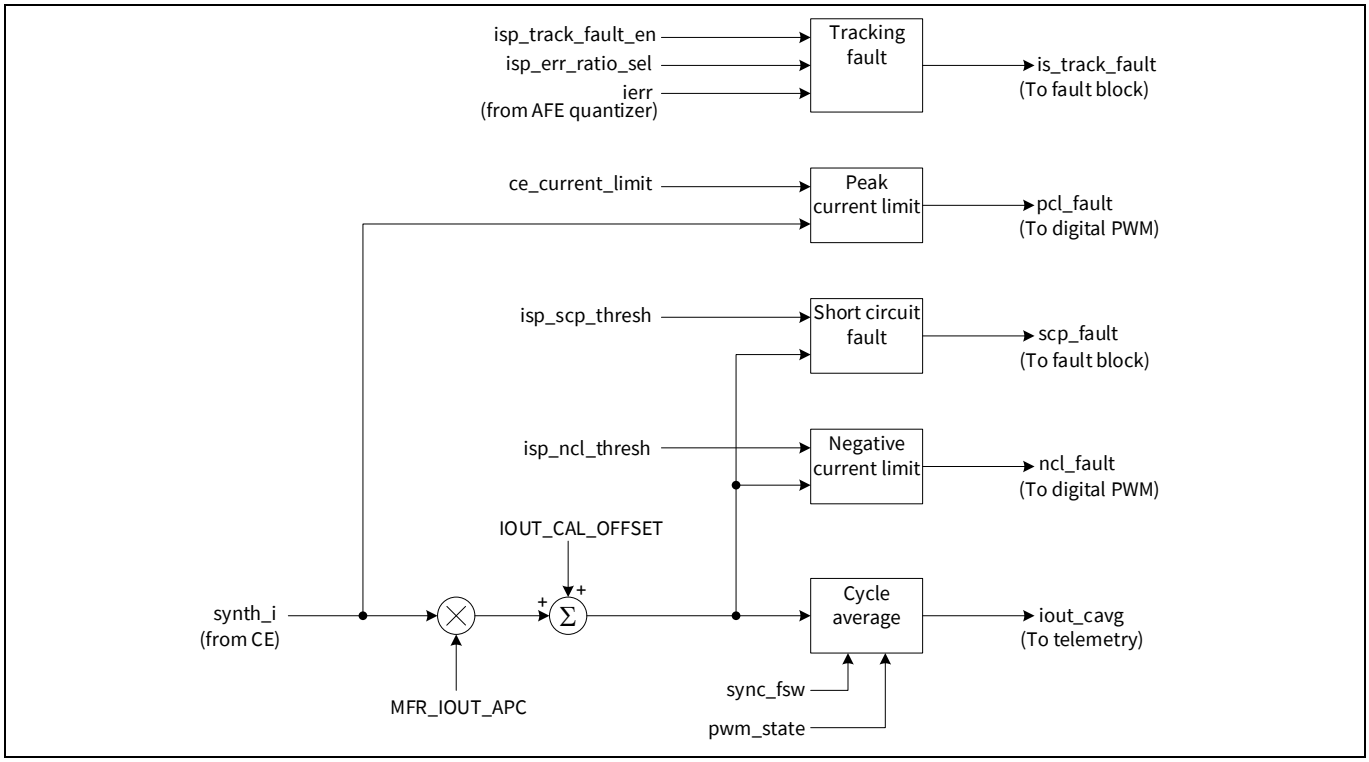

Figure 30 Simplified block diagram of the ISP

The ISP also includes several phase current-based fault checks, as shown in Figure 30. These fault checks include:

- Peak current limit (PCL)
- Negative current limit
- Short-circuit fault
- Error tracking fault

The main functionalities of the ISP are described in the following subsections, including the relevant user-definable parameters.

3.3.1 ADC codes to amps

The process to convert the ADC codes into amps requires three steps and is illustrated in Figure 30. The relevant steps are as follows:

- CE output synth_i (represented in ADC codes) is multiplied by the PMBus command MFR_IOUT_APC (APC refers to “amps per code”, converting the code-based word synth_i into a digital representation of amps)
- The resulting APC multiplication output is then summed with the PMBus command IOUT_CAL_OFFSET to adjust for offset errors
- The reconstructed waveform in amps is averaged over one switching cycle to produce a cycle average current

The selection over which switching cycle the current is averaged is done via register **isp_fsw_sync_sel**. The user can decide whether the switching cycle is averaged over Loop 0 or Loop 1. The averaged current is further processed downstream in the telemetry and fault blocks.

Note that the PMBus Page 0 commands for MFR_IOUT_APC and IOUT_CAL_OFFSET correspond to the ISEN path, and the PMBus Page 1 commands correspond to the BISEN path.

3.3.2 Peak current limiting

PCL is implemented by comparing:

- The reconstructed current waveform from the CE in IADC codes
- PCL threshold, **ceX_current_limit**

Register **ceX_current_limit** sets the cycle-by-cycle PCL threshold, which is defined in IADC codes. It can be computed from amps according to Equation (3.12).

$$ceX_current_limit = \frac{Current\ in\ Amps}{MFR_IOUT_APC}, \tag{3.12}$$

The **ce0_current_limit** sets the limit for ISEN input, and correspondingly the **ce1_current_limit** sets the limit for BISEN input. If the sensed current exceeds the current limit, the duty cycle is truncated by turning off the PWM signal associated with the on-state (indicated by **ceX_on_mask0** and **ceX_on_mask1)** in order to prevent the inductor current from increasing further. The PCL protection can be disabled by setting the threshold to zero.

3.3.3 Negative current limiting

The negative current limiting (NCL) block is illustrated in Figure 30, and it compares:

- Instantaneous phase current (prior to cycle averaging)
- NCL threshold, **ispX_ncl_thresh**

If the phase current drops below the NCL threshold, the FETs associated with the off-state (defined by **ceX_off_mask0** and **ceX_off_mask1**) are turned off.

In isolated topologies the FETs associated with the off-time are the synchronous FETs. In general, turning off the SR FETs with a large negative inductor current is undesirable. Therefore, the recommendation is to disable NCL in isolated topologies. This fault protection and its response can be disabled by setting the register **isp_ncl_thresh** to the maximum positive setting, 7Fh.

3.3.4 Short-circuit protection

Short-circuit protection (SCP) is implemented by comparing:

- Instantaneous phase current (prior to cycle averaging)
- SCP fault threshold, **isp_scp_thresh**

For this SCP only a single sample above the threshold is needed to trip. Therefore, the threshold value should be set the highest among the various overcurrent protection (OCP) thresholds (e.g., IOUT_OC_WARN_LIMIT, IOUT_OC_FAULT_LIMIT, MFR_IOUT_OC_FAST_ FAULT_LIMIT).

In multiphase topologies, the SCP threshold is applied per phase, and the response to a short-circuit is to immediately shut down the output of the loop on which the fault occurred. This is different from the PMBus-based current faults, which operate on the total loop current. This protection can be disabled by setting the threshold value in register **isp_scp_thresh** to zero.

3.3.5 Error tracking fault detection

Error tracking fault detects the inability of the IADC tracking mechanism to track the input. This fault typically indicates a board-level problem, for instance one of the ISEN/IREF inputs is not connected properly. The tracking fault detection is enabled by setting **ispX_track_fault_en** to 1.

The error tracking fault detection monitors the IADC quantizer output, which has six levels, ±1, ±3, and ±5. The IADC's ability to track the input can be estimated by comparing the number of ±5 samples relative to the number of ±1 and ±3 samples.

The current sense tracking fault detection is based on a user-definable error ratio of ±5 to total samples, and it can be set via register **ispX_err_ratio_sel**. Table 15 describes the **ispX_err_ratio_sel** programming including the error threshold, the ratio of ±5 to total samples, as well as the minimum number of samples and time to fault declaration. The minimum is based on a continuous stream of ±5 samples, which causes the tracking fault counter to count up continuously with no intervening down counts.

Table 15 Current sense tracking error threshold

ispX_err_ration_sel	Error threshold	Error ratio (%)	Min. samples to fault	Min. time to fault (µs)
0	4	4/36 = 11.1 percent	124	4.96
1	8	8/40 = 20.0 percent	248	9.92
2	12	12/44 = 27.3 percent	372	14.88
3	16	16/48 = 33.3 percent	496	19.84
4	24	24/56 = 42.9 percent	744	29.76
5	32	32/64 = 50.0 percent	992	39.68
6	80	48/80 = 60.0 percent	1488	59.52
7	96	64/96 = 66.7 percent	1984	79.36

The error tracking fault detection is based on the operation of three counters:

- Reference counter, cnt_ref, counts the number of ±1 and ±3 samples
- Error counter, cnt_err, counts the number of ±5 samples
- Tracking fault counter, cnt_track_fault, counts the tracking fault

If the error counter reaches the **ispX_err_ratio_sel** before the reference counter reaches 32, the tracking fault counter is incremented. Correspondingly, in case the reference counter reaches 32 before the error counter reaches the **ispX_err_ratio_sel**, the tracking fault counter is decremented. Thus, a tracking fault is declared if the tracking fault counter reaches a count of 31.

Tracking fault is cleared only with shutdown via the EN pin or PMBus OPERATION command. However, shutdown due to the tracking fault itself (or any other fault), without subsequent EN pin or OPERATION state change, does not clear the fault. This is because tracking fault typically indicates a board-level problem such as unconnected ISEN or IREF input.

Lower-valued settings of **ispX_err_ratio_sel** are very sensitive and may cause false fault declaration, particularly in noisy current sense environments. Therefore, it is recommended to use only the higher-valued **ispX_err_ratio_sel** settings of 6 or 7.

3.4 Current sense registers

The relevant current sense registers and their descriptions are provided in Table 16.

Table 16 Current sense-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
analog	isen1_gain_mode	RW	7000_0400h	[13:12]	ISEN1 (ISEN) gain mode select. Defines LSB weight of ISEN1 ADC. Also defines expected reference voltage level on IREF1 (IREF). 0: Reserved 1: LSB = 100 µV, IREF level = GND 2: LSB = 1.45 mV, IREF level = GND 3: LSB = 1.45 mV, IREF level = 1.2 V
analog	isen2_gain_mode	RW	7000_0400h	[15:14]	ISEN2 (BISEN) gain mode select. Defines LSB weight of ISEN2 ADC. Also defines expected reference voltage level on IREF2 (BIREF). 0: Reserved 1: LSB = 100 µV, BIREF level = GND 2: LSB = 1.45 mV, BIREF level = GND 3: LSB = 1.45 mV, BIREF level = 1.2 V
isen	ce_ktrack_hiz	RW	7000_2400h (ISEN) 7000_2800h (BISEN)	[3:0]	Current sense tracking gain in the HIZ state. The tracking gain defines the relative strength of the IADC-based correction term applied to the emulated current waveform. The HIZ state generally refers to the state when all primary and secondary switches are off. Note that this register should be set to 0 for primary-side current sense and for the boost topology. LSB = 1/16 V/V, range = 0 to 15/16 V/V
isen	ce_ktrack_off	RW	7000_2400h (ISEN) 7000_2800h (BISEN)	[7:4]	Current sense tracking gain in the off-state. The tracking gain defines the relative strength of the IADC-based correction term applied to the emulated current waveform. The off-state generally refers to the state when the output inductor current is in its downward slope cycle (e.g., primary FETs off, secondary FETs on in a bridge

Peripheral	Field name	Access	Address	Bits	Description
					topology). Note that this register should be set to 0 for primary-side current sense and for the boost topology. LSB = 1/16 V/V, range = 0 to 15/16 V/V
isen	ce_ktrack_on	RW	7000_2400h (ISEN) 7000_2800h (BISEN)	[11:8]	Current sense tracking gain in the on-state. The tracking gain defines the relative strength of the IADC-based correction term applied to the emulated current waveform. The on-state generally refers to the state when the output inductor current is in its upward slope cycle (e.g., primary FETs on in a bridge topology). LSB = 1/16 V/V, range = 0 to 15/16 V/V
isen	ce_kslope_didv	RW	7000_2400h (ISEN) 7000_2800h (BISEN)	[22:12]	Output inductor (L_{OUT}) current sense slope normalized to code/samples at 1.0 V. **For secondary current sense:** ce_kslope_didv = 1.0 V * 10 ns / (L_{OUT}(nH) * APC(A)) **For primary current sense:** ce_kslope_didv = 1.0 V * 10 ns / (N_{turn} * L_{OUT}(nH) * APC(A)) LSB = 2^-13 V/V, range = 0.0 to 0.25 V/V
isen	ce_pwmwin_dly	RW	7000_2400h (ISEN) 7000_2800h (BISEN)	[27:23]	Defines delay used to align internal PWM signals to incoming current sense waveform. Delay = (ce_pwmwin_dly + 1) * 10 ns LSB = 10 ns, range = 10 to 320 ns
isen	ce_ladj_en	RW	7000_2400h (ISEN) 7000_2800h (BISEN)	[30]	Inductor slope correction function enable. When enabled, the controller will attempt to auto-correct for errors in ce_kslope_didv based on the incoming current sense waveform. 0: Disabled 1: Enabled
isen	ce_ps_current_emu	RW	7000_2400h (ISEN) 7000_2800h (BISEN)	[31]	Primary-side, secondary-side current sense select. 0: Secondary-side current sense 1: Primary-side current sense

Peripheral	Field name	Access	Address	Bits	Description
isen	ce_on_mask0	RW	7000_2404h (ISEN) 7000_2804h (BISEN)	[11:0]	Defines the on-state of the CE with respect to the PWM outputs where the on-state generally refers to the state when the output inductor current is in its upward slope cycle (e.g., primary FETs in a bridge topology). In bridge topologies there are two on-states per switching cycle. ce_on_mask0 defines the first (even) on-state and ce_on_mask1 defines the second (odd) on-state. In non-bridge topologies (e.g., ACF, buck) ce_on_mask0 and ce_on_mask1 are typically set to the same value. On-state equation: ON = &(~ce_on_mask0[11:0] | pwm[11:0]) || &(~ce_on_mask1[11:0] | pwm[11:0]) where [0] corresponds to PWM1, and [11] corresponds to PWM12 **Primary topology set: ce_on_mask0 bits corresponding to** ACF Q1 HB Q1 FB Q1, Q3 Buck HSFET
isen	ce_on_mask1	RW	7000_2404h (ISEN) 7000_2804h (BISEN)	[23:12]	Defines the on-state of the CE with respect to the PWM outputs where the on-state generally refers to the state when the output inductor current is in its upward slope cycle (e.g., primary FETs on in a bridge topology). In bridge topologies there are two on-states per switching cycle. ce_on_mask0 defines the first (even) on-state and ce_on_mask1 defines the second (odd) on-state. In non-bridge topologies (e.g., ACF, buck) ce_on_mask0 and ce_on_mask1 are typically set to the same value. On-state equation: ON = &(~ce_on_mask0[11:0] | pwm[11:0]) || &(

Peripheral	Field name	Access	Address	Bits	Description
					~ce_on_mask1[11:0] | pwm[11:0]) where [0] corresponds to PWM1, and [11] corresponds to PWM12 **Primary topology set: ce_on_mask0 bits corresponding to** ACF Q1 HB Q2 FB Q2, Q4 Buck HSFET
isen	ce_dt_l_slope	RW	7000_2404h (ISEN) 7000_2804h (BISEN)	[31:24]	Defines the slope of the output inductance dependence on current. Although actual inductor variation with current is non-linear, this compensation introduces a linear correction to the inductor slope. It is recommended to use the 0 A inductance as one point. The user can choose the second point for the best fit curve. **Computed as follows:** L0 = inductance at 0 A L1 = inductance at user selected current, I1 APC = amps per code defined by MFR_IOUT_APC ce_dt_l_slope = round(2^14 * (L0/L1 - 1) * APC/I1) **Example:** L0 = 470 nH, L1 = 420 nH at 50 A, APC = 0.25 ce_dt_l_slope = round(2^14 * (470/420 - 1) * 0.25/50) = 10 LSB = 2^-14 Vs/HA, range = 0.0 to 0.015564 Vs/HA
isen	ce_off_mask0	RW	7000_2408h (ISEN) 7000_2808h (BISEN)	[11:0]	Defines the off-state of the CE with respect to the PWM outputs where the off-state generally refers to the state when the output inductor current is in its downward slope cycle (e.g., primary FETs off, secondary FETs in a bridge topology). In bridge topologies there are two off-states per switching cycle. ce_off_mask0 defines the first (even) off-state and

Peripheral	Field name	Access	Address	Bits	Description
					ce_on_mask1 defines the second (odd) off-state. In non-bridge topologies (e.g., ACF, buck) ce_off_mask0 and ce_off_mask1 are typically set to the same value. Off-state equation: OFF = &(~ce_off_mask0[11:0] | pwm[11:0]) && &(~ce_off_mask1[11:0] | pwm[11:0]) && !ON where [0] corresponds to PWM1, [11] corresponds to PWM12 and on is defined by ce_on_mask0 and ce_on_mask1 **Secondary topology set: ce_off_mask0 bits corresponding to** ACF SR2 CT SR2 FW SR2, SR4 CDR SR1 Buck LSFET
isen	ce_off_mask1	RW	7000_2408h (ISEN) 7000_2808h (BISEN)	[23:12]	Defines the off-state of the CE with respect to the PWM outputs where the off-state generally refers to the state when the output inductor current is in its downward slope cycle (e.g., primary FETs off, secondary FETs in a bridge topology). In bridge topologies there are two off-states per switching cycle. ce_off_mask0 defines the first (even) off-state and ce_on_mask1 defines the second (odd) off-state. In non-bridge topologies (e.g., ACF, buck) ce_off_mask0 and ce_off_mask1 are typically set to the same value. Off-state equation: OFF = &(~ce_off_mask0[11:0] | pwm[11:0]) && &(~ce_off_mask1[11:0] | pwm[11:0]) && !ON where [0] corresponds to PWM1, [11] corresponds to PWM12 and on is defined by ce_on_mask0 and ce_on_mask1

Peripheral	Field name	Access	Address	Bits	Description
					Secondary topology set: ce_off_mask0 bits corresponding to ACF SR2 CT SR1 FW SR1, SR3 CDR SR2 Buck LSFET
isen	ce_ltrace	RW	7000_2408h (ISEN) 7000_2808h (BISEN)	[28:24]	Defines parasitic trace inductance as seen by current sense input. Computed as follows: ce_ltrace = L_{trace} / (R_{trace} * dt) where L_{trace} = parasitic trace inductance being compensated R_{trace} = current sense trace resistance dt = 10 ns Example: L_{trace} = 300 pH, R_{trace} = 0.5 mΩ ce_ltrace = 300 pH/(1.0mΩ * 10 ns) = 30 LSB = 2 H/Ω-s, range = 0 to 62 H/Ω-s
isen	isp_scp_thresh	RW	7000_240Ch (ISEN) 7000_280Ch (BISEN)	[7:0]	SCP fault threshold. The SCP threshold should be set highest among the various current protection thresholds, as it requires only a single sample above the threshold to trip. The SCP threshold is applied per phase on multiphase topologies. Set this threshold to 0 to disable the fault detection. LSB = 1 A, range = 0 to 255 A
isen	isp_ncl_thresh	RW	7000_240Ch (ISEN) 7000_280Ch (BISEN)	[15:8]	Inductor NCL fault threshold. The NCL threshold is compared against the instantaneous phase current. When the phase current drops below the NCL threshold, the response is to turn off the FETs associated with the off-state defined in ce_off_mask0,1. In isolated topologies this corresponds to the SR FETs. In general, turning off the SR FETs with a large negative inductor current does not produce a

Peripheral	Field name	Access	Address	Bits	Description
					desirable result. It is therefore recommended to disable this feature in isolated topologies. Set to the maximum positive setting, 127, to disable this fault protection and its response. LSB = 0.25 A, range = -32 to +31.75 A
isen	ce_blank_ne_dly	RW	7000_240Ch (ISEN) 7000_280Ch (BISEN)	[18:16]	Defines the time after a PWM pulse falling edge during which the current sense is in emulation mode only (i.e., the tracking feedback from the IADC is ignored). This parameter can be used to blank the ADC feedback if the falling PWM edge causes noise on the current sense signal. LSB = 40 ns, range = 0 to 280 ns
isen	ce_blank_pe_dly	RW	7000_240Ch (ISEN) 7000_280Ch (BISEN)	[21:19]	Defines the time after a PWM pulse rising edge during which the current sense is in emulation mode only (i.e., the tracking feedback from the IADC is ignored). This parameter can be used to blank the ADC feedback if the rising PWM edge causes noise on the current sense signal. LSB = 40 ns, range = 0 to 280 ns
isen	ce_kslope_lm	RW	7000_2410h (ISEN) 7000_2810h (BISEN)	[8:0]	Transformer magnetizing inductance (L_m) current sense slope normalized to code/samples at 1.0 V. Only used for primary-side current sense. ce_kslope_lm = 1.0 V * 10 ns * N_{turn} / (L_m(nH) * APC(A)) LSB = 2^-13 V/V, range = 0.0 to 0.06238 V/V
isen	ce_iterm	RW	7000_2410h (ISEN) 7000_2810h (BISEN)	[12:9]	Current sense tracking error integrator coefficient. Set to 0 to disable error accumulation. LSB = 2^-13 A/A, range = 0.0 to 1.833e-3 A/A
isen	ce_vout_sel	RW	7000_2410h (ISEN) 7000_2810h (BISEN)	[14:13]	Current sense V_{OUT} source select. Note that CDR in the table below indicates current-doubler topology on the secondary. **CE0 (ISEN):** 0: Loop 0 V_{OUT}

Peripheral	Field name	Access	Address	Bits	Description
					1: Loop 0 V_{OUT} CDR 2: Loop 0 V_{OUT} 3: Loop 0 V_{OUT} CDR **CE1 (BISEN):** 0: Loop 0 V_{OUT} 1: Loop 0 V_{OUT} CDR 2: Loop 1 V_{OUT} 3: Loop 1 V_{OUT} CDR
isen	ce_vrect_sel	RW	7000_2410h (ISEN) 7000_2810h (BISEN)	[17:15]	Current sense V_{RECT} (V_{IN}) source select. **CE0 (ISEN):** 0: V_{RECT} (VRSEN/VRREF) 1: BV_{RECT} (BVRSEN/BVRREF) 2: pid_ff_vrect_override 3: PRISEN 4: V_{RECT} (VRSEN/VRREF) 5: BV_{RECT} (BVRSEN/BVRREF) 6: pid_ff_vrect_override 7: PRISEN **CE1 (BISEN):** 0: V_{RECT} (VRSEN/VRREF) 1: BV_{RECT} (BVRSEN/BVRREF) 2: pid_ff_vrect_override 3: PRISEN 4: Loop 0 V_{OUT} 5: Loop 0 V_{OUT} 6: Loop 0 V_{OUT} 7: Loop 0 V_{OUT}
isen	ce_current_limit	RW	7000_2410h (ISEN) 7000_2810h (BISEN)	[26:18]	Inductor PCL fault threshold in IADC codes. The PWM pulse will be truncated beyond this limit. Set to 0 to disable PCL fault protection. ce_current_limit = current_limit(amps) / (MFR_IOUT_APC (amps/code)) LSB = 1 code, range = 0 to 511 codes
isen	ce_topology	RW	7000_2410h (ISEN) 7000_2810h (BISEN)	[28:27]	Current emulator topology select. Defines the inductor voltage equations for V_{ON}, V_{OFF}. 0: Buck, ACF, HB, FB 1: Boost 2, 3: Buck-boost
isen	isp_fsw_sync_sel	RW	7000_2410h (ISEN) 7000_2810h (BISEN)	[29]	Defines which loop's frequency is used for averaging current over switching cycle. 0: Loop 0 F_{switch} 1: Loop 1 F_{switch}

Peripheral	Field name	Access	Address	Bits	Description
					If ISEN is assigned to Loop 0 and BISEN is assigned to Loop 1, then set isen0.isp_fsw_sync_sel=0 and isen1.isp_fsw_sync_sel=1. If both ISEN and BISEN are assigned to Loop 0, then set both parameters to 0.
isen	isp_track_fault_en	RW	7000_2414h (ISEN) 7000_2814h (BISEN)	[0]	Current sense tracking fault enable. The tracking fault detects the inability of the current sense emulator to track the incoming current sense signal. This fault is typically an indication of board problem (e.g., missing sense resistor, bad pin connection). 0: Disabled 1: Enabled
isen	isp_err_ratio_sel	RW	7000_2414h (ISEN) 7000_2814h (BISEN)	[3:1]	Current sense tracking fault error ratio select. 0: 4 (11.1 percent threshold) 1: 8 (20.0 percent threshold) 2: 12 (27.3 percent threshold) 3: 16 (33.3 percent threshold) 4: 24 (42.9 percent threshold) 5: 32 (50.0 percent threshold) 6: 48 (60.0 percent threshold) 7: 64 (66.7 percent threshold)
isen	isp_apc	RW	7000_2418h (ISEN) 7000_2818h (BISEN)	[10:0]	IADC APC, maps the current sense ADC LSB into amps. **Computed by FW from PMBus command as follows:** isp_apc = MFR_IOUT_APC LSB = 1.9531 mA, range = 0.0 to 3.998 A
isen	isp_ioffset	RW	7000_2418h (ISEN) 7000_2818h (BISEN)	[7:0]	Phase current offset **Computed by FW from PMBus command as follows:** isp_offset = IOUT_CAL_OFFSET LSB = 0.125 A, range = -16 to +15.875 A

3.5 Current sense PMBus commands

The relevant PMBus-related commands for current sensing are given in Table 17.

Table 17 Current sense-related PMBus commands

Command name	Access	Length	Address	Bits	Description
IOUT_CAL_OFFSET	RW	Word	39h	[15:0]	The IOUT_CAL_OFFSET is used to null out any offsets in the output current sensing circuit. This command is most often used in conjunction with the IOUT_CAL_GAIN command (above) to minimize the error of the current sensing circuit. Linear11 format with suggested exponent -3. Unit = amps
FW_CONFIG_REGULATION	RW	Block 14 bytes	C5h	[1]	EN_PRIM_ISENSE: 0 = Only ISEN enabled in single-loop design 1 = Both ISEN and BISEN enabled in single-loop design
MFR_IOUT_APC	RW	Word	EAh	[15:0]	Current sense APC. Linear11 format with suggested exponent -8 or -9 depending on magnitude. Unit = amps

4 Telemetry sense

This chapter introduces the telemetry sense (TS) functionality and the relevant registers. The TS module performs analog-to-digital conversion for the following modules:

- Temperature telemetry
- Input voltage telemetry
- Current sharing
- X-valent address programming
- General-purpose ADC

VS and current sense modules cover the ADCs for the relevant voltage and current signals, as described in Section 2 and Section 3, respectively. The TS module consists of the following submodules:

- Telemetry sense ADC (TSADC)
- Input mux
- TS current DAC
- TS processor

These submodules and their connections are shown in Figure 31, and the following subsections introduce these submodules in more detail.

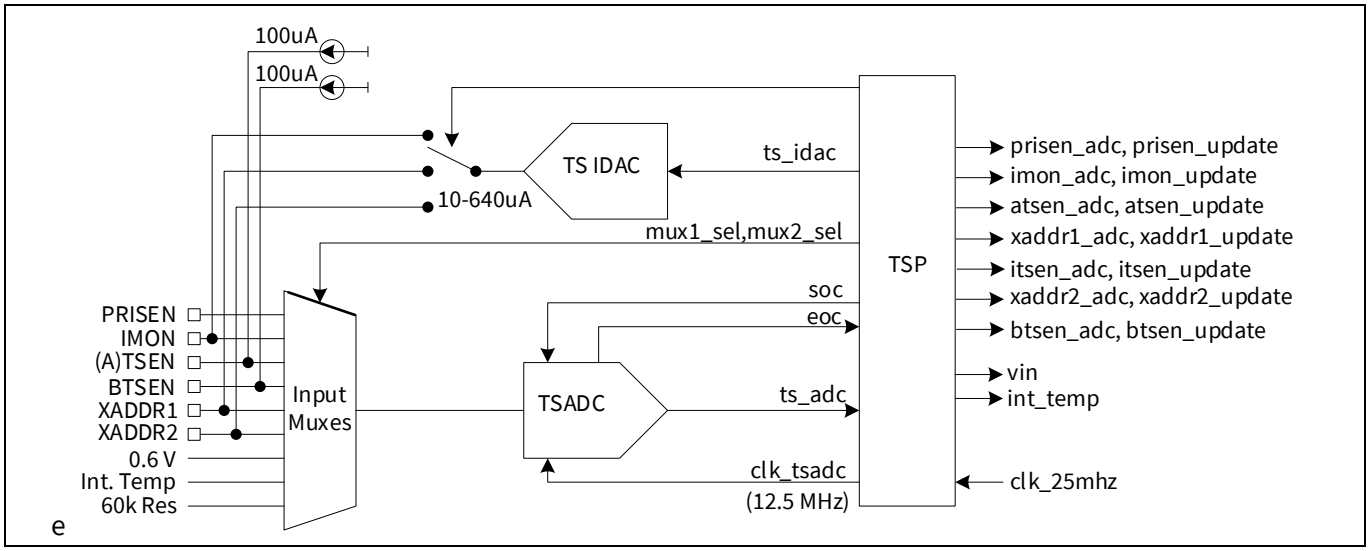

Figure 31 TS block diagram

4.1 Telemetry sense ADC

The TSADC is a successive approximation ADC, and it has the following specifications:

- 9 bits
- 1 Msps conversion rate
- Input range 0.0 V to 1.2 V
- LSB = 2.34375 mV

4.2 TS input mux and sequencing

The submodule input mux consists of six inputs, which are shown in Figure 32. The input signals are assigned to the following XDPP1100 input pins: PRISEN, IMON, TSEN, BTSEN, XADDR1, and XADDR2. If a pin is not used for the associated named function, the input is available to be used as a general-purpose ADC input.

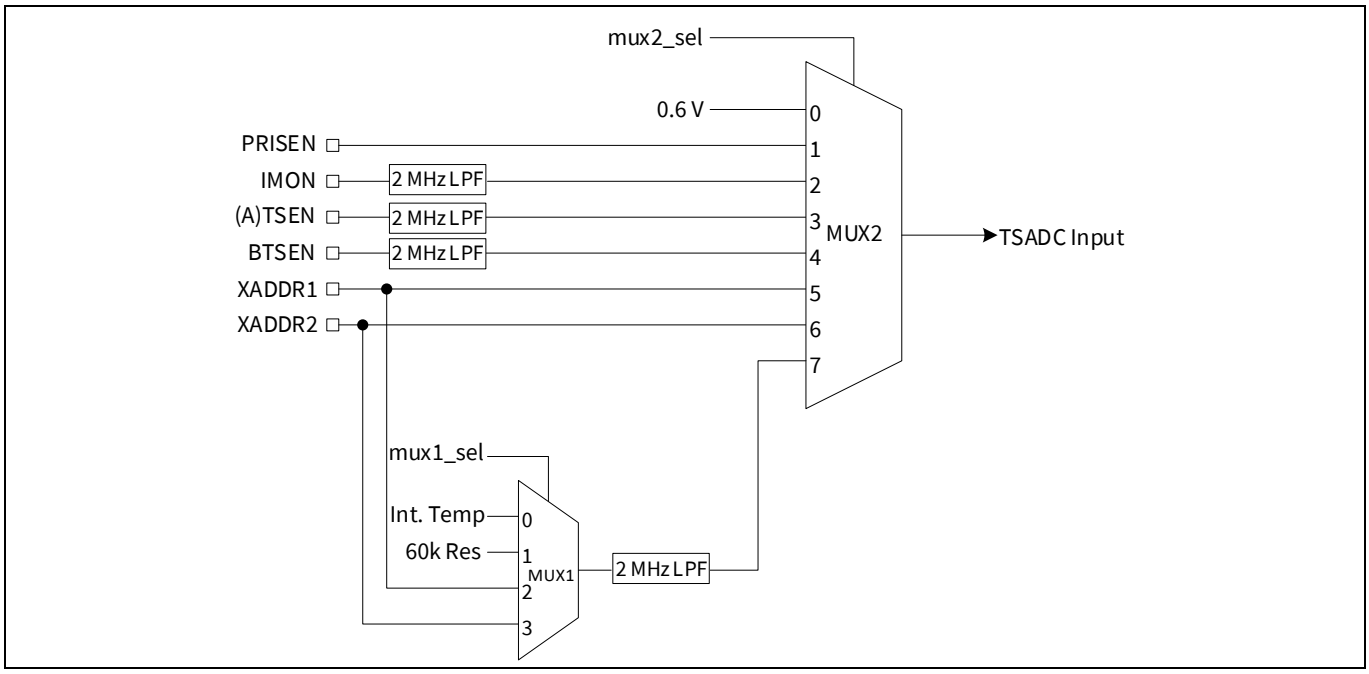

Figure 32 TS input muxes

The TS module contains eight timing slots with sequencing defined by the register **ts_muxmode** as provided in Table 18. Each slot represents a single conversion at 1 µs conversion rate, and 8 µs are required to cycle through the entire sequence.

Table 18 Input sequencing

	TSADC MUX2 output sequence							
ts_muxmode	**Slot 1**	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**
0-3	Fixed, defined by ts_muxctrl1, 2							
4	PRISEN	IMON	PRISEN	ATSEN	PRISEN	IMON	PRISEN	BTSEN
5	PRISEN	XADDR1	PRISEN	ATSEN	PRISEN	MUX1	PRISEN	BTSEN
6	PRISEN	XADDR1	IMON	ATSEN	PRISEN	MUX1	IMON	BTSEN
7	PRISEN	XADDR1	IMON	ATSEN	PRISEN	MUX1	XADDR2	BTSEN

The **ts_muxmode** settings from 4 to 7 provide four auto-sequence modes as shown in Table 18.Correspondingly, when **ts_muxmode** is less than 4, the input mux selection is defined by registers **ts_muxctrl1** and **ts_muxctrl2**, as shown in Table 19 and Table 20.

Table 19 MUX1 selection

ts_muxctrl1	MUX1 output
0	Internal temperature (ITSEN)
1	60 K resistor (test only)
2	XADDR1
3	XADDR2

Table 20 MUX2 selection

ts_muxctrl2	MUX2 output
0	0.6 V reference (test only)
1	PRISEN
2	IMON
3	ATSEN
4	BTSEN
5	XADDR1
6	XADDR2
7	MUX1 output

In addition to **ts_muxmode**, **ts_muxctrl1**, and **ts_muxctrl2**, the following registers are required to enable an input for ADC conversion:

- PRISEN
  - **prisen_meas_en** enables the PRISEN input for ADC
- IMON
  - **imon_meas_en** enables the IMON input for ADC
  - **ts_tsidac_imon_sel** enables the IMON current DAC output path when used for current sharing or as a general-purpose DAC
- (A)TSEN
  - **atsen_meas_en** enables the TSEN input for ADC
  - **ts_tsidac_antc_sel** enables the TSEN 100 µA current source when used with an NTC or PTC sense element
- BTSEN
  - **btsen_meas_en** enables the BTSEN input for ADC
  - **ts_tsidac_bntc_sel** enables the BTSEN 100 µA current source when used with an NTC or PTC sense element
- ITSEN
  - **itsen_meas_en** enables the internal temperature sense input for ADC
- XADDR1, 2
  - Selection of a sequence via **ts_muxmode** containing XADDR1 or XADDR2 is sufficient to enable these input paths when used as a general-purpose ADC
  - See Section 4.4.5 for additional parameters related to HW-defined XADDR1, two-pin measurement

Figure 33 shows a portion of the conversion sequence with **ts_muxmode** = 7 and **ts_muxctrl1** = 0. The PRISEN conversion is highlighted along with the subsequent V_{IN} computation.

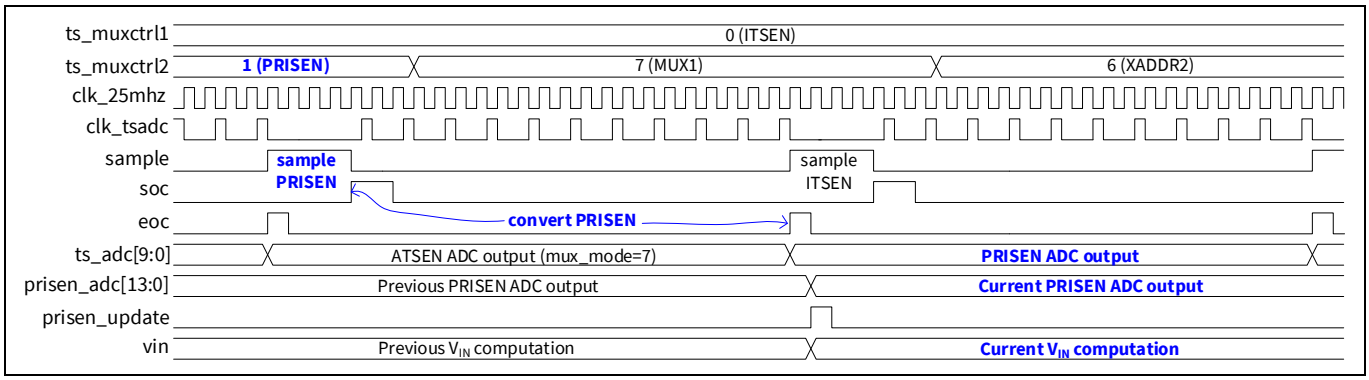

Figure 33 TS input sequencing

4.3 Telemetry sense current DAC

The telemetry sense current DAC (TS IDAC) is connected to either the XADDR1, XADDR2 or IMON pin to develop a voltage across resistors connected between pin and GND, as was shown in Figure 31. The selection of the pin connection is controlled by the TS processor (TSP).

The TS IDAC has the following specifications:

- 6-bit current output DAC
- LSB = 10 µA
- Range = 0 to 630 µA

When it is used for current sharing, additional resolution (up to 10 bits total) is created through dithering. It also requires a capacitor in parallel to the resistor from IMON to GND.

4.4 Telemetry sense processor

The TSP is shown in Figure 34, and it consists of the following submodules:

- TS sequencer
- Gain/offset correction
- V_{IN} computation
- Internal temperature computation
- X-valent measurement

These TSP submodules are described in detail in the following subsections.

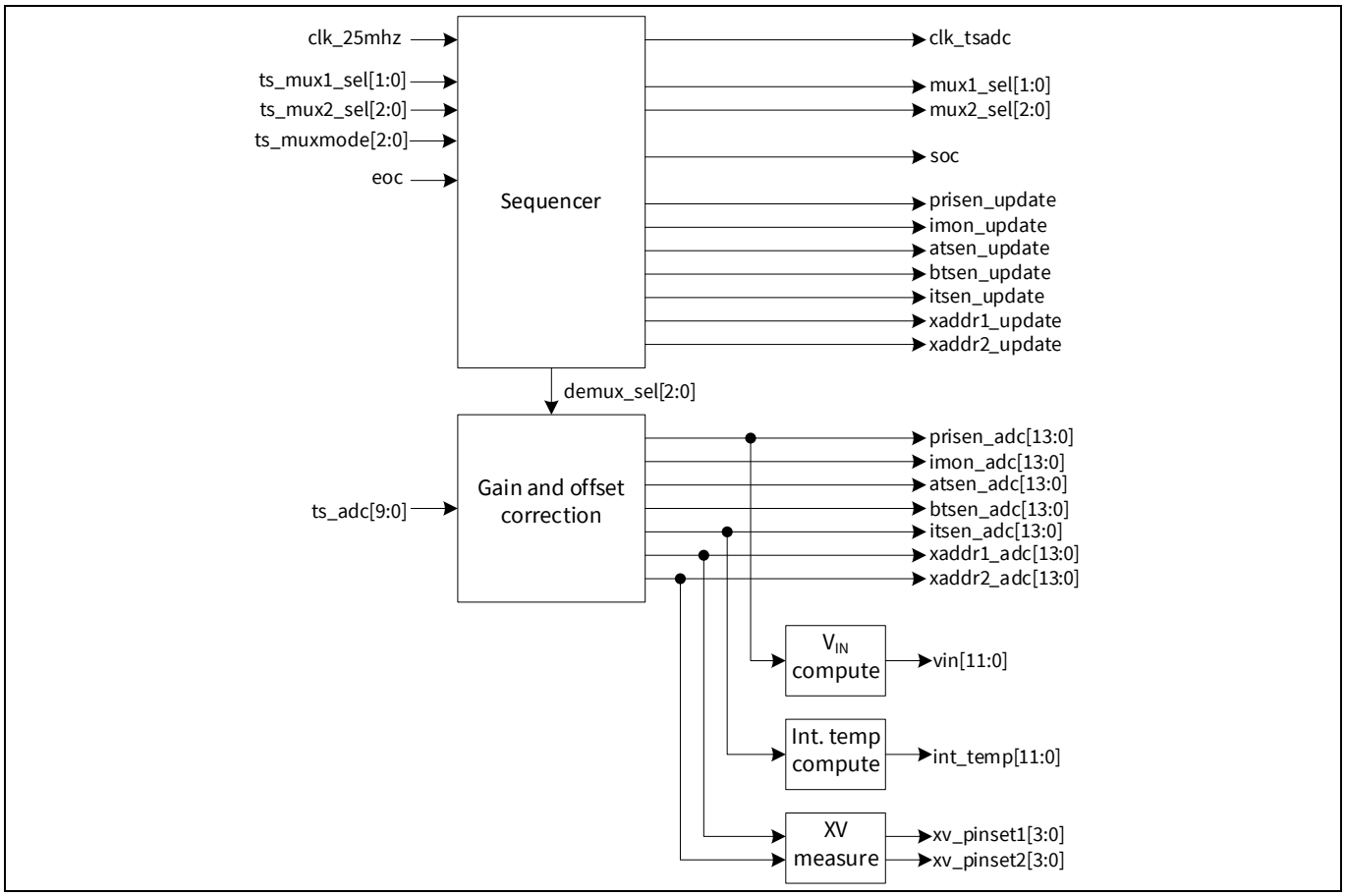

Figure 34 TSP block diagram

4.4.1 Sequencer

The sequencer submodule controls the input mux selection and sequencing, as discussed in Section 4.2. It sends the start of conversion (SOC) signal to the TSADC and receives the end of conversion (EOC) signal from the TSADC. In addition, it controls the demuxing of the TSADC output into individual channels at the gain/offset output and generates update signals aligned to the demuxed ADC outputs.

4.4.2 Gain and offset correction

The gain/offset submodule performs gain and offset trim for each input channel. It also demuxes the TSADC output into individual channel ADC outputs. These outputs are then aligned to the associated update signal generated in the sequencer, and they are accessible through the following read-only registers:

- **ts_prisen_adc**
- **ts_imon_adc**
- **ts_atsen_adc**
- **ts_btsen_adc**
- **ts_itsen_adc**
- **xv1_adc** and **xv2_adc**

The individual channel gain and offset trim parameters are measured and set during IC testing, and are not user-accessible.

4.4.3 V_{IN} computation

When V_{IN} is measured using the PRISEN input, the TSP can compute V_{IN} from the ADC codes. This is performed based on the following registers and according to Equation (4.1):

- **vin_pwl_slope**
- **vin_trim**

$$V_{IN} = ADC * vin_pwl_slope + vin_trim \quad (4.1)$$

The computation process is also illustrated in Figure 35. It should be noted that **vin_trim** is an offset adjust and not a true trim parameter. The computed V_{IN} is selected as the V_{IN} telemetry input source using register **tlm_vin_src_sel**. Alternatively, the prisen_adc data is available directly through register **ts_prisen_adc** or filtered through register **tlm_prisen_adc_lpf** (see Section 8.10). Both can be read by FW for a user-specific computation of V_{IN} from the PRISEN input.

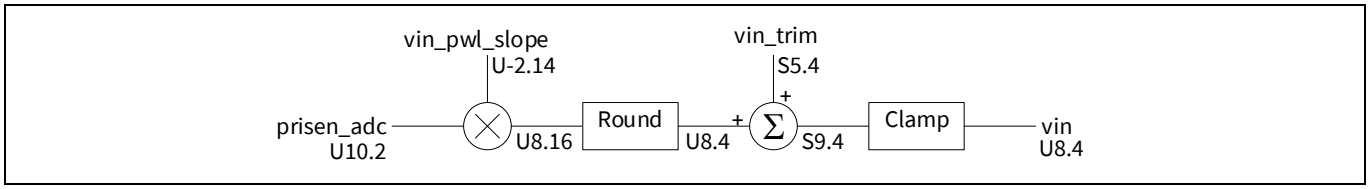

Figure 35 V_{IN} computation

4.4.4 Internal temperature (ITSEN) computation

The XDPP1100 internal temperature is computed from ADC codes based on the following registers and according to Equation 4.2:

- **ptat_0c_code**
- **ptat_pwl_slope**
- **ptat_temp_trim**

$$Int.Temp = (ADC - ptat_0C_code)^{*}\ ptat_pwl_slope + ptat_temp_trim \quad (4.2)$$

The computation process is also illustrated in Figure 36. Note that **ptat_temp_trim** is an offset adjust and not a true trim parameter.

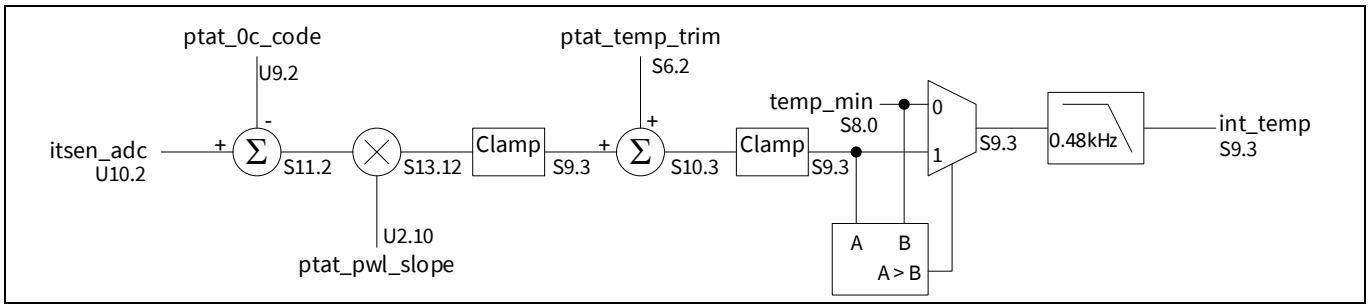

Figure 36 Internal temperature computation and filtering

The parameters to compute the internal temperature are not intended to be user programmable. They are set either during IC testing or through the GUI to the following fixed settings:

- **ptat_0c_code** = 926
- **ptat_pw_slope** = 990
- **ptat_temp_trim** = 0

A register, **temp_min**, is provided to clamp the minimum computed internal temperature. The default setting for this parameter is –128°C. After the clamp, the internal temperature is low-pass filtered at a fixed BW of 0.48 kHz. This filtered version of the internal temperature may be read directly via read-only register **int_temp**. Alternatively, the itsen_adc data is available directly through the following registers:

- **ts_itsen_adc**
- **tlm_itsen_fs**
- Filtered through parameter **tlm_itsen_lpf** (see Section 8.10)

Either may be read by FW for a user-specific computation of internal temperature.

4.4.5 X-valent measurement

The last submodule of the TSP is X-valent (XV) measurement, providing a HW-controlled method to measure and decode the resistor-programmed pinset on the XADDR1, two pins. The XV measurement is enabled through the registers **xv_pinset1_meas** for XADDR1 and **xv_pinset2_meas** for XADDR2. When enabled, the XV measurement occurs during the boot under the following conditions:

- Subsequent to the trim parameter OTP download
- Prior to the configuration parameter and PMBus OTP download

Two fixed LUTs are provided and selected through register **xv_decode_sel** according to the following definitions:

- **xv_decode_sel** = 0 selects the eight-value table shown in Table 21
- **xv_decode_sel** = 1 selects the 16-value table shown in 0

Table 21 8-value XADDR decode

XADDR resistance (Ω)	Eight-value XADDR decode
1200	7
1800	6
2700	5
3900	4
6800	3
10000	2
18000	1
47000	0

Table 22 16-value XADDR decode

XADDR resistance (Ω)	16-value XADDR decode
780	15
1100	14
1500	13
2020	12
2700	11
3520	10
4700	9
6070	8
8000	7
10200	6
13200	5
17200	4
22470	3
29200	2
39000	1
56000	0

The decoded results are available on read-only registers **tlm_xaddr1_pinset** and **tlm_xaddr2_pinset**. Alternatively, a user-defined FW patch may be used to measure XADDR1, two-resistor programming. The following settings should be used:

- HW measurement is disabled by setting registers **xv_pinset1_meas** = 0 and **xv_pinset2_meas** = 0
- Registers **ts_muxctrl1** and **ts_muxctrl2** can be used to select XADDR1, 2 as the fixed input to the TSADC when ts_muxmode is less than 4
- Certain values can be forced on the IDAC through registers **idac_fw_en** and **idac_fw_frc**
- Registers **ts_tsidac_xv1_sel** and **ts_tsidac_xv2_sel** enable IDAC output on XADDR1 and XADDR2, respectively

The unfiltered ADC output is available on read-only registers **xv1_adc** and **xv2_adc**, whereas the filtered ADC output is available on read-only parameters tlm_xaddr1_adc_lpf and tlm_xaddr2_adc_lpf.

4.5 Telemetry sense registers

The relevant TS registers and their descriptions are provided in Table 23.

Table 23 TS-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
tsen	prisen_meas_en	RW	7000_4C00h	[0]	TSADC PRISEN input measurement enable. When enabled, the TSADC will measure the PRISEN input when selected by ts_muxmode and tx_muxctrl2. When disabled, no PRISEN measurement will occur, even if selected by ts_muxmode and ts_muxctrl2.

Peripheral	Field name	Access	Address	Bits	Description
					0: Disabled 1: Enabled
tsen	imon_meas_en	RW	7000_4C00h	[1]	TSADC IMON input measurement enable. When enabled, the TSADC will measure the IMON input when selected by ts_muxmode and tx_muxctrl2. When disabled, no IMON measurement will occur, even if selected by ts_muxmode and ts_muxctrl2. 0: Disabled 1: Enabled
tsen	atsen_meas_en	RW	7000_4C00h	[2]	TSADC ATSEN input measurement enable. When enabled, the TSADC will measure the ATSEN input when selected by ts_muxmode and tx_muxctrl2. When disabled, no ATSEN measurement will occur, even if selected by ts_muxmode and ts_muxctrl2. 0: Disabled 1: Enabled
tsen	btsen_meas_en	RW	7000_4C00h	[3]	TSADC BTSEN input measurement enable. When enabled, the TSADC will measure the BTSEN input when selected by ts_muxmode and tx_muxctrl2. When disabled, no BTSEN measurement will occur, even if selected by ts_muxmode and ts_muxctrl2. 0: Disabled 1: Enabled
tsen	itsen_meas_en	RW	7000_4C00h	[4]	TSADC ITSEN measurement enable. When enabled, the TSADC will measure ITSEN when selected by ts_muxmode and tx_muxctrl2. When disabled, no ITSEN measurement will occur, even if selected by ts_muxmode, ts_muxctrl1 and ts_muxctrl2. 0: Disabled 1: Enabled
tsen	ts_tsidac_antc_sel	RW	7000_4C00h	[6]	ATSEN output current source enable. This current source should be enabled when using the ATSEN input to measure an NTC or PTC temperature sense element. The current source may be disabled to use the ATSEN input as a general-purpose ADC input. 0: Disabled 1: Enabled

Peripheral	Field name	Access	Address	Bits	Description
tsen	ts_tsidac_bntc_sel	RW	7000_4C00h	[7]	BTSEN output current source enable. This current source should be enabled when using the BTSEN input to measure an NTC or PTC temperature sense element. The current source may be disabled to use the BTSEN input as a general-purpose ADC input. 0: Disabled 1: Enabled
tsen	ts_tsidac_imon_sel	RW	7000_4C00h	[8]	IMON output current DAC enable. This current DAC should be enabled when using the IMON pin for current sharing. The current DAC should be disabled otherwise. 0: Disabled 1: Enabled
tsen	ts_tsidac_xv1_sel	RW	7000_4C00h	[10]	XADDR1 output current DAC enable. This current DAC is enabled by FW during the XADDR1 resistor pinset measurement. The current DAC should be disabled otherwise. 0: Disabled 1: Enabled
tsen	ts_tsidac_xv2_sel	RW	7000_4C00h	[11]	XADDR2 output current DAC enable. This current DAC is enabled by FW during the XADDR2 resistor pinset measurement. The current DAC should be disabled otherwise. 0: Disabled 1: Enabled
tsen	xv_decode_sel	RW	7000_4C00h	[12]	XV address table select. Defines the size of the lookup table used to decode the XADDR1 and XADDR2 resistor pinset values. 0: Eight-value table 1: 16-value table
tsen	xv_pinset1_meas	RW	7000_4C00h	[13]	XADDR1 pinset measure control. Enable when using the XADDR1 with an eight- or 16-value resistor set lookup table. 0: XADDR1 measurement disabled 1: XADDR1 measurement enabled
tsen	xv_pinset2_meas	RW	7000_4C00h	[14]	XADDR2 pinset measure control. Enable when using the XADDR2 with an eight- or 16-value resistor set lookup table. 0: XADDR2 measurement disabled 1: XADDR2 measurement enabled

Peripheral	Field name	Access	Address	Bits	Description
tsen	ts_muxctrl1	RW	7000_4C00h	[16:15]	TSADC MUX1 input source select. The output of MUX1 is connected to MUX2 input 7. The most common setting of this register is 0 to measure the internal temperature of the controller. 0: ITSEN 1: 60 K resistor (test only) 2: XADDR1 filtered 3: XADDR2 filtered
tsen	ts_muxctrl2	RW	7000_4C00h	[19:17]	TSADC MUX2 input source select. The output of MUX2 is connected to the TSADC input. Generally, this mux is auto-sequenced by the TSADC (see ts_muxmode below) but may be overridden by setting parameter ts_tsmuxmode=0. 0: 0.6 V reference (test only) 1: PRISEN 2: IMON 3: ATSEN 4: BTSEN 5: XADDR1 unfiltered 6: XADDR2 unfiltered 7: MUX1 (see ts_tsmuxctrl1)
tsen	ts_muxmode	RW	7000_4C00h	[22:20]	TSADC input sequence control. When bit [2] is 0, the TSADC input is entirely determined by the settings of ts_muxctrl1 and ts_muxctrl2. When bit [2] is 1, MUX2 auto-sequences its input using the pattern in the table below. If the sequence includes MUX2 input 7 (MUX1), the source in this timeslot is determined by the setting of ts_muxctrl1. 0 to 3: Defined by ts_muxctrl1, 2 4: Auto-sequence: 1, 2, 1, 3, 1, 2, 1, 4 5: Auto-sequence: 1, 5, 1, 3, 1, 7, 1, 4 6: Auto-sequence: 1, 5, 2, 3, 1, 7, 2, 4 7: Auto-sequence: 1, 5, 2, 3, 1, 7, 6, 4
tsen	ptat_temp_trim	RW	7000_4C04h	[15:8]	ITSEN offset term. Internal temperature computed as: Int. temp. = (ADC - 0C_code) * pwl_slope + temp_trim LSB = 0.25°C, range = -32 to 31.7°C
tsen	ptat_pwl_slope	RW	7000_4C04h	[27:16]	ITSEN piecewise linear slope term. Internal temperature computed as: Int. temp. = (ADC - 0C_code) * pwl_slope + temp_trim

Peripheral	Field name	Access	Address	Bits	Description
					LSB = 2^-10°C/code, range = 0 to 3.9990°C/code
tsen	ptat_0c_code	RW	7000_4C08h	[10:0]	ITSEN 0°C code. Internal temperature computed as: Int. temp. = (ADC - 0C_code) * pwl_slope + temp_trim LSB = 0.25 ADC codes, range = 0.0 to 511.75 ADC codes
tsen	temp_min	RW	7000_4C08h	[18:11]	Defines the lower limit for the internal temperature computation. Computed results less than this value are clamped to this value. LSB = 1°C, range = -128 to +127°C
tsen	xv_gain_scale	RW	7000_4C08h	[26:19]	XADDR1, 2 computation gain scale. The TSADC result is scaled by (1.0 + xv_gain_scale) during the XADDR computations. LSB = 1/256 V/V, range = -0.25 to +0.24805 V/V
tsen	vin_pwl_slope	RW	7000_4C0Ch	[11:0]	TSADC PRISEN input voltage (V_{IN}) piecewise linear slope term. V_{IN} computed as: V_{IN} = ADC * vin_pwl_slope + vin_trim LSB = 2^-14 V/code, range = 0 to 0.24994 V/code
tsen	vin_trim	RW	7000_4C0Ch	[21:12]	TSADC PRISEN input voltage (V_{IN}) offset term. V_{IN} computed as: V_{IN} = ADC * vin_pwl_slope + vin_trim LSB = 62.5 mV, range = -32 to +31.9375 V
tsen	ts_xv_complete	R	7000_4C10h	[0]	TSADC XADDR1, 2 resistor pinset measurement complete status. Used by FW to detect completion of XV measurement. 0: Measurement not started or in progress 1: Measurement complete
tsen	ts_prisen_adc	R	7000_4C14h	[13:0]	Gain and offset corrected PRISEN TSADC output. LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tsen	ts_imon_adc	R	7000_4C18h	[13:0]	Gain and offset corrected IMON TSADC output. LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tsen	ts_atsen_adc	R	7000_4C1Ch	[13:0]	Gain and offset corrected ATSEN TSADC output.

Peripheral	Field name	Access	Address	Bits	Description
					LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tsen	ts_btsen_adc	R	7000_4C20h	[13:0]	Gain and offset corrected BTSEN TSADC output. LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tsen	ts_itsen_adc	R	7000_4C24h	[13:0]	Gain and offset corrected ITSEN TSADC output. LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tsen	xv1_adc	R	7000_4C2Ch	[13:0]	Gain and offset corrected XADDR1 TSADC output. LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tsen	xv2_adc	R	7000_4C30h	[13:0]	Gain and offset corrected XADDR2 TSADC output. LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tsen	int_temp	R	7000_4C34h	[11:0]	Low-pass filtered ITSEN measurement. Filter BW fixed at approximately 950 Hz when the ITSEN measurement is enabled via ts_muxmode and ts_muxctrl1. LSB = 0.125°C, range = -256 to +255.875°C
tlmcom	tlm_xaddr1_pinset	R	7000_5070h	[3:0]	XADDR1 pinset measurement decoded value.
tlmcom	tlm_xaddr2_pinset	R	7000_5070h	[7:4]	XADDR2 pinset measurement decoded value.
tlmcom	tlm_xaddr1_adc_lpf	R	7000_5074h	[13:0]	Low-pass filtered XADDR1 telemetry output when used as a general-purpose ADC. LSB = 0.0625 ADC code, range = 0 to 1023.9375 ADC codes
tlmcom	tlm_xaddr2_adc_lpf	R	7000_5078h	[13:0]	Low-pass filtered XADDR2 telemetry output when used as a general-purpose ADC. LSB = 0.0625 ADC code, range = 0 to 1023.9375 ADC codes
common	idac_fw_frc	RW	7000_3088h	[9:0]	When idac_fw_en is high, this register overrides the HW current DAC output with a FW-controlled setting. LSB = 0.625 µA, range = 0.0 to 639.375 µA
common	idac_fw_en	RW	7000_3088h	[10]	Enables FW-controlled current DAC output via idac_fw_frc. 0: Use HW-determined current DAC output 1: Use idac_fw_frc

5 Voltage control

This chapter introduces the voltage control module and its submodules, as well as describing the relevant registers and PMBus commands.

The purpose of the voltage control module is to set the control or reference voltage with respect to which the output voltage is regulated. The voltage control module consists of the following submodules:

- Ramp generator
- Interrupt generator (IRQ)
- Multi-segment droop, also referred to as active voltage positioning (AVP)
- Low-pass filters
- Output summation and clamping

The voltage control module diagram is shown in Figure 37. The highlighting in the figure indicates read/write and read-only access parameters.

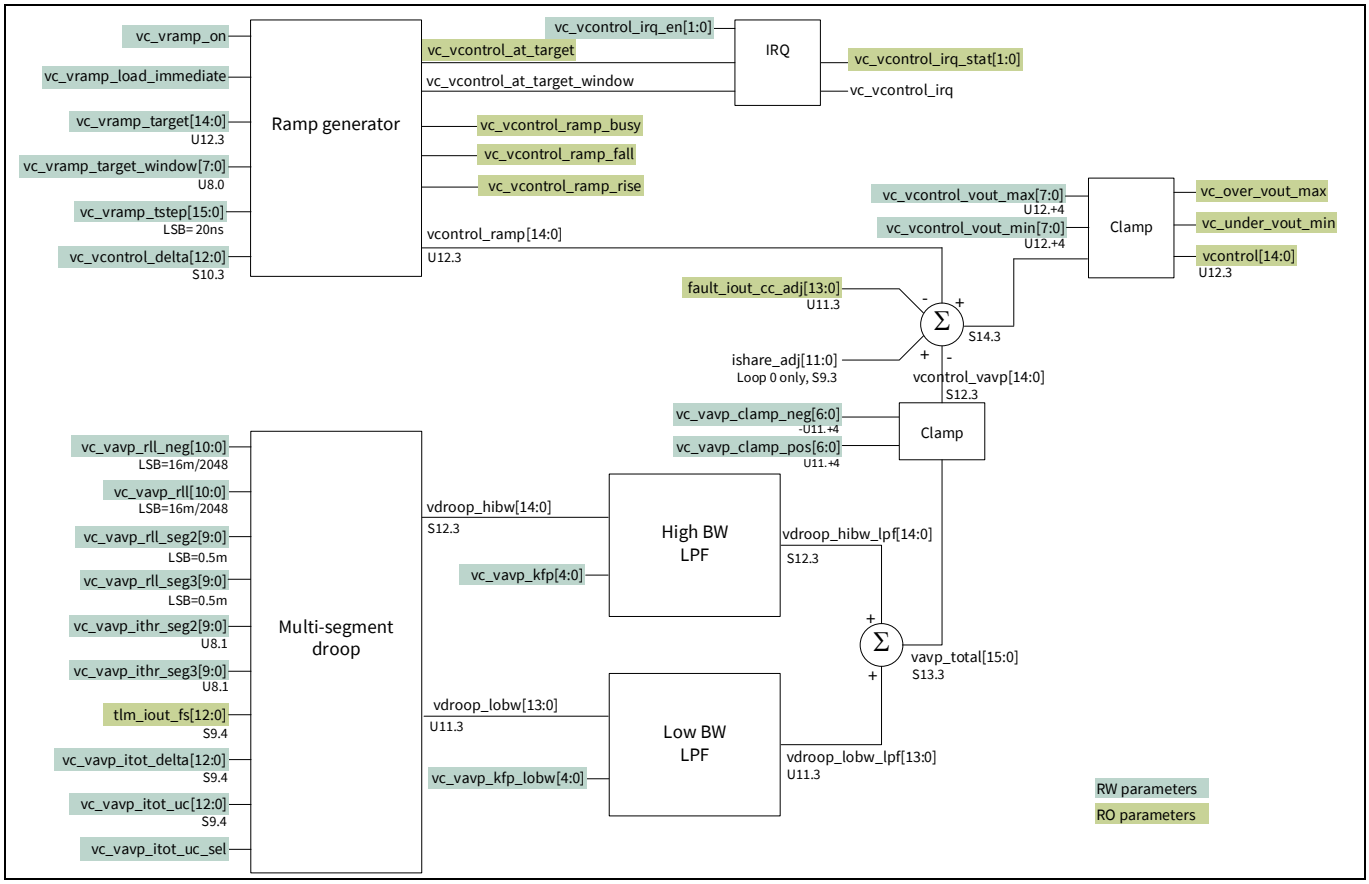

Figure 37 Voltage control block diagram

The control voltage, $V_{control}$ in the figure, is needed for the duty cycle definition. The VSP module (Section 2.1) compares the measured voltage to $V_{control}$ and forms the error voltage which is then used by the compensator to define the duty cycle. Voltage parameters within the voltage control module are defined relative to the sense voltage (VSEN or BVSEN) after the output voltage resistor divider scaling.

5.1 PMBus commands to HW parameters

The voltage control module implements the PMBus reference voltage generation model from a high-level perspective, as described in the PMBus Specification, Part II, Revision 1.3.1, section 9.2 and shown in Figure 38. PMBus commands are converted to module input parameters by FW, as shown in Figure 39. Details of the command to parameter computations for individual parameters can be found in Section 5.6.

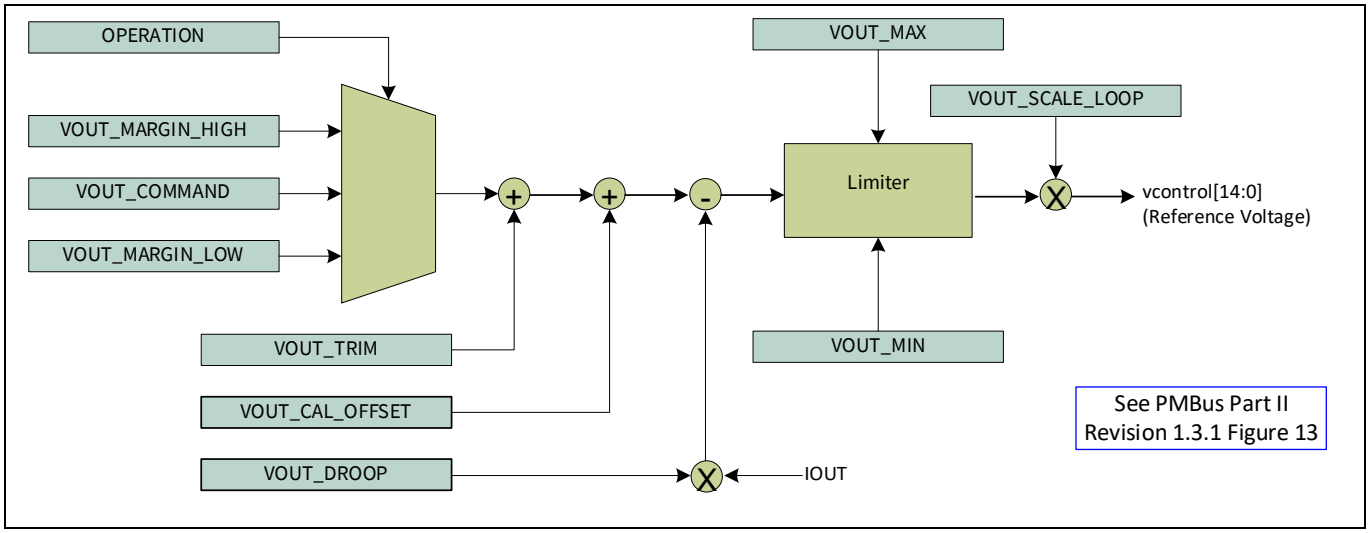

Figure 38 PMBus target voltage generation

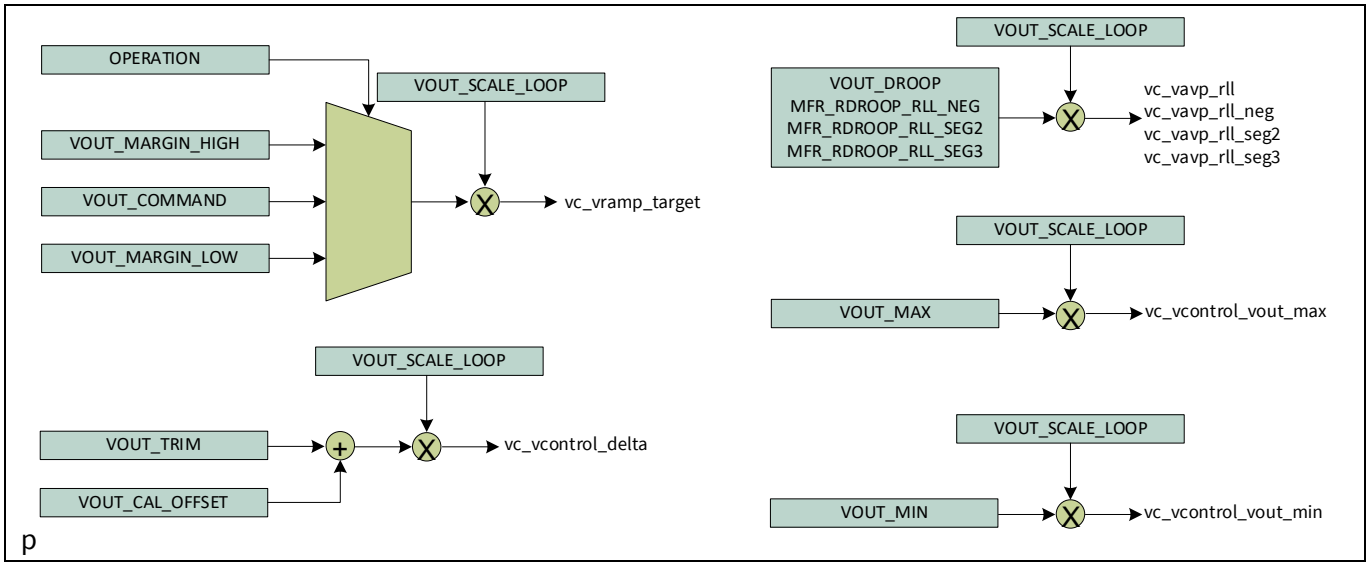

Figure 39 Mapping of PMBus commands to voltage control parameter inputs

5.2 Ramp generator

The ramp generator is responsible for slewing the target output voltage during soft-start and shutdown, as well as in response to commanded output voltage changes. For example:

- During soft-start; from 0 V (or pre-bias voltage) to VOUT_COMMAND at the TON_RISE rate
- During regulation; between two VOUT_COMMAND voltages at VOUT_TRANSITION_RATE
- During shutdown; from VOUT_COMMAND to 0 V at the TOFF_FALL rate

A simplified block diagram of the ramp generator is shown in Figure 40, where the target output voltage consists of the sum of the following registers:

- **vc_vramp_target**, which is the VSEN referenced target voltage computed by FW from PMBus commands according to Equation (5.1)
- **vc_vcontrol_delta**, which is the VSEN referenced offset voltage computed by FW from PMBus commands according to Equation (5.2)

$$vc_vramp_target = VOUT_COMMAND * VOUT_SCALE_LOOP * \frac{6400}{2^{(16-VOUT_MODE[4:0])}} \quad (5.1)$$

$$vc_vcontrol_delta = (VOUT_TRIM + VOUT_CAL_OFFSET) * VOUT_SCALE_LOOP * \frac{6400}{2^{(16-VOUT_MODE[4:0])}} \quad (5.2)$$

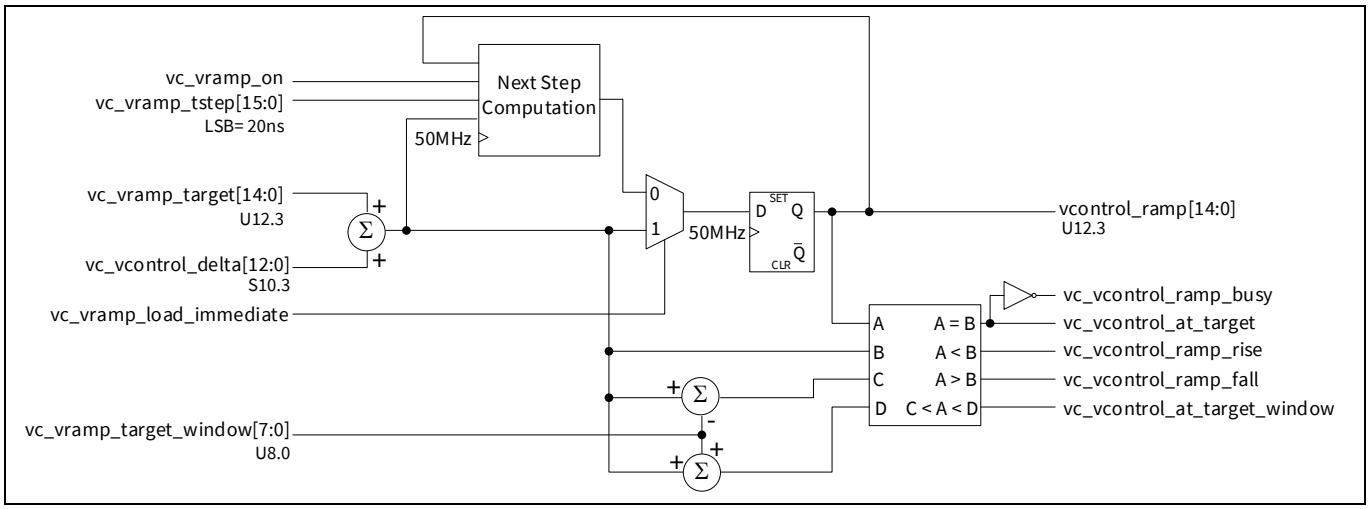

Figure 40 Ramp generator diagram

The slew rate on the ramp generator output, vcontrol_ramp, is programmed via register **vc_vramp_tstep**. It defines the time between 1.25 mV steps (referenced to VSEN) and it can be computed based on the PMBus commands according to the following equations:

- During soft-start: Equation (5.3)
- During shutdown: Equation (5.4)
- During regulation: Equation (5.5)

$$vc_vramp_tstep = \left[\frac{62.5 * 2^{22} * TON_RISE}{VOUT_COMMAND * VOUT_SCALE_LOOP}\right] - 1 \quad (5.3)$$

$$vc_vramp_tstep = \left[\frac{62.5 * 2^{22} * TOFF_FALL}{VOUT_COMMAND * VOUT_SCALE_LOOP}\right] - 1 \quad (5.4)$$

$$vc_vramp_tstep = \left[\frac{62.5 * 2^{19}}{VOUT_TRANSITION_RATE * VOUT_SCALE_LOOP}\right] - 1 \quad (5.5)$$

The programmable slew rate on the vcontrol_ramp is possible to bypass through register **vc_vramp_load_immediate**. Setting this value high immediately loads the **vc_vramp_target** into ramp output, bypassing the next step computation function. Otherwise, with the **vc_vramp_load_immediate** value low, the ramp output is slewed to **vc_vramp_target** at a rate determined by register **vc_vramp_tstep**. The intended use for this feature is to allow the FW to start the soft-start ramp from the measured pre-bias voltage.

Care should be taken if using this feature during active regulation, because the power stage may not be capable of tracking large steps in the target voltage without significant overshoot/undershoot.

The ramp generator also provides the following read-only status registers:

- **vc_vcontrol_at_target** indicates vcontrol_ramp is at the target voltage and is no longer slewing
- **vc_vcontrol_ramp_busy** indicates vcontrol_ramp is slewing to a new target
- **vc_vcontrol_ramp_rise** and **vc_vcontrol_ramp_fall** indicate the direction of the vcontrol_ramp change

5.3 Interrupts

The voltage control interrupt generation is illustrated in Figure 41. The interrupts are generated based on the following ramp generator outputs:

- **vc_vcontrol_at_target**, which is the previously discussed read-only register
- **vc_vcontrol_at_target_window**, which is an internal signal that indicates vcontrol_ramp is within a window around the target voltage

The target voltage window is defined by parameter **vc_vramp_target_window**, and this indicator may be used to change system behavior prior to arriving at the target voltage, such as turning on the SR FETs during the diode emulation soft start.

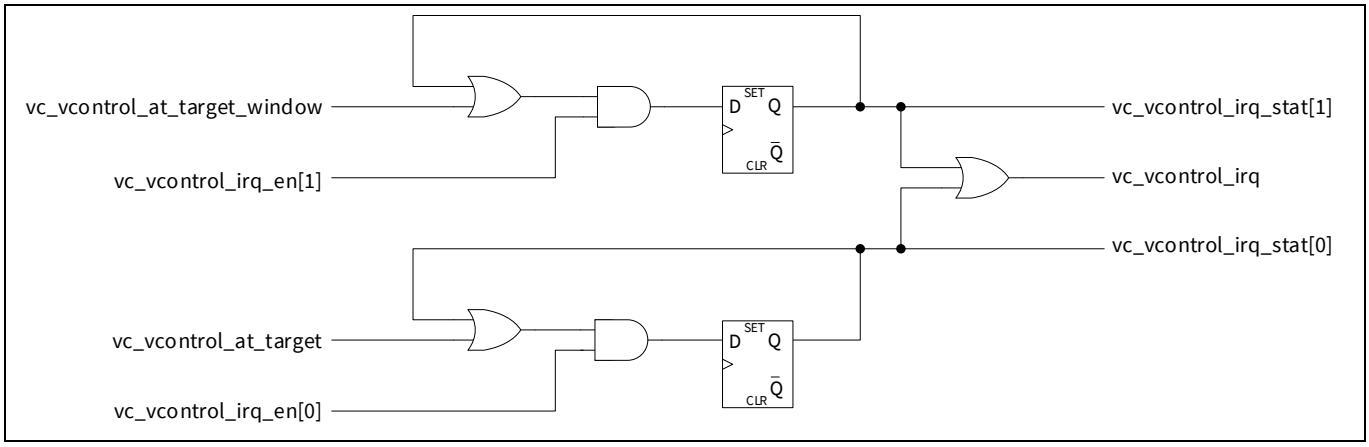

Figure 41 Interrupts diagram

An interrupt is enabled by setting the appropriate bit high in the register **vc_vcontrol_irq_en**, and correspondingly interrupts are cleared by setting the same bit low in the register **vc_vctonrol_irq_en**. The interrupt status is observable via read-only register **vc_vcontrol_irq_status**.

5.4 Multi-segment droop (load-line)

The XDP1100 multi-segment droop module allows programming of three independent droop resistances at positive current as well as an independent negative current droop resistance, as shown in Figure 42.

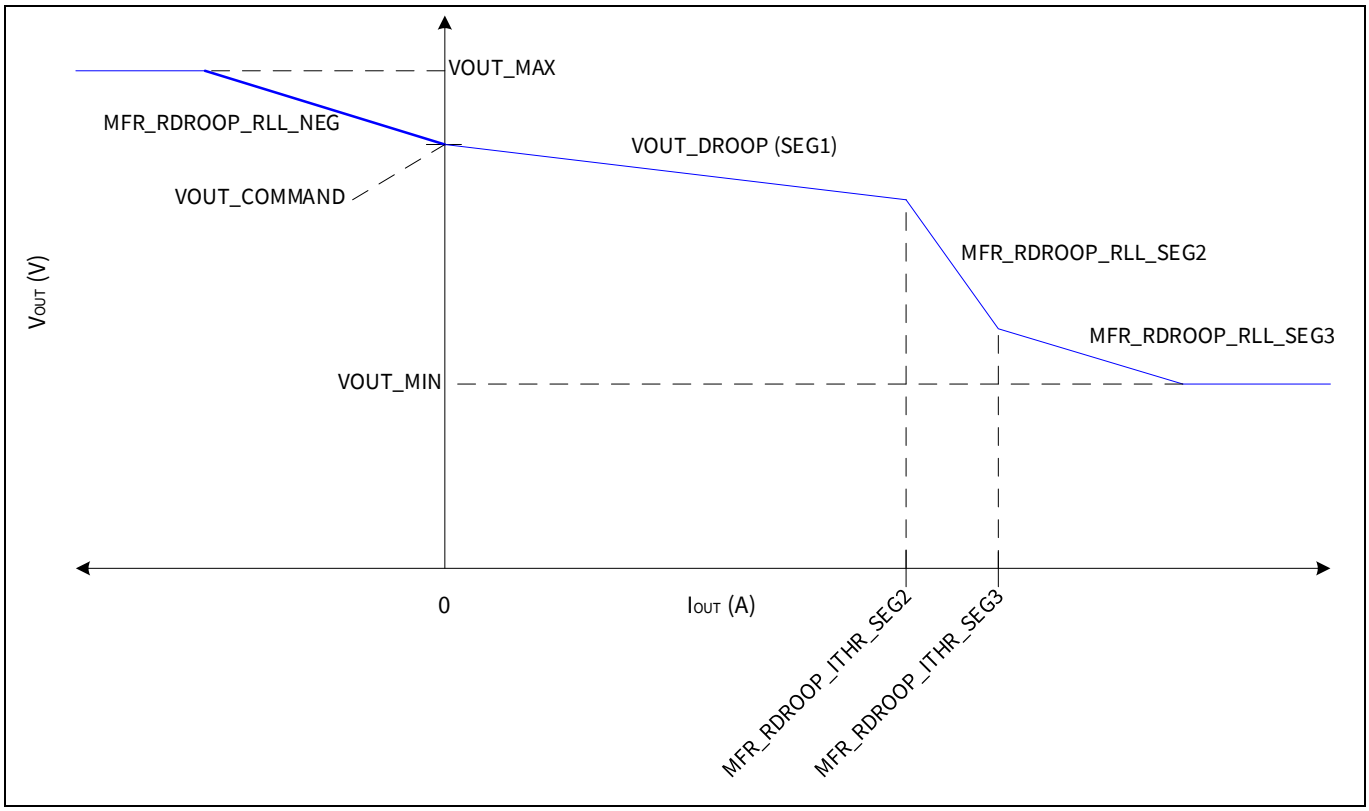

Figure 42 Multi-segment droop: V_{OUT} vs. I_{OUT}

The first positive segment (SEG1) is the standard operating droop, which is programmed through the PMBus command VOUT_DROOP. The higher-current segments (SEG2 and SEG3) allow a linear approximation to constant current and constant power operation before the output is clamped to VOUT_MIN. The negative current segment allows the negative current droop to be independently programmed according to the system requirements.

5.4.1 Droop voltage computation

The droop resistance parameters are computed for each segment using the PMBus commands according to the following equations:

- **SEG1:** Equations (5.6) and (5.7)
- **SEG2:** Equations (5.8) and (5.9)
- **SEG3:** Equations (5.10) and (5.11)
- **Negative segment:** Equations (5.12) and (5.13)

$$vc_vavp_rll = \frac{VOUT_DROOP * VOUT_SCALE_LOOP}{2^{(y+9)}} \quad (5.6)$$

$$y = -1 * LINEAR11\ exponent\ of\ VOUT_DROOP \quad (5.7)$$

$$vc_vavp_rll_seg2 = \frac{MFR_RDROOP_RLL_SEG2 * VOUT_SCALE_LOOP}{2^{(y+15)}} \quad (5.8)$$

$$y = -1 * LINEAR11\ exponent\ of\ MFR_RDROOP_RLL_SEG2 \quad (5.9)$$

$$vc_vavp_rll_seg3 = \frac{MFR_RDROOP_RLL_SEG3 * VOUT_SCALE_LOOP}{2^{(y+15)}} \quad (5.10)$$

$$y = -1 * LINEAR11\ exponent\ of\ MFR_RDROOP_RLL_SEG3 \quad (5.11)$$

$$vc_vavp_rll_neg = \frac{MFR_RDROOP_RLL_NEG * VOUT_SCALE_LOOP}{2^{(y+9)}} \quad (5.12)$$

$$y = -1 * LINEAR11\ exponent\ of\ MFR_RDROOP_RLL_NEG \quad (5.13)$$

It should be noted that the **vc_vavp_rll_xxx** registers are relative to VSEN while the PMBus droop commands are relative to the output voltage. The block diagram of the droop voltage computation is shown in Figure 43.

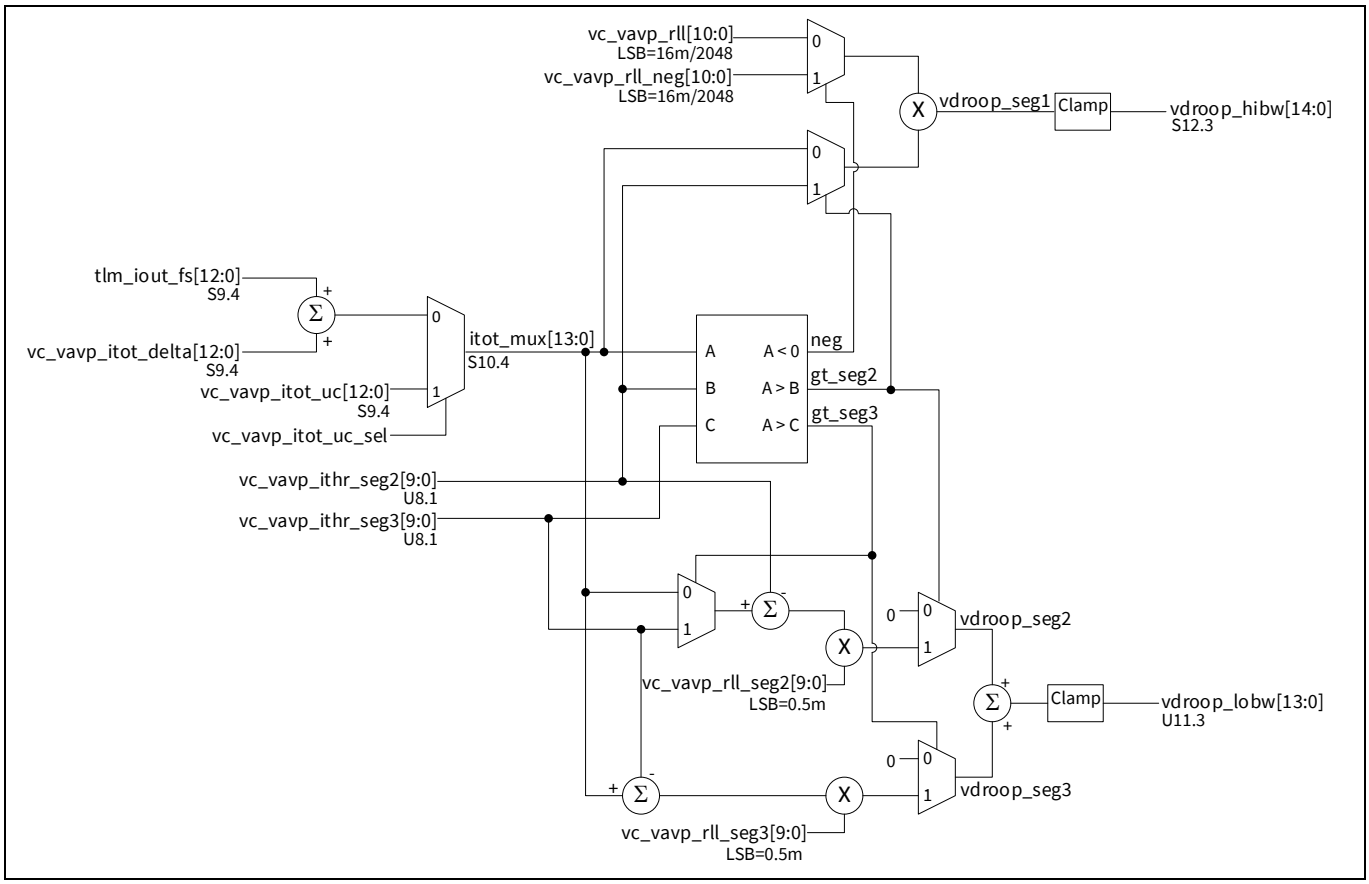

Figure 43 Multi-segment droop block diagram

The current thresholds between the segments are defined through the following registers:

- **vc_vavp_ithr_seg2** for SEG1/SEG2
- **vc_vavp_ithr_seg3** for SEG2/SEG3

They are programmed via PMBus commands MFR_RDROOP_ITHR_SEG2 and MFR_RDROOP_ITHR_SEG3, respectively. The output current used for the droop computation, **tlm_iout_fs**, comes from the telemetry module. It is the switching-cycle averaged, unfiltered representation of I_{OUT}. The FW can add an offset to the measured output current via register **vc_vavp_itot_delta**. In addition, the FW can completely override the measured I_{OUT} for use in the droop computation through registers **vc_vavp_itot_uc** and **vc_vavp_itot_uc_sel**.

To determine in which current segment the XDPP1100 is operating, the current itot_mux is compared to the segment current thresholds. The computed droop voltages in each of the four segments are described in Table 24. The negative and SEG1 droop voltage (output on vdroop_hibw) is computed independently from the SEG2/SEG3 droop voltage (output on vdroop_lobw). Segments 2 and 3 typically have higher droop resistance requiring lower BW filtering to prevent output voltage oscillation.

Table 24 Computed droop voltage by segment at high BW and low BW LPF inputs

Segment	Current range	vdroop_hibw	vdroop_lobw
neg	I < 0	I * rll_neg	0
seg1	0 < I < ithr_seg2	I * rll	0
seg2	ithr_seg2 < I ithr_seg3	ithr_seg2 * rll	(I - ithr_seg2) * rll_seg2
seg3	I > ithr_seg3	ithr_seg2 * rll	(ithr_seg3 - ithr_seg2) * rll_seg2 + (I - ithr_seg3) * rll_seg3

5.4.2 Droop voltage filtering

The droop voltage outputs go to two independent low-pass filters (LPFs) as shown in Figure 37 and discussed in the previous subsection. The filter BWs are programmed through registers:

- **vc_vavp_kfp** for the high BW LPF according to Table 25
- **vc_vavp_kfp_lobw** for the low BW LPF according to Table 26

Table 25 High BW AVP filter programming (F_s = 25 MHz and LSB = 6.104e-5)

vc_vavp_kfp	kfp	kfp_real	F3db (kHz)	vc_vavp_kfp	kfp	kfp_real	F3db (kHz)
0	4	0.0002	0.97	16	64	0.0039	15.60
1	5	0.0003	1.21	17	80	0.0049	19.52
2	6	0.0004	1.46	18	96	0.0059	23.45
3	7	0.0004	1.70	19	112	0.0068	27.39
4	8	0.0005	1.94	20	128	0.0078	31.33
5	10	0.0006	2.43	21	160	0.0098	39.24
6	12	0.0007	2.92	22	192	0.0117	47.18
7	14	0.0009	3.40	23	224	0.0137	55.15
8	16	0.0010	3.89	24	256	0.0156	63.16
9	20	0.0012	4.86	25	320	0.0195	79.26
10	24	0.0015	5.84	26	384	0.0234	95.49
11	28	0.0017	6.81	27	448	0.0273	111.86
12	32	0.0020	7.79	28	512	0.0313	128.35
13	40	0.0024	9.74	29	640	0.0391	161.74
14	48	0.0029	11.69	30	768	0.0469	195.68
15	56	0.0034	13.65	31	na	na	bypass

Table 26 Low BW AVP filter programming (F_s = 25 MHz and LSB = 7.629e-6)

vc_vavp_kfp	kfp	kfp_real	F3db (kHz)	vc_vavp_kfp	kfp	kfp_real	F3db (kHz)
0	4	0.0000	0.12	16	64	0.0005	1.94
1	5	0.0000	0.15	17	80	0.0006	2.43
2	6	0.0000	0.18	18	96	0.0007	2.92
3	7	0.0001	0.21	19	112	0.0009	3.40
4	8	0.0001	0.24	20	128	0.0010	3.89

vc_vavp_kfp	kfp	kfp_real	F3db (kHz)	vc_vavp_kfp	kfp	kfp_real	F3db (kHz)
5	10	0.0001	0.30	21	160	0.0012	4.86
6	12	0.0001	0.36	22	192	0.0015	5.84
7	14	0.0001	0.43	23	224	0.0017	6.81
8	16	0.0001	0.49	24	256	0.0020	7.79
9	20	0.0002	0.61	25	320	0.0024	9.74
10	24	0.0002	0.73	26	384	0.0029	11.69
11	28	0.0002	0.85	27	448	0.0034	13.65
12	32	0.0002	0.97	28	512	0.0039	15.60
13	40	0.0003	1.21	29	640	0.0049	19.52
14	48	0.0004	1.46	30	768	0.0059	23.45
15	56	0.0004	1.70	31	na	na	bypass

The LPF outputs are summed to create the total droop voltage, vavp_total, which is further clamped by the following registers to create vcontrol_vavp:

- **vc_vavp_clamp_pos**, which clamps the positive droop defined as decreasing voltage due to positive I_{OUT}
- **vc_vavp_clamp_neg**, which clamps the negative droop defined as increasing voltage due to negative I_{OUT}

5.5 Output summation and clamping

The final $V_{control}$ output is a sum of:

- vcontrol_ramp, which is the output of the ramp generator
- vcontrol_vavp, which results from the droop computations
- fault_iout_cc_adj, which is the I_{OUT} fault constant current voltage adjustment (discussed in Section 9)
- ishare_adj, which is the current sharing voltage adjustment (discussed in Section 10)

The summation and clamping of the final $V_{control}$ is shown in Figure 44.

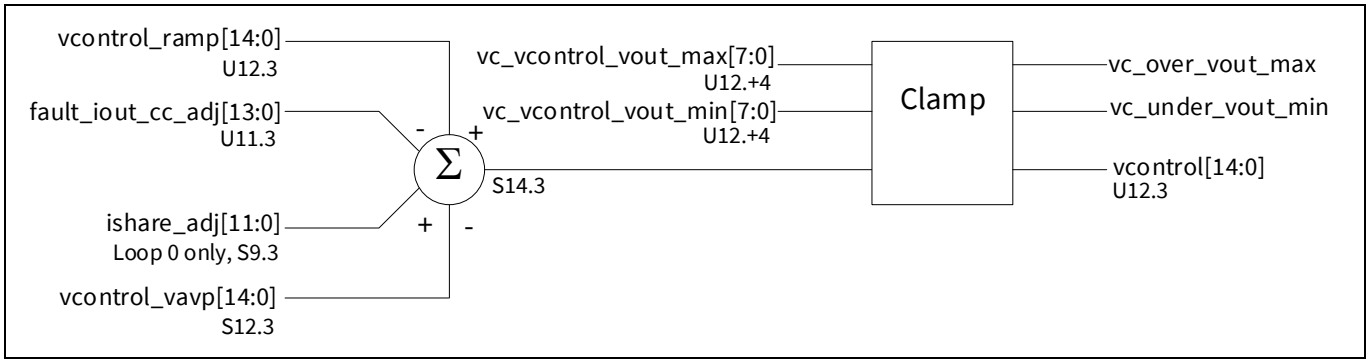

Figure 44 Final voltage summation and clamping

Subsequent to the summation, a final set of clamps is applied. The maximum and minimum clamp values can be computed from PMBus commands according to Equations (5.14) and (5.15) and set via the following registers:

- **vc_vcontrol_vout_max** according to Equation (5.14)
- **vc_vcontrol_vout_min** according to Equation (5.15)

$$vc_vcontrol_vout_max = \frac{VOUT_MAX * VOUT_SCALE_LOOP * 50}{2^{(24)}} \quad (5.14)$$

$$vc_vcontrol_vout_min = \frac{VOUT_MIN * VOUT_SCALE_LOOP * 50}{2^{(24)}} \quad (5.15)$$

Read-only registers **vc_over_vout_max** and **vc_under_vout_min** indicate when these clamp voltages are exceeded at the summation output and the clamp is applied. Additionally, they go to the fault module for reporting on STATUS_VOUT.

5.6 Voltage control registers

The relevant voltage control module registers and their descriptions are provided in Table 27.

Table 27 Voltage control module-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
vcontrol	vc_vavp_clamp_neg	RW	7000_1400h (vcontrol0) 7000_1800h (vcontrol1)	[6:0]	Negative droop (load-line) clamp voltage can be used to limit negative droop voltage independent of VOUT_MAX (e.g., set to 0 to disable negative droop). **Note:** Positive droop refers to decreasing voltage with positive I_{OUT}, negative droop refers to increasing voltage with negative I_{OUT}. LSB = -20 mV, range = 0.0 to -2.54 V
vcontrol	vc_vavp_clamp_pos	RW	7000_1400h (vcontrol0) 7000_1800h (vcontrol1)	[13:7]	Positive droop (load-line) clamp voltage can be used to limit positive droop voltage independent of VOUT_MIN. **Note:** Positive droop refers to decreasing voltage with positive I_{OUT}, negative droop refers to increasing voltage with negative I_{OUT}. LSB = 20 mV, range = 0.0 to 2.54 V
vcontrol	vc_vavp_kfp	RW	7000_1400h (vcontrol0) 7000_1800h (vcontrol1)	[18:14]	LPF coefficient for “high” BW filter applied to the negative and VOUT_DROOP segments, set to all 1s to bypass. kfp_exp = vavp_kfp_i[4:2], kfp_man = 4 + vavp_kfp_i[1:0] kfp = kfp_man * 2^kfp_exp * 2^-14 F3db(MHz) = [kfp/(1-kfp)] * 25 MHz / 2*pi

Peripheral	Field name	Access	Address	Bits	Description
vcontrol	vc_vavp_kfp_lobw	RW	7000_1400h (vcontrol0) 7000_1800h (vcontrol1)	[23:19]	LPF coefficient for “low” BW filter applied to the higher-gain segments 2 and 3, set to all 1s to bypass. kfp_exp = vavp_kfp_i[4:2], kfp_man = 4 + vavp_kfp_i[1:0] kfp = kfp_man * 2^kfp_exp * 2^-17 F3db(MHz) = [kfp/(1-kfp)] * 25 MHz / 2 * pi
vcontrol	vc_vramp_target	RW	7000_1404h (vcontrol0) 7000_1804h (vcontrol1)	[14:0]	Ramp target value referenced to VSEN input (i.e., after scaling by VOUT_SCALE_LOOP). Note that while digitally this command can handle a wider range, the target value is limited to 2.1 V by the analog circuitry. **Computed by FW from PMBus commands as follows:** vc_vramp_target(U12.3) = VOUT_COMMAND(U16-X.X) * VOUT_SCALE_LOOP(U0.16) * 6400 / (2^X * 2^16) where, X = negative of VOUT_MODE exponent **Note:** VOUT_COMMAND above may be replaced with VOUT_MARGIN_HIGH or VOUT_MARGIN_LOW based on OPERATION setting LSB = 0.15625 mV, range = 0.0 to 2.1 V
vcontrol	vc_vramp_target_window	RW	7000_1404h (vcontrol0) 7000_1804h (vcontrol1)	[22:15]	Ramp target window, used to define interrupt prior to reaching ramp target (e.g., during soft-start may be used by FW to enable SR FETs prior to reaching final target). LSB = 1.25 mV, range = 0.0 to 318.75 mV
vcontrol	vc_vramp_load_immediate	RW	7000_1404h (vcontrol0) 7000_1804h (vcontrol1)	[23]	Immediately loads vc_vramp_target into ramp output when high. Otherwise the ramp output is slewed to vc_vramp_target at a rate determined by vc_vramp_tstep. The typical use for this field is during pre-bias start-up. The pre-

Peripheral	Field name	Access	Address	Bits	Description
					bias voltage can be immediately loaded into the ramp to define the ramp start voltage, then the target voltage (e.g., VOUT_COMMAND) can be set on vc_vramp_target with load_immediate set low to slew to the final voltage. 0: Slew ramp output to vc_vramp_target 1: Immediately set ramp output to vc_vramp_target
vcontrol	vc_vramp_on	RW	7000_1404h (vcontrol0) 7000_1804h (vcontrol1)	[24]	When high, indicates the ramp should respond to target changes and that that loop is actively regulating. **Note:** In addition to $V_{control}$, this signal is used by the PID, PWM and CE as an indication that we are actively regulating the output.
vcontrol	vc_vramp_tstep	RW	7000_1408h (vcontrol0) 7000_1808h (vcontrol1)	[15:0]	Sets the output voltage slew rate by defining the time between 1.25 mV steps (referenced to VSEN). **Computed by FW from PMBus commands as follows:** **Using VOUT_TRANSITION_RATE:** tstep(U18.-2) = [(1.25 mV/0.02 µs) * (2^3 * 2^16) / (VOUT_TRANSITION_RATE(U6.3) * VOUT_SCALE_LOOP(U0.16))] - 1 = [62.5 * 2^19 / (VOUT_TRANSITION_RATE(U6.3) * VOUT_SCALE_LOOP(U0.16))] - 1 **Using TON_RISE assuming ramping from 0 V:** tstep(U18.-2) = [(1.25 mV/0.02 µs) * (2^-2 * 2^8 * 2*16) * TON_RISE(U5.2) / (VOUT_COMMAND(U8.8) * VOUT_SCALE_LOOP(U0.16))] - 1 = [62.5 * 2^22 * TON_RISE(U5.2) / (VOUT_COMMAND(U8.8) * VOUT_SCALE_LOOP(U0.16))] - 1

Peripheral	Field name	Access	Address	Bits	Description
					Using TOFF_FALL assuming ramping to 0 V: tstep(U18.-2) = [(1.25 mV/0.02 µs) * (2^-2 * 2^8 * 2*16) * TOFF_FALL(U5.2) / (VOUT_COMMAND(U8.8) * VOUT_SCALE_LOOP(U0.16))] - 1 = [62.5 * 2^22 * TOFF_FALL(U5.2) / (VOUT_COMMAND(U8.8) * VOUT_SCALE_LOOP(U0.16))] - 1 **Note:** VOUT_COMMAND above may be replaced with VOUT_MARGIN_HIGH or VOUT_MARGIN_LOW based on OPERATION setting LSB = 20 ns, range = 0.0 to 1.3107 ms
vcontrol	vc_vavp_itot_delta	RW	7000_140Ch (vcontrol0) 7000_180Ch (vcontrol1)	[12:0]	Total current offset term applied to droop computations only. Allows FW to apply offset to HW computed current. LSB = 62.5 mA, range = -256 to +255.9375 A
vcontrol	vc_vavp_rll	RW	7000_1410h (vcontrol0) 7000_1810h (vcontrol1)	[10:0]	Load-line (droop) resistance referenced to VSEN input (i.e., after scaling by VOUT_SCALE_LOOP). **Computed by FW from PMBus commands as follows:** vc_vavp_rll(U4.7) = VOUT_DROOP(Ux.y) * VOUT_SCALE_LOOP(U0.16) / (2^y * 2^(16-7)) where, y = -1 * LINEAR11 exponent of VOUT_DROOP LSB = 7.8125 µΩ, range = 0.0 to 15.992 mΩ
vcontrol	vc_vcontrol_delta	RW	7000_1414h (vcontrol0) 7000_1814h (vcontrol1)	[12:0]	$V_{control}$ output offset adjust referenced to the VSEN input (i.e., after scaling by VOUT_SCALE_LOOP). **Computed by FW from PMBus commands as follows:** vcontrol_delta(S10.0) = (VOUT_TRIM(S16-X.X) + VOUT_CAL_OFFSET(S16-X.X)) *

Peripheral	Field name	Access	Address	Bits	Description
					VOUT_SCALE_LOOP(U0.16) * 6400 / (2^X * 2^16) where, X = negative of VOUT_MODE exponent LSB = 0.15625 mV, range = -640 to +639.84375 mV
vcontrol	vc_vcontrol_vout_max	RW	7000_1418h (vcontrol0) 7000_1818h (vcontrol1)	[7:0]	V_{OUT} max. limit referenced to VSEN input (i.e., after scaling by VOUT_SCALE_LOOP). Target voltages (including droop) above this limit will be clamped to this level. **Computed by FW from PMBus commands as follows:** vout_max(U12.-4) = VOUT_MAX(U8.8) * VOUT_SCALE_LOOP(U0.16) * 50 / (2^8 * 2^16) LSB = 20 mV, range = 0.0 to 5.1 V
vcontrol	vc_vcontrol_vout_min	RW	7000_1418h (vcontrol0) 7000_1818h (vcontrol1)	[15:8]	V_{OUT} min. limit referenced to VSEN input (i.e., after scaling by VOUT_SCALE_LOOP. Target voltages (including droop) below this limit will be clamped to this level. **Computed by FW from PMBus commands as follows:** vout_min(U12.-4) = VOUT_MIN(U8.8) * VOUT_SCALE_LOOP(U0.16) * 50 / (2^8 * 2^16) LSB = 20 mV, range = 0.0 to 5.1 V
vcontrol	vc_vavp_rll_neg	RW	7000_141Ch (vcontrol0) 7000_181Ch (vcontrol1)	[10:0]	Load-line (droop) resistance referenced to VSEN input (i.e., after scaling by VOUT_SCALE_LOOP) when load current is negative. **Computed by FW from PMBus commands as follows:** vc_vavp_rll_neg(U4.7) = MFR_RDROOP_RLL_NEG(Ux.y) * VOUT_SCALE_LOOP(U0.16) / (2^y * 2^(16-7)) where, y = -1 * LINEAR11 exponent of MFR_RDROOP_RLL_NEG LSB = 7.8125 µΩ, range = 0.0 to 15.9922 mΩ

Peripheral	Field name	Access	Address	Bits	Description
vcontrol	vc_vcontrol_irq_en	RW	7000_1420h (vcontrol0) 7000_1820h (vcontrol1)	[1:0]	Interrupt enable for $V_{control}$. When the bit indicated is high, the corresponding signal is enabled to generate an interrupt. 0: Ramp at target 1: Ramp within vc_vramp_target window of target voltage
vcontrol	vc_vcontrol_irq_stat	R	7000_1424h (vcontrol0) 7000_1824h (vcontrol1)	[1:0]	Interrupt status for $V_{control}$. 0: Ramp at target 1: Ramp within vc_vramp_target window of target voltage
vcontrol	vc_over_vout_max	R	7000_1424h (vcontrol0) 7000_1824h (vcontrol1)	[2]	V_{OUT} max. status flag 0: $V_{control}$ output below vc_vout_max 1: $V_{control}$ output clamped to vc_vout_max
vcontrol	vc_under_vout_min	R	7000_1424h (vcontrol0) 7000_1824h (vcontrol1)	[3]	V_{OUT} min. status flag 0: Commanded target above vc_vout_min 1: Commanded target below vc_vout_min, will be clamped to vout_min (target of 0 is an exception that will not be clamped)
vcontrol	vc_vcontrol_at_target	R	7000_1424h (vcontrol0) 7000_1824h (vcontrol1)	[4]	Status flag indicating the $V_{control}$ ramp is at the target voltage. 0: Ramp not at target voltage 1: Ramp at target voltage
vcontrol	vc_vcontrol_ramp_busy	R	7000_1424h (vcontrol0) 7000_1824h (vcontrol1)	[5]	Status flag indicating the $V_{control}$ ramp is slewing (inverse of vc_vcontrol_at_target). 0: Ramp not slewing 1: Ramp slewing
vcontrol	vc_vcontrol_ramp_fall	R	7000_1424h (vcontrol0) 7000_1824h (vcontrol1)	[6]	Status flag indicating the $V_{control}$ ramp is slewing in a falling direction. 0: Ramp not falling (i.e., rising or at target) 1: Ramp falling
vcontrol	vc_vcontrol_ramp_rise	R	7000_1424h (vcontrol0) 7000_1824h (vcontrol1)	[7]	Status flag indicating the $V_{control}$ ramp is slewing in a rising direction. 0: Ramp not rising (i.e., falling or at target) 1: Ramp rising

Peripheral	Field name	Access	Address	Bits	Description
vcontrol	vc_vcontrol	R	7000_1428h (vcontrol0) 7000_1828h (vcontrol1)	[14:0]	$V_{control}$ status register. Reflects target voltage referenced to the VSEN input (i.e., after scaling by VOUT_SCALE_LOOP) including offsets, droop, etc. LSB = 0.15625 mV, range = 0.0 to 5.119844 V
vcontrol	vc_vavp_rll_seg2	RW	7000_142Ch (vcontrol0) 7000_182Ch (vcontrol1)	[9:0]	Load-line (droop) resistance referenced to VSEN input (i.e., after scaling by VOUT_SCALE_LOOP) when load current exceeds vc_vavp_ithr_seg2. **Computed by FW from PMBus commands as follows:** vc_vavp_rll_seg2(U9.1) = MFR_RDROOP_RLL_SEG2(Ux.y) * VOUT_SCALE_LOOP(U0.16) / (2^y * 2^(16-1)) where y = -1 * LINEAR11 exponent of MFR_RDROOP_RLL_SEG2 LSB = 0.5 mΩ, range = 0.0 to 511.5 mΩ
vcontrol	vc_vavp_rll_seg3	RW	7000_1430h (vcontrol0) 7000_1830h (vcontrol1)	[9:0]	Load-line (droop) resistance referenced to VSEN input (i.e., after scaling by VOUT_SCALE_LOOP) when load current exceeds vc_vavp_ithr_seg3. **Computed by FW from PMBus commands as follows:** vc_vavp_rll_seg3(U9.1) = MFR_RDROOP_RLL_SEG3(Ux.y) * VOUT_SCALE_LOOP(U0.16) / (2^y * 2^(16-1)) where y = -1 * LINEAR11 exponent of MFR_RDROOP_RLL_SEG3 LSB = 0.5 mΩ, range = 0.0 to 511.5 mΩ
vcontrol	vc_vavp_ithr_seg2	RW	7000_1434h (vcontrol0) 7000_1834h (vcontrol1)	[8:0]	When load current is above vc_vavp_ithr_seg2 and less than or equal to vc_vavp_ithr_seg3 the load-line (droop) resistance will be defined by vc_vavp_rll_seg2. Note that a

Peripheral	Field name	Access	Address	Bits	Description
					setting of 0 will disable this segment. **Computed by FW from PMBus command as follows:** vc_vavp_ithr_seg2(U8.1) = MFR_RDROOP_ITHR_SEG2(UX.Y) * 2^Y where Y = LINEAR11 exponent of MFR_RDROOP_ITHR_SEG2 LSB = 0.5 A, range = 0.0 to 255.5 A
vcontrol	vc_vavp_ithr_seg3	RW	7000_1438h (vcontrol0) 7000_1838h (vcontrol1)	[8:0]	When load current is above vc_vavp_ithr_seg3 the load-line (droop) resistance will be defined by vc_vavp_rll_seg3. Note that a setting of 0 will disable this segment. **Computed by FW from PMBus command as follows:** vc_vavp_ithr_seg3(U8.1) = MFR_RDROOP_ITHR_SEG3(UX.Y) * 2^Y where Y = LINEAR11 exponent of MFR_RDROOP_ITHR_SEG3 LSB = 0.5 A, range = 0.0 to 255.5 A
vcontrol	vc_vavp_itot_uc	RW	7000_143Ch (vcontrol0) 7000_183Ch (vcontrol1)	[12:0]	Total current replacement term. Allows FW to override HW computed total current for droop computation when selected by vc_vavp_itot_uc_sel. LSB = 62.5 mA, range = -256 to +2559375 A
vcontrol	vc_vavp_itot_uc_sel	RW	7000_143Ch (vcontrol0) 7000_183Ch (vcontrol1)	[13]	Selects total current used by AVP (droop) function 0: HW computed current +vc_vavp_itot_delta 1: vc_vavp_otpt_uc_sel

5.7 Voltage control PMBus commands

The relevant voltage control-related PMBus commands are given in Table 28.

Table 28 Voltage control-related PMBus commands

Command name	Access	Length	Address	Bits	Description
VOUT_MODE	RW	Byte	20h	[7:0]	Defines the format of the V_{OUT}-related PMBus commands. [7:5] Only ULINEAR16 mode supported, x00b

Command name	Access	Length	Address	Bits	Description
					[4:0] Exponent of V_{OUT}-related commands, 14h to 18h (-8d to -12d) supported
VOUT_COMMAND	RW	Word	21h	[15:0]	Defines the output voltage when selected by the OPERATION command. Format is ULINEAR16 with the exponent defined by VOUT_MODE[4:0]
VOUT_TRIM	RW	Word	22h	[15:0]	Used to apply a fixed offset voltage to the output voltage command value. It is most typically used by the end user to trim the output voltage at the time the PMBus device is assembled into the end user's system. Format is SLINEAR16 with the exponent defined by VOUT_MODE[4:0]
VOUT_CAL_OFFSET	RW	Word	23h	[15:0]	Used to apply a fixed offset voltage to the output voltage command value. It is most typically used by the PMBus device manufacturer to calibrate a device in the factory. Format is SLINEAR16 with the exponent defined by VOUT_MODE[4:0]
VOUT_MAX	RW	Word	24h	[15:0]	Sets an upper limit on the output voltage the unit can command regardless of any other commands or combinations. The intent of this command is to provide a safeguard against a user accidentally setting the output voltage to a possibly destructive level rather than to be the primary output overvoltage protection. Format is ULINEAR16 with the exponent defined by VOUT_MODE[4:0]
VOUT_MARGIN_HIGH	RW	Word	25h	[15:0]	Defines the output voltage when selected by the OPERATION command. Format is ULINEAR16 with the exponent defined by VOUT_MODE[4:0]
VOUT_MARGIN_LOW	RW	Word	26h	[15:0]	Defines the output voltage when selected by the OPERATION command. Format is ULINEAR16 with the exponent defined by VOUT_MODE[4:0]
VOUT_TRANSITION_RATE	RW	Word	27h	[15:0]	When a PMBus device receives either a VOUT_COMMAND or OPERATION (margin high, margin low, margin off) that causes the output voltage to change, this command sets the rate in mV/µs at which the output should

Command name	Access	Length	Address	Bits	Description
					change voltage. This commanded rate of change does not apply when the unit is commanded to turn on or turn off. Format is LINEAR11 with exponent = -3.
VOUT_DROOP	RW	Word	28h	[15:0]	Sets the V_{OUT} droop rate in mV/A when I_{OUT} is positive but is less than MFR_RDROOP_ITHR_SEG2. Format is LINEAR11 with typical exponent in the range -7 to +2
VOUT_SCALE_LOOP	RW	Word	29h	[15:0]	Defines the V_{OUT} sense resistor divider ratio, V_{sense}/V_{OUT}. Format is DIRECT, U0.16. VOUT_SCALE_LOOP = INT(2^16 * V_{sense} / V_{OUT})
VOUT_MIN	RW	Word	2Bh	[15:0]	Sets a lower limit on the output voltage the unit can command regardless of any other commands or combinations. The intent of this command is to provide a safeguard against a user accidentally setting the output voltage to a possibly destructive level rather than to be the primary output undervoltage protection. Format is ULINEAR16 with the exponent defined by VOUT_MODE[4:0]
TON_RISE	RW	Word	61h	[15:0]	Sets the time, in ms, from when the output starts to rise until the voltage has entered the regulation band. Format is LINEAR11 with exponent = -2.
TOFF_FALL	RW	Word	65h	[15:0]	Sets the time, in ms, from the end of the turn-off delay time until the voltage is commanded to zero. Format is LINEAR11 with exponent = -2
FW_CONFIG_REGULATION	RW	Block 14 bytes	C5h	[47:32]	**MFR_RDROOP_RLL_SEG2**: Sets the V_{OUT} droop rate in mV/A when I_{OUT} exceeds MFR_RDROOP_ITHR_SEG2 but is less than MFR_RDROOP_ITHR_SEG3. Format is LINEAR11
				[63:48]	**MFR_RDROOP_RLL_SEG3**: Sets the V_{OUT} droop rate in mV/A when I_{OUT} exceeds MFR_RDROOP_ITHR_SEG3. Format is LINEAR11
				[79:64]	**MFR_RDROOP_RLL_NEG**: Sets the V_{OUT} droop rate in mV/A when I_{OUT} is negative. Format is LINEAR11

Command name	Access	Length	Address	Bits	Description
				[95:80]	**MFR_RDROOP_ITHR_SEG2**: Defines the output current threshold at which to switch droop resistance between VOUT_RDROOP and MFR_RDROOP_RLL_SEG2. Format is LINEAR11
				[111:96]	**MFR_RDROOP_ITHR_SEG3**: Defines the output current threshold at which to switch droop resistance between MFR_RDROOP_RLL_SEG2 and MFR_RDROOP_RLL_SEG3. Format is LINEAR11

6 Compensator

This chapter describes in more detail the XDPP1100 compensator implementation and main functionality. The relevant user-programmable parameters and corresponding registers are discussed as well as the input FF term and its relevant settings. Most of this chapter assumes that compensator output is directly the duty cycle as in VMC, and in Section 6.3, PCMC and its relevant compensator-related register settings are shown. PCMC is discussed in detail in the next chapter and is therefore only briefly mentioned in this chapter.

The simplified block diagram of the compensator is shown in Figure 45, and it consists of:

- Compensation filter, shown within the dashed lines
- Input voltage FF function

These two main subfunctions of the compensator are described in more detail in the sections below.

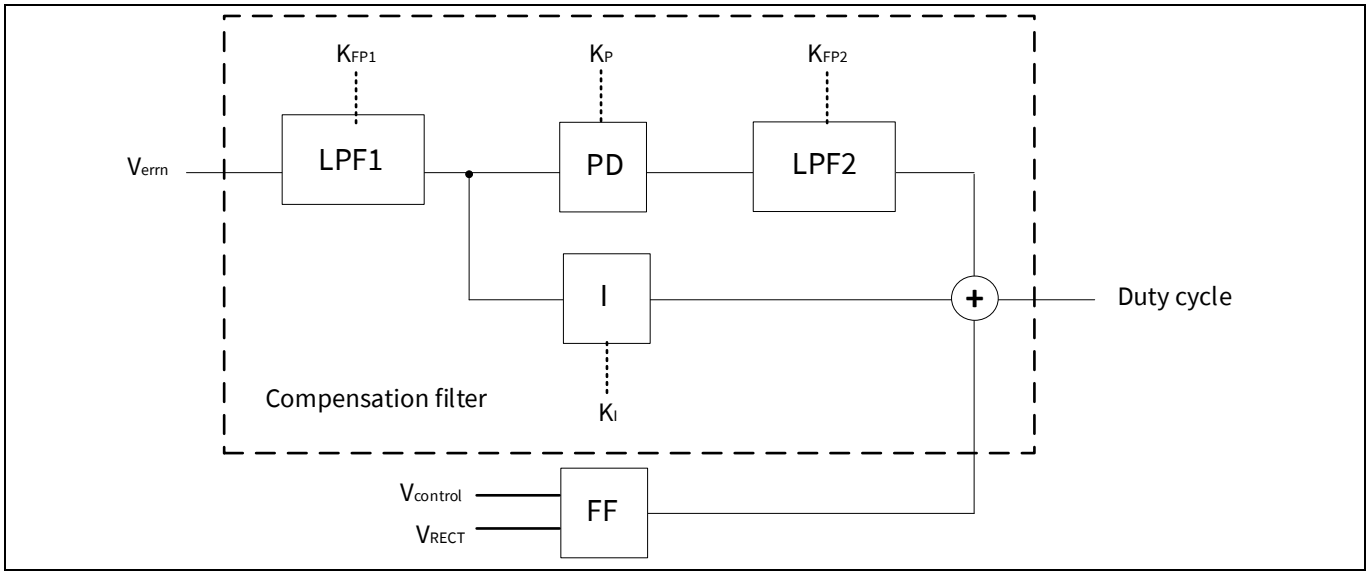

Figure 45 Functional block diagram of the compensator

6.1 Compensation filter

This section describes the proportional-integral-derivative (PID) filter implementation in more detail as well as its specific user-definable features and adjustments.

The compensation filter, inside the dashed lines in Figure 45, consists of:

- PID filter
- Pre- and post-LPFs

It receives as input the computed error signal, V_{errn}, obtained from the VS processor, discussed in Section 2.3.1. The compensator output is the target duty cycle, utilized by the PWM to generate output pulses.

The transfer function implemented by the compensation filter is provided in Equation (6.1).

$$\frac{DutyCycle}{Verrn} = \left[\frac{K_{FP1}}{1-(1-K_{FP1})z^{-1}}\right]\left[\left(K_P + K_D(1-z^{-1})\right)\left(\frac{K_{FP2}}{1-(1-K_{FP2})z^{-1}}\right) + \frac{K_I}{1-z^{-1}}\right] + FF \quad (6.1)$$

The terms K_P, K_D and K_I are the PID loop coefficients, and $K_{FP1,2}$ are the LPF coefficients which determine the locations of the poles and zeros. The zero locations are defined by the coefficients K_P, K_I and K_D, which are programmed via following registers:

- pid_kp_index_1ph for K_P
- pid_ki_index_1ph for K_I
- pid_kd_index_1ph for K_D

The PID provides a pole in the origin and two mid-band zeros while the LPFs present two high-frequency poles. This corresponds to the Type III compensation response which can provide up to 180 degrees of phase boost. A typical Type III compensator gain is illustrated in Figure 46, where f_{z1}, f_{z2}, f_{p1} and f_{p2} represent the pole and zero frequencies.

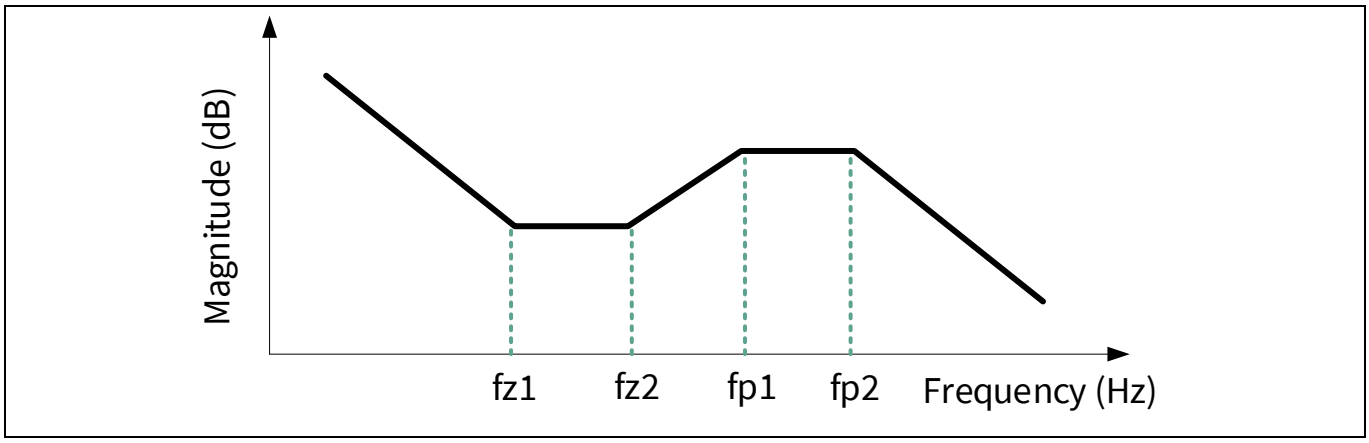

Figure 46 Typical Type III compensator gain

6.1.1 Pre-filter

The pre-filter is a single-pole LPF, implemented as shown in Figure 47, and it consists of:

- One input, the error voltage V_{errn}
- Two outputs, verrn_filt and verrn_slope

The verrn_filt is the low-pass filtered version of the computed error, V_{errn}, and it is used by the proportional and integral terms of the PID. The second output, verrn_slope, is the derivative of the filtered error and it is used by the derivative term of the PID.

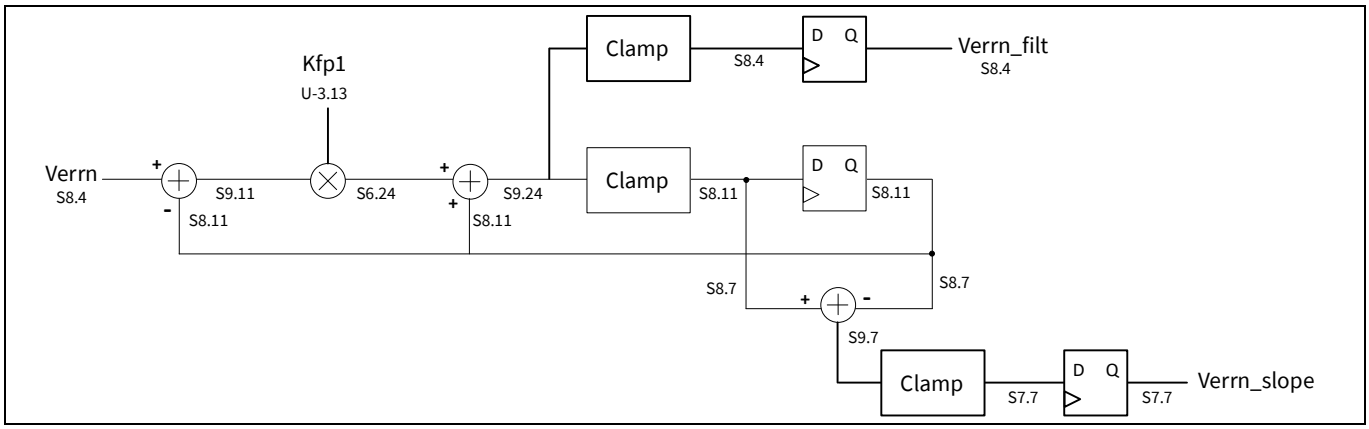

Figure 47 Pre-filter block diagram

The pre-filter creates the first high-frequency pole, f_{p1}, in the compensator transfer function. The pole location (i.e., LPF BW) is defined by the filter coefficient K_{FP1}, which is programmed via register **pid_kfp1_index_1ph**.

The filter coefficient parameters use an exponent mantissa format to provide an extended range utilizing fewer total bits. The upper three bits of kfp1_index represent the exponent, and the lower three bits represent the mantissa. The integer and real number representations of K_{FP1} are computed as shown in Equations (6.2) to (6.5).

$$k_{fp1_}exp = MIN\left(6, k_{fp1_}index[5:3]\right) \tag{6.2}$$

$$k_{fp1_}man = \left(k_{fp1_}index[5:3] > 6\right)?\,7 : k_{fp1_}index[2:0] \tag{6.3}$$

$$K_{FP1} = \left(8 + k_{fp1_}man\right) * 2^{k_{fp1_}exp} \tag{6.4}$$

$$K_{fp1_}real = K_{FP1} * 2^{-13} \tag{6.5}$$

The location of the first high-frequency pole can be computed as provided in Equation (6.6), using the previously computed real number representation for K_{FP1}.

$$fp1 = \left(\frac{1}{2\pi T_s}\right)\left(\frac{K_{fp1_}real}{1-K_{fp1_}real}\right), T_{SAMPLE} = \frac{1}{50MHz} \tag{6.6}$$

Note that **pid_kfp1_index_1ph** is clamped to 55 internally corresponding to a maximum exponent of 6. Table 29 shows the corresponding integer and real number representations of K_{FP1} for each kfp1_index value.

Table 29 kfp_index to K_{FP}, kfp_real

kfp_index	K_{FP}	kfp_real
0	8	0.0010
1	9	0.0011
2	10	0.0012
3	11	0.0013
4	12	0.0015
5	13	0.0016
6	14	0.0017
7	15	0.0018
8	16	0.0020
9	18	0.0022
10	20	0.0024
11	22	0.0027
12	24	0.0029
13	26	0.0032
14	28	0.0034
15	30	0.0037
16	32	0.0039
17	36	0.0044

18	40	0.0049
19	44	0.0054
20	48	0.0059
21	52	0.0063
22	56	0.0068
23	60	0.0073
24	64	0.0078
25	72	0.0088
26	80	0.0098
27	88	0.0107
28	96	0.0117
29	104	0.0127
30	112	0.0137
31	120	0.0146
32	128	0.0156
33	144	0.0176
34	160	0.0195
35	176	0.0215
36	192	0.0234
37	208	0.0254
38	224	0.0273
39	240	0.0293
40	256	0.0313
41	288	0.0352
42	320	0.0391
43	352	0.0430
44	384	0.0469
45	416	0.0508
46	448	0.0547
47	480	0.0586
48	512	0.0625
49	576	0.0703
50	640	0.0781
51	704	0.0859
52	768	0.0938
53	832	0.1016
54	896	0.1094
55	960	0.1172

Figure 48 shows the pre-filter BW as a function of the parameter kfp1_index. The post-filter uses the same index to coefficient mapping as the pre-filter, so the coefficient discussion in this section also applies to the post-filter (K_{FP2} and kfp2_index).

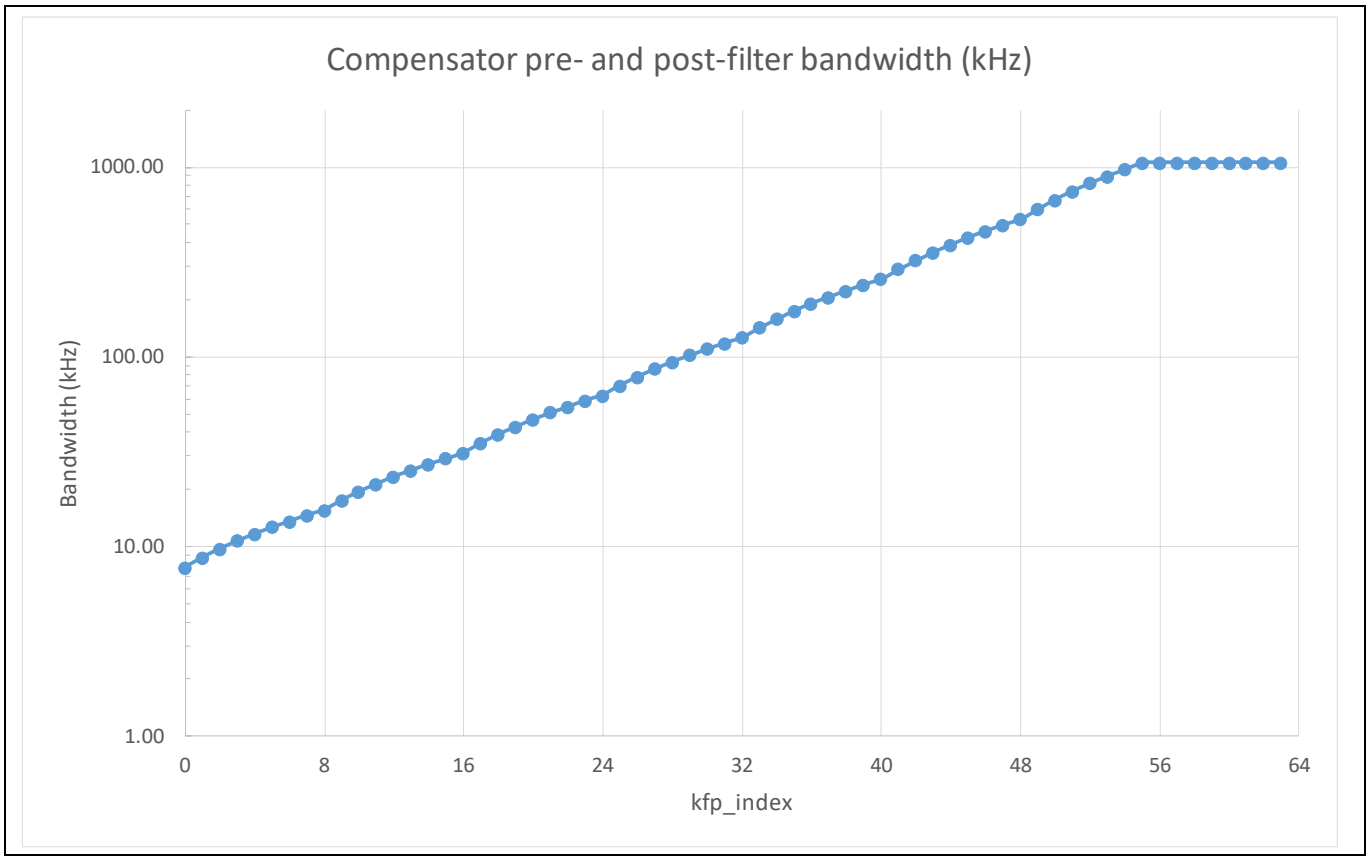

Figure 48 Pre- and post-filter BW as a function of kfp_index

6.1.2 PID term computation

The PID coefficients are computed based on the pre-filter output signals:

- Proportional (P_term) and integral (I_term) terms are obtained from the filtered error signal verrn_filt
- Derivative term (D_term) is computed based on verrn_slope, the derivative of the filtered error signal

The p_term and d_term are added to produce the pd_term, which is downstream-processed by the post-filter. The high-frequency gain of the integral term is negligible compared to the proportional and derivative terms. Therefore, no need exists for further low-pass filtering. The computation block diagram for proportional, integral, and derivative terms is shown in Figure 49.

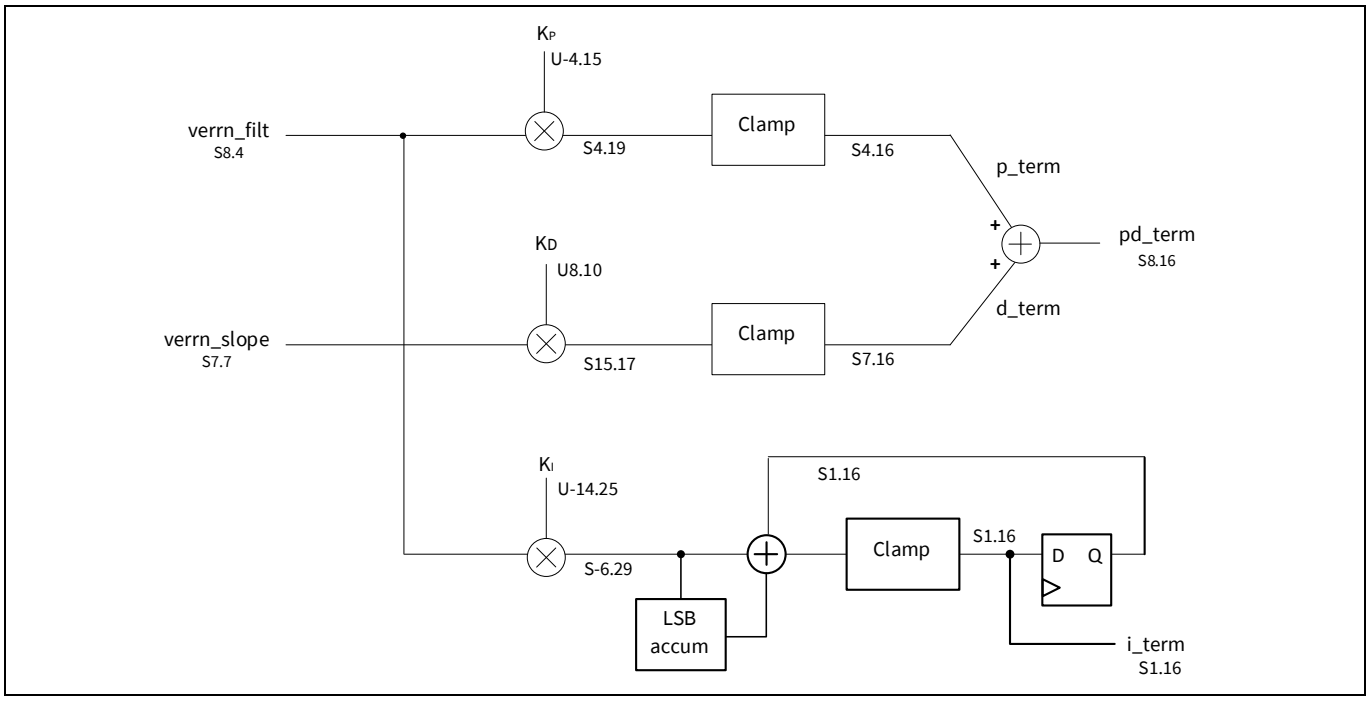

Figure 49 PID computation block diagram

Similarly to the pre-filter, the PID parameters use an exponent mantissa format to provide an extended range using fewer total bits:

- The upper three bits of K_P_index and K_I_index represent the exponent, and the lower three bits represent the mantissa
- For K_D_index, the upper four bits represent the exponent, and the lower three bits represent the mantissa

The integer and real number representations of K_P are computed as provided in Equations (6.7) to (6.10).

$$K_P_exp = K_P_index[5:3] \tag{6.7}$$

$$K_P_man = K_P_index[2:0] \tag{6.8}$$

$$K_P = (8 + K_P_man) * 2^{K_P_exp} \tag{6.9}$$

$$K_P_real = K_P * 2^{-15} \tag{6.10}$$

Table 30 provides the corresponding integer and real number representations of K_P for each K_P_index value.

Table 30 K_P_index to K_P, K_P_real

K_P_index	K_P	K_P_real
0	8	0.0002
1	9	0.0003
2	10	0.0003
3	11	0.0003
4	12	0.0004
5	13	0.0004

K_P_index	K_P	K_P_real
6	14	0.0004
7	15	0.0005
8	16	0.0005
9	18	0.0005
10	20	0.0006
11	22	0.0007
12	24	0.0007
13	26	0.0008
14	28	0.0009
15	30	0.0009
16	32	0.0010
17	36	0.0011
18	40	0.0012
19	44	0.0013
20	48	0.0015
21	52	0.0016
22	56	0.0017
23	60	0.0018
24	64	0.0020
25	72	0.0022
26	80	0.0024
27	88	0.0027
28	96	0.0029
29	104	0.0032
30	112	0.0034
31	120	0.0037
32	128	0.0039
33	144	0.0044
34	160	0.0049
35	176	0.0054
36	192	0.0059
37	208	0.0063
38	224	0.0068
39	240	0.0073
40	256	0.0078
41	288	0.0088
42	320	0.0098
43	352	0.0107
44	384	0.0117

K_P_index	K_P	K_P_real
45	416	0.0127
46	448	0.0137
47	480	0.0146
48	512	0.0156
49	576	0.0176
50	640	0.0195
51	704	0.0215
52	768	0.0234
53	832	0.0254
54	896	0.0273
55	960	0.0293
56	1024	0.0313
57	1152	0.0352
58	1280	0.0391
59	1408	0.0430
60	1536	0.0469
61	1664	0.0508
62	1792	0.0547
63	1920	0.0586

Correspondingly, the integer and real number representations of K_I are computed as given in Equations (6.11) to (6.14). Table 31 shows the corresponding integer and real number representations of K_I for each K_I_index value.

$$K_I_exp = K_I_index[5:3] \tag{6.11}$$

$$K_I_man = K_I_index[2:0] \tag{6.12}$$

$$K_I = (8 + K_I_man) * 2^{K_I_exp} \tag{6.13}$$

$$K_I_real = K_I * 2^{-25} \tag{6.14}$$

Table 31 K_I_index to K_I, K_I_real

K_I_index	K_I	K_I_real
0	8	2.38E-07
1	9	2.68E-07
2	10	2.98E-07
3	11	3.28E-07
4	12	3.58E-07
5	13	3.87E-07
6	14	4.17E-07

K_I_index	K_I	K_I_real
7	15	4.47E-07
8	16	4.77E-07
9	18	5.36E-07
10	20	5.96E-07
11	22	6.56E-07
12	24	7.15E-07
13	26	7.75E-07
14	28	8.34E-07
15	30	8.94E-07
16	32	9.54E-07
17	36	1.07E-06
18	40	1.19E-06
19	44	1.31E-06
20	48	1.43E-06
21	52	1.55E-06
22	56	1.67E-06
23	60	1.79E-06
24	64	1.91E-06
25	72	2.15E-06
26	80	2.38E-06
27	88	2.62E-06
28	96	2.86E-06
29	104	3.10E-06
30	112	3.34E-06
31	120	3.58E-06
32	128	3.81E-06
33	144	4.29E-06
34	160	4.77E-06
35	176	5.25E-06
36	192	5.72E-06
37	208	6.20E-06
38	224	6.68E-06
39	240	7.15E-06
40	256	7.63E-06
41	288	8.58E-06
42	320	9.54E-06
43	352	1.05E-05
44	384	1.14E-05
45	416	1.24E-05

K_I_index	K_I	K_I_real
46	448	1.34E-05
47	480	1.43E-05
48	512	1.53E-05
49	576	1.72E-05
50	640	1.91E-05
51	704	2.10E-05
52	768	2.29E-05
53	832	2.48E-05
54	896	2.67E-05
55	960	2.86E-05
56	1024	3.05E-05
57	1152	3.43E-05
58	1280	3.81E-05
59	1408	4.20E-05
60	1536	4.58E-05
61	1664	4.96E-05
62	1792	5.34E-05
63	1920	5.72E-05

The integer and real number representations of K_D are computed as provided in Equations (6.15) to (6.18). Table 32 shows the corresponding integer and real number representations of K_D for each kd_index value. Note that K_D_index is clamped to 119, internally corresponding to a maximum exponent of 14.

$$K_D_exp = MIN(14, K_D_index[6:3]) \quad (6.15)$$

$$K_D_man = (K_D_index[6:3] > 14)?\,7 : K_D_index[2:0] \quad (6.16)$$

$$K_D = (8 + K_D_man) * 2^{K_D_exp} \quad (6.17)$$

$$K_D_real = K_D * 2^{-10} \quad (6.18)$$

Table 32 K_D_index to K_D, K_D_real

K_D_index	K_D	K_D_real
0	8	0.0078
1	9	0.0088
2	10	0.0098
3	11	0.0107
4	12	0.0117
5	13	0.0127
6	14	0.0137
7	15	0.0146

K_D_index	K_D	K_D_real
8	16	0.0156
9	18	0.0176
10	20	0.0195
11	22	0.0215
12	24	0.0234
13	26	0.0254
14	28	0.0273
15	30	0.0293
16	32	0.0313
17	36	0.0352
18	40	0.0391
19	44	0.0430
20	48	0.0469
21	52	0.0508
22	56	0.0547
23	60	0.0586
24	64	0.0625
25	72	0.0703
26	80	0.0781
27	88	0.0859
28	96	0.0938
29	104	0.1016
30	112	0.1094
31	120	0.1172
32	128	0.1250
33	144	0.1406
34	160	0.1563
35	176	0.1719
36	192	0.1875
37	208	0.2031
38	224	0.2188
39	240	0.2344
40	256	0.2500
41	288	0.2813
42	320	0.3125
43	352	0.3438
44	384	0.3750
45	416	0.4063
46	448	0.4375

K_D_index	K_D	K_D_real
47	480	0.4688
48	512	0.5000
49	576	0.5625
50	640	0.6250
51	704	0.6875
52	768	0.7500
53	832	0.8125
54	896	0.8750
55	960	0.9375
56	1024	1.0000
57	1152	1.1250
58	1280	1.2500
59	1408	1.3750
60	1536	1.5000
61	1664	1.6250
62	1792	1.7500
63	1920	1.8750
64	2048	2.000
65	2304	2.250
66	2560	2.500
67	2816	2.750
68	3072	3.000
69	3328	3.250
70	3584	3.500
71	3840	3.750
72	4096	4.000
73	4608	4.500
74	5120	5.000
75	5632	5.500
76	6144	6.000
77	6656	6.500
78	7168	7.000
79	7680	7.500
80	8192	8.000
81	9216	9.000
82	10240	10.000
83	11264	11.000
84	12288	12.000
85	13312	13.000

K_D_index	K_D	K_D_real
86	14336	14.000
87	15360	15.000
88	16384	16.000
89	18432	18.000
90	20480	20.000
91	22528	22.000
92	24576	24.000
93	26624	26.000
94	28672	28.000
95	30720	30.000
96	32768	32.000
97	36864	36.000
98	40960	40.000
99	45056	44.000
100	49152	48.000
101	53248	52.000
102	57344	56.000
103	61440	60.000
104	65536	64.000
105	73728	72.000
106	81920	80.000
107	90112	88.000
108	98304	96.000
109	106496	104.000
110	114688	112.000
111	122880	120.000
112	131072	128.000
113	147456	144.000
114	163840	160.000
115	180224	176.000
116	196608	192.000
117	212992	208.000
118	229376	224.000
119	245760	240.000

The PID creates two mid-band zeros, f_{z1} and f_{z2}, in the compensator transfer function. Based on the real number representation of the PID coefficients, the location of the zeros can be obtained according to Equations (6.19) to (6.20), where T_{SAMPLE} equals $\frac{1}{50MHz}$.

$$f_{z1} = \left(\frac{1}{2\pi T_{SAMPLE}}\right)\left(K_P_real - \sqrt{K_P_real^2 - 4K_D_real * K_I_real}\right)/2K_D_real \quad (6.19)$$

$$f_{z2} = \left(\frac{1}{2\pi T_{SAMPLE}}\right)\left(K_P_real + \sqrt{K_P_real^2 - 4K_D_real * K_I_real}\right)/2K_D_real \quad (6.20)$$

6.1.3 Post-filter and summation

The post-filter is a single-pole low-pass filter, and it receives its input, pd_term, from the PID filter. The integral term has negligible gain compared to the proportional and derivative terms at high frequency, so no additional filtering is needed.

The compensator duty cycle output is created by adding the post-filtered output to:

- Integral term output, I_term
- Voltage FF term, ff_duty

The implementation of the post-filter is illustrated in Figure 50.

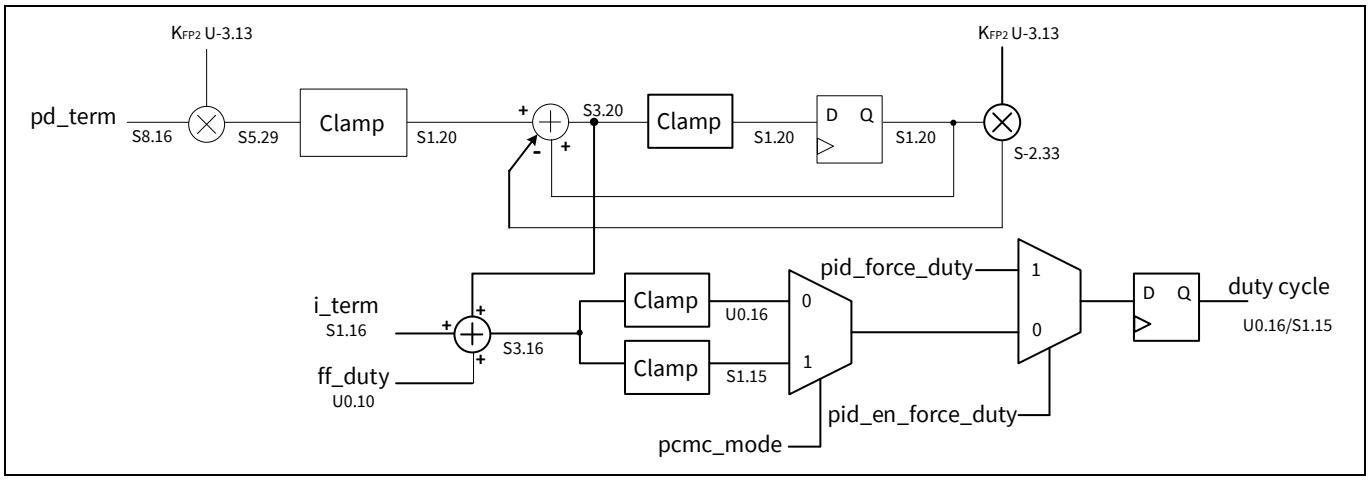

Figure 50 Post-filter block diagram

The post-filter creates the second high-frequency pole, f_{p2}, in the compensator transfer function. The pole location (i.e., LPF BW) is defined by the filter coefficient K_{FP2}, which is programmable through register **pid_kfp2_index_1ph**.

Similarly to the pre-filter, K_{FP2}_index uses exponent mantissa format to provide an extended range using fewer total bits. The upper three bits of K_{FP2}_index represent the exponent and the lower three bits represent the mantissa. The integer and real number representations of K_{FP2} are computed as given in Equations (6.21) to (6.24). Note that K_{FP2}_index is clamped to 55, internally corresponding to a maximum exponent of 6.

$$K_{FP2}_exp = MIN(6, K_{FP2}_index[5:3]) \quad (6.21)$$

$$K_{FP2}_man = (K_{FP2}_index[5:3] > 6)?\,7 : K_{FP2}_index[2:0] \quad (6.22)$$

$$K_{FP2} = (8 + K_{FP2}_man) * 2^{K_{FP2}_exp} \quad (6.23)$$

$$K_{FP2}_real = K_{FP2} * 2^{-13} \quad (6.24)$$

The location of the second high-frequency pole can be computed as provided in Equation (6.25), using the previously computed real number representation for K_{FP2}.

$$fp2 = \left(\frac{1}{2\pi T_{SAMPLE}}\right)\left(\frac{K_{FP2_real}}{1-K_{FP2_real}}\right), T_{SAMPLE} = 50\ MHz \quad (6.25)$$

The corresponding integer and real number representations of K_{FP2} for each K_{FP2}_index value are the same as for K_{FP1} provided in Table 29 in the pre-filter Section 6.1.1. Correspondingly, the post-filter BW as a function of parameter K_{FP2}_index is the same as shown in Figure 48.

6.1.4 Input/output clamping of the compensation filter

The input and the output of the PID filter can be clamped. The input clamp is user-programmable through register **vsp_verrn_clamp_thresh**, as discussed in Section 2.3.1, and it limits the maximum error seen by the compensation filter.

The compensation filter output, duty_cycle, is clamped to a fixed 16-bit width. In the case of VMC it refers a value between 0.0 and 1.0. The PCMC behavior and the corresponding compensator output value is discussed in Section 6.3.

6.1.5 Output override – forced duty cycle

The PID output of the compensation filter can be overridden. The user can force the duty-cycle value by selecting:

- Register **pid_force_duty_en**
- Then setting a value in register **pid_force_duty**

This functionality is illustrated in the post-filter block diagram in Figure 50.

Because this override is applied at the PID output, downstream adjustments to the duty cycle are still applied. These adjustments include:

- Current balance in an interleaved (multiphase) design
- Flux balance in a FB design

However, if the downstream adjustments are required to be overridden, a separate pair of override registers is provided:

- **ramp0_force_duty**
- **ramp1_force_duty**

These functions are discussed in more detail in Section 7.3.4.

6.1.6 Coefficient scaling

The PID coefficients are scaled with V_{RECT} to maintain constant loop gain despite the input voltage variations. The user can define a reference V_{RECT} voltage through register **pid_vrect_ref**, at which the gain scale is 1.0. This coefficient scale factor is defined as given in Equation (6.26).

$$Coefficient\ scale\ factor = \frac{V_{RECT}}{pid_vrect_ref} \quad (6.26)$$

The register **pid_vrect_ref** value should be set to the expected nominal V_{RECT} voltage prior to the PID coefficient optimization.

An example for selecting a proper **pid_vrect_ref** value is given below for a FB topology with nominal input voltage of 48 V and transformer turns ratio of 3. The nominal rectified voltage is computed in Equation (6.27) and the resulting value should be set via register **pid_vrect_ref**.

$$V_{RECT-nom} = \frac{V_{IN-nom}}{N_{turns}} = \frac{48\ V}{3} = 16\ V \quad (6.27)$$

Subsequently, the optimized PID coefficients can be observed from registers:

- **pid_kp_eff**, for K_P
- **pid_ki_eff**, for K_I
- **pid_kd_eff**, for K_D

6.1.7 Freeze, reset accumulator

Undesired integrator “windup” could occur in the following operating conditions:

- Burst mode (BM) operation, when PID is not controlling the output duty cycle
- Duty cycle output is 0 and a negative V_{errn} input is received
- PCL has been exceeded and a positive V_{errn} is received
- PW exceeds the maximum PW (duty cycle exceeds the maximum duty cycle) and a positive V_{errn} is received

Under these conditions, the XDPP1100 HW freezes the integrator term accumulator.

In addition to the HW freeze conditions, it is possible to freeze the integrator accumulator through register **pid_freeze_accum**. Another register, **pid_reset_accum**, allows the FW to reset the integrator accumulator to 0.

6.2 Input voltage feed-forward

This section describes the input voltage FF term computation and the user-programmable parameters and relevant registers. The objective of input voltage FF is to minimize the V_{IN} variation impact on the overall controller performance. The FF is only applied in VMC. PCMC provides inherent input voltage disturbance rejection, and therefore in this mode the FF is automatically set to zero.

Input voltage FF is part of the compensator, and it is summed with the PID output to create the PWM duty cycle target, as was shown in Figure 45. The FF term is obtained based on:

- Target output voltage, $V_{control}$
- Selected input voltage measurement source

The HW implementation for the FF computation is shown in Figure 51. The computation of the target output voltage as well as the selection of the input voltage measurement source are discussed in the following subsections.

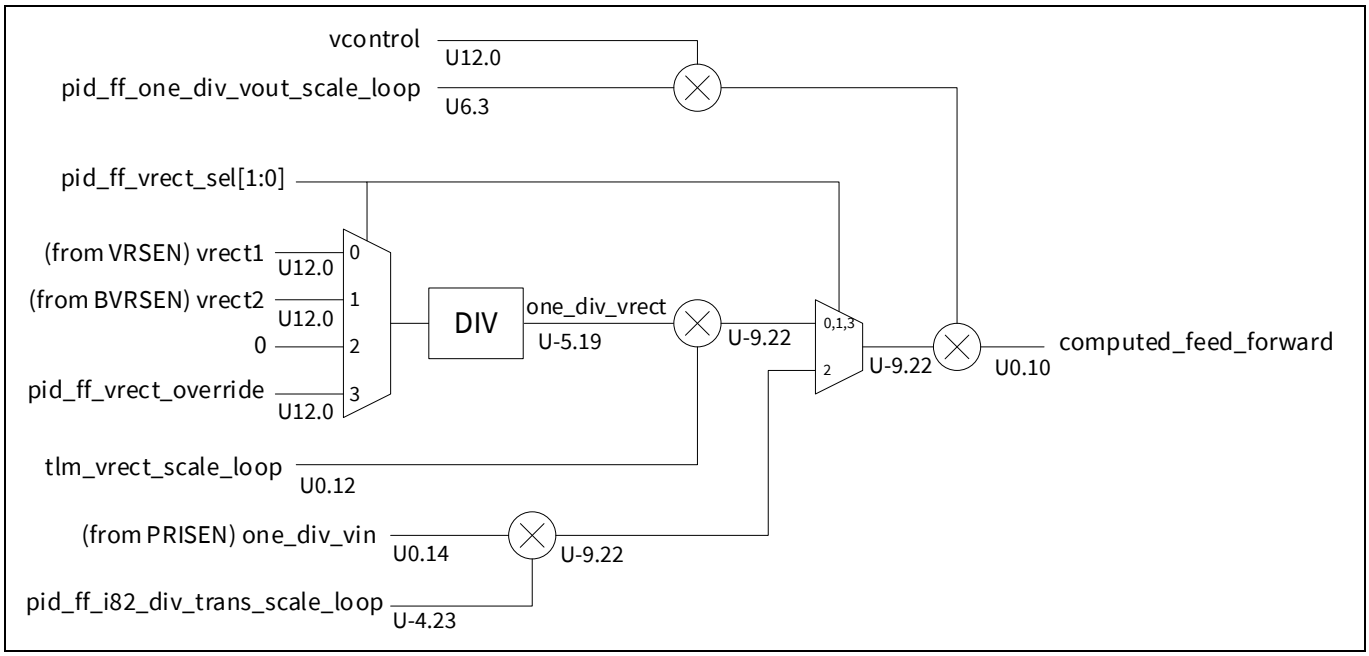

Figure 51 FF computation diagram

6.2.1 Output voltage target computation

The target output voltage is required for the FF computation. In order to use the representation of the true V_{OUT}, the internal reference voltage, $V_{control}$, needs to be scaled. The target output voltage can be computed as given in Equation (6.28).

$$V_{OUT,target} = \frac{v_{control}}{VOUT_SCALE_LOOP}, \quad (6.28)$$

Where:

- $V_{control}$ is the internal reference voltage
- VOUT_SCALE_LOOP is a PMBus command for external resistive divider

In the actual HW implementation, as shown in Figure 51, the scaling is obtained through register **pid_ff_one_div_vout_scale_loop**. This parameter represents (1 / VOUT_SCALE_LOOP) and is automatically computed by FW on initial configuration download and on any subsequent programming on VOUT_SCALE_LOOP.

6.2.2 Input voltage source select and computation

The input voltage for the FF computation is selected through register **pid_ff_vrect_sel** from the following sources:

- VRSEN (V_{RECT1})
- BVSEN_BVRSEN (V_{RECT2})
- PRISEN (TS V_{IN})
- **Pid_ff_vrect_override**

The rectified voltage V_{RECT1} is the sensed voltage of the first loop at the VRSEN pin on the secondary side. In the case of single-loop interleaved topology, the BVSEN_BVRSEN pin senses the phase two rectified voltage V_{RECT2}. These sensed voltages are obtained directly from the high-speed VSADC, which offers the best FF performance.

The TS PRISEN is one of the general-purpose ADC input channels, and it can be configured to sense the input voltage on the primary side. PRISEN is obtained from the low-speed TSADC, and thereafter low-pass filtered in telemetry. In this case, the FF performance is affected by the LPF BW, which can be programmed through register **tlm_kfp_vin**.

The FF computation can be overridden with FW, using the register **pid_ff_vrect_override**. This option can be used with:

- General-purpose ADCs
- Non-linear or non-positive slope LUTs

The V_{RECT} override for FF computation uses the same data format as VRSEN/BVSEN_BVRSEN, as the computation is performed in internal VSADC format.

Depending on whether the input voltage is sensed on the primary or secondary side, certain voltage scaling is needed. The required scalings are defined as follows:

- Secondary side: V_{RECT} is sensed through an external resistor divider, and therefore the input sources VRSEN, BVSEN_BVRSEN and **pid_ff_vrect_override** need to be scaled. The register **tlm_vrect_scale_loop** defines the V_{RECT} resistor divider and it can be configured through PMBus command MFR_VRECT_SCALE
- Primary side: The sensed input voltage needs to be scaled by the transformer turns ratio. This is performed through register **pid_ff_i82_div_trans_scale_loop**. This value is computed automatically by the FW based on the PMBus command MFR_TRANSFORMER_SCALE

An example of a special case for the FF calculation is when the ACF converter (Loop 0) is followed by post-buck topology (Loop 1). Thus, the Loop 1 input 1 is the output of Loop 0, and the following settings should be applied:

- Input source for Loop 1 should be selected through **pid1_ff_vrect_sel** to PRISEN (TS V_{IN})
- Input voltage telemetry source should be set to 2 (Loop 0 V_{OUT}) through register **tlm_vin_src_sel**
- Filtering by Loop 0 output voltage telemetry LPF (register **tlm0_kfp_vout**) as well as Loop 1 input voltage telemetry LPF (register **tlm1_kfp_vin**)

6.2.3 Override and adjustment options for FF

For the FF term there is a possibility of overriding the HW-computed FF with FW and/or perform other adjustments depending on the application needs. These features are discussed below in more detail, and their implementation is shown in Figure 52.

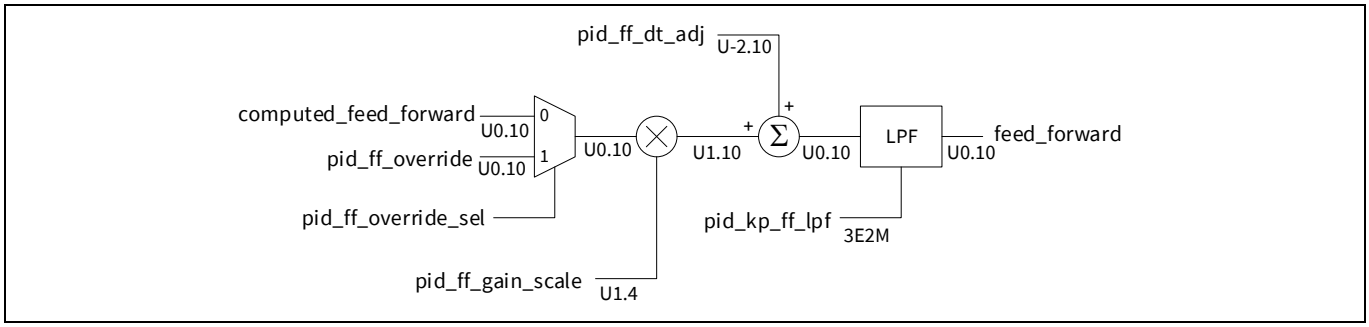

Figure 52 FF override and adjustment diagram

The option of overriding the computed FF term can be set via:

- Register **pid_ff_override_sel** (select the override option)
- Register **pid_ff_override** (set a suitable FF value)

This option is suitable for boost or buck-boost derived topologies, and it can be also used to disable the FF functionality by setting the **pid_ff_override** to 0.

Further FF adjustments include the following:

- Scalable FF gain via register **pid_ff_gain_scale** decreases or increases the FF effect depending on the gain value. Typically, the gain is set to 1.0
- Adjustable offset for the FF term compensates for the dead time error through register **pid_ff_dt_adj**. As an example, if the dead time is 1 percent of the period, this register needs to be set to 1 percent. If the value is set to 0, the PID integrator will compensate

In addition to these adjustments, optional low-pass filtering can be applied to the computed FF term before being used by the PID compensator for duty calculation. The BW of this filter can be set via register **pid_kp_ff_lpf**. High BW results in fastest FF response, whereas low BW might smooth the noise at the sensed signal but will slow the FF response. The **pid_kp_ff_lpf** uses exponential format, where the upper three bits represent the exponent and the lower two bits represent the mantissa, as shown in Equations (6.29) to (6.32).

$$K_{FP}_exp = pid_kp_ff_lpf[4:2] \tag{6.29}$$

$$K_{FP}_man = 4 + pid_kp_ff_kpf[1:0] \tag{6.30}$$

$$K_{FP} = K_{FP}_man * 2^{K_{FP}_exp} * 2^{-9} \tag{6.31}$$

$$F3dB(MHz) = \left[\frac{K_{FP}}{1-K_{FP}}\right] * \frac{50\ MHz}{2\pi} \tag{6.32}$$

Table 33 K_P_ff index to BW

kfp_index	K_{FP}	kfp_real	F3dB (kHz)
0	4	0.0078	62.7
1	5	0.0098	78.5
2	6	0.0117	94.4
3	7	0.0137	110.3
4	8	0.0156	126.3
5	10	0.0195	158.5
6	12	0.0234	191.0
7	14	0.0273	223.7
8	16	0.0313	256.7
9	20	0.0391	323.5
10	24	0.0469	391.4
11	28	0.0547	460.4
12	32	0.0625	530.5
13	40	0.0781	674.4

kfp_index	K_{FP}	kfp_real	F3dB (kHz)
14	48	0.0938	823.2
15	56	0.1094	977.3
16	64	0.1250	1136.8
17	80	0.1563	1473.7
18	96	0.1875	1836.4
19	112	0.2188	2228.2
20	128	0.2500	2652.6
21	160	0.3125	3617.2
22	192	0.3750	4774.6
23	224	0.4375	6189.4
24	256	0.5000	7957.7
25	320	0.6250	13262.9
26	384	0.7500	23873.2
27	448	0.8750	55704.2
28	448	0.8750	55704.2
29	448	0.8750	55704.2
30	448	0.8750	55704.2
31	N/A	N/A	Bypass

6.3 Control mode selection – peak current mode

This section describes the selection of control mode in more detail, focusing on the compensator-related register settings for PCMC. However, most PCMC-related circuitry and detailed functionality is discussed in the PWM chapter that follows.

The control mode for the compensator can be selected via registers **mode_control_loop0**, **mode_control_loop1**, for Loop 0 and Loop 1 respectively. The control mode options are:

- VMC
- PCMC on secondary
- PCMC on primary

The difference between these control methods is that for VMC, the compensator output is directly the duty cycle, whereas for the PCMC the compensator output is a reference current that is compared to the sensed current. Therefore, depending on the desired control method, the compensator output format varies:

- In VMC the compensator output is clamped to the unsigned range 0.0 to 1.0
- In PCMC the compensator output is “normalized reference current” where the normalization is within the IADC maximum range, leading to a signed output range -1.0 to +1.0

The inherent nature of current mode control provides simple dynamics, and therefore, typically for PCMC a Type II compensator is sufficient. The difference between Type II and III compensators is that Type II consists of PI and a single pole, whereas Type III is PID with two poles. The compensator in Figure 45 provides a Type III response, and in order to obtain the Type II response for PCMC the following need to be considered:

- The compensator includes two single-pole LPFs but neither of them is possible to bypass. However, by setting one of the filter coefficients via register **pid_kfp1_index_1ph** or **pid_kfp2_index_1ph** to maximum BW, the Type II response can be approximated
- Due to the exponential nature of the PID coefficients, setting register **pid_kd_index_1ph** to zero does not result in a zero-valued K_D. However, when PCMC mode is selected (register **mode_control_loop0/1)** it is possible to override the exponential setting and force K_D to be zero via register **pid_kd_index_1ph**. If a non-zero K_D is desired in PCMC, it can be set via **pid_kd_index_1ph** to a non-zero setting

The FF is automatically set to 0 when a loop is configured for PCMC mode and input voltage-based PID coefficient adjustment is automatically disabled.

6.4 Open sense fault detection

The feedback loop is broken if the external resistor divider is not properly placed. The compensator is capable of detecting a missing sense resistor within the external resistor divider, between the output voltage and VSEN (BVSEN) (Figure 2). The fault detection is based on searching for duty-cycle value which is larger than the FF term. The following settings can be defined for the fault detection:

- Register **pid_osp_duty_thr** defines the minimum duty cycle (i.e., compensator output) at which to begin looking for a fault
- Register **pid_osp_ff_thr** defines the minimum PID FF value at which to begin looking for a fault
- Register **pid_osp_ff_scale** defines how much greater the duty must be than the FF to declare a fault

Equation (6.33) shows the condition for a fault declaration.

$$duty > pid_osp_ff_scale * FF \tag{6.33}$$

Both the duty cycle and FF must be above their corresponding thresholds, **pid_osp_duty_thr** and **pid_osp_ff_thr**, in order to prevent false fault detection at low voltages during start-up, when duty cycle is much larger than the FF (e.g., at V_{OUT} = 0 V). The region where the fault is declared is emphasized in blue in Figure 53. This OSP fault region is formed assuming the minimum duty cycle limit of 0.3 and the minimum FF threshold of 0.05 and a scaling factor of 3.0.

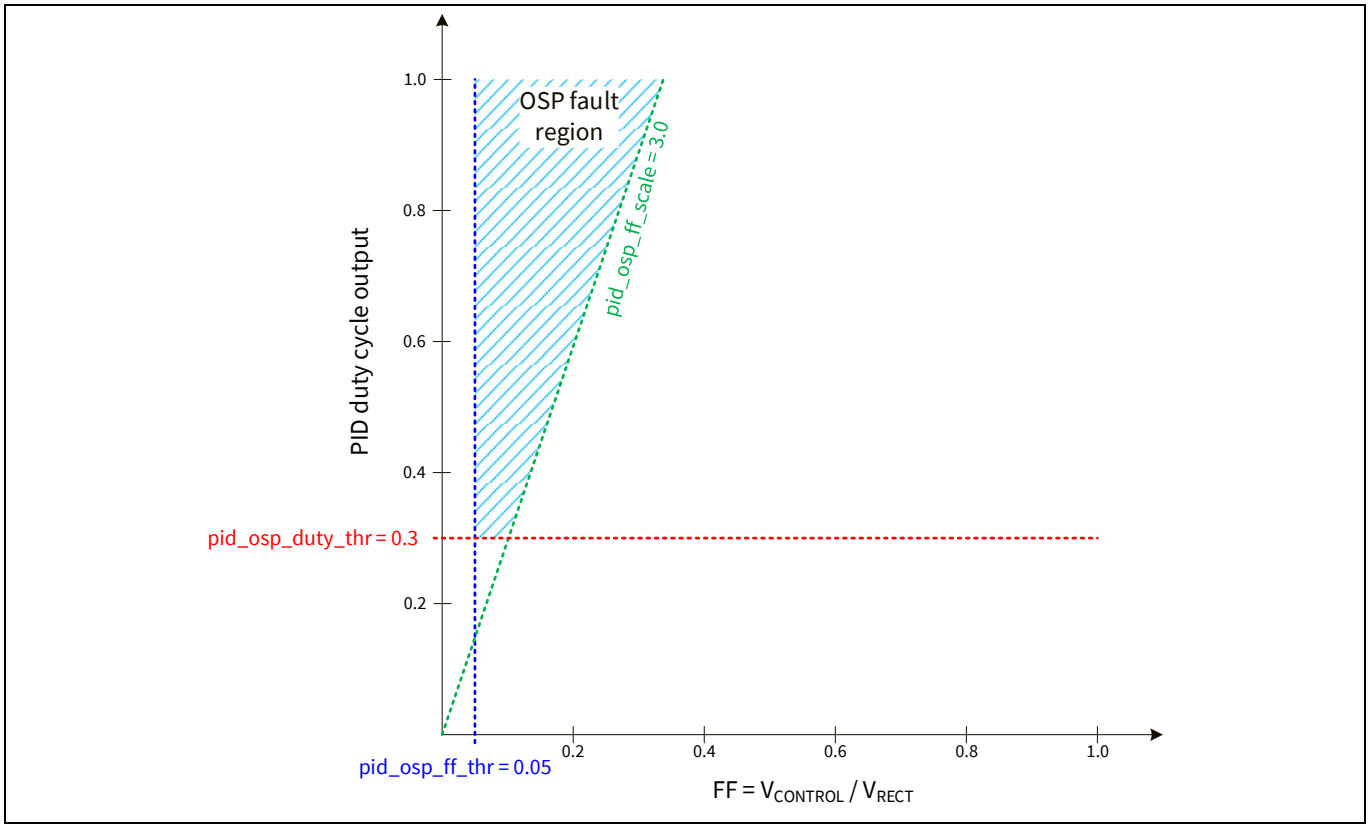

Figure 53 PID-based open VS detection

6.5 Compensation filter registers

The relevant compensator-related registers and their descriptions are provided in Table 34.

Table 34 Compensator-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
vsen	vsp_verrn_clamp_thresh	RW	7000_0800h (VSEN) 7000_1000h (BVSEN)	[2:0]	V_{errn} clamp threshold at PID input where V_{errn} is defined as the control voltage minus the sensed V_{OUT}. Ignoring droop, V_{errn} = VOUT_COMMAND*VOUT_SCALE_LOOP - V(VSEN) 0: Clamp range = -40 to 40 mV 1: Clamp range = -80 to 80 mV 2: Clamp range = -120 to 120 mV 3: Clamp range = -160 to 160 mV 4: Clamp range = -200 to 200 mV 5: Clamp range = -240 to 240 mV

Peripheral	Field name	Access	Address	Bits	Description
					6: Clamp range = -280 to 280 mV 7: Clamp range = -318.75 to 318.75 mV
pid	pid_ff_dt_adj	RW	7000_1C00h (pid0) 7000_2000h (pid1)	[7:0]	PID FF adjustment for dead time. Note that adjustment is always positive because dead time reduces duty cycle. Computed for: **Bridge topology:** pid_ff_dt_adj = T_{dead} / (T_{switch} / 2) **Non-bridge topology:** pid_ff_dt_adj = T_{dead} / T_{switch} **Example:** Bridge topology, T_{dead} = 100 ns, T_{switch} = 4 µs pid_ff_dt_adj = 100 ns / 2000 ns = 0.05 LSB = 2^-10, range = 0 to 0.2490
pid	pid_kp_ff_lpf	RW	7000_1C00h (pid0) 7000_2000h (pid1)	[12:8]	PID FF LPF coefficient. The computed FF term is passed through a LPF before being used by the PID as part of the duty-cycle calculation. This parameter sets the BW of this filter. A setting of 31 bypasses the filter. kfp_exp = pid_kp_ff_kpf[4:2] kfp_man = 4 + pid_kp_ff_kpf[1:0] kfp = kfp_man * 2^kfp_exp * 2^-9 F3db(MHz) = [kfp / (1 - kfp)] * 50 MHz / 2*pi
pid	pid_ff_gain_scale	RW	7000_1C00h (pid0) 7000_2000h (pid1)	[17:13]	FF gain scale. This parameter may be used to apply a gain to the computed FF term. The typical setting is 16, which corresponds to a gain of 1.0. LSB = 0.0625, range = 0.0 to 1.9375
pid	pid_ff_vrect_sel	RW	7000_1C00h (pid0) 7000_2000h (pid1)	[19:18]	FF V_{RECT} source select. FF is computed as (V_{OUT} / V_{RECT}) for isolated and as (V_{OUT} / V_{IN}) for non-isolated buck-derived topologies. Note that the FF computation may be overriden entirely using parameters pid_ff_override_sel and pid_ff_override (e.g., to override the HW-computed FF with a FW computation appropriate for boost or buck-boost derived topologies). 0: VS1 (VRSEN) V_{RECT}

Peripheral	Field name	Access	Address	Bits	Description
					1: VS2 (BVSEN_BVRSEN) V_{RECT} 2: TS V_{IN} (e.g., PRISEN) 3: pid_ff_vrect_override
pid	pid_vrect_ref	RW		[27:20]	PID coefficient scaling reference voltage. PID coefficients are scaled with V_{RECT} to maintain a constant loop gain. This parameter defines the reference V_{RECT} voltage at which the gain scale is 1.0. This parameter should be set to the expected nominal V_{RECT} voltage, and it should be set before optimization of PID coefficients K_p, K_i and K_D. **Example:** V_{IN_nom} = 48 V, FB topology, N_{turn} = 3 pid_vrect_ref = V_{RECT_nom} = 48 V / 3 = 16 V LSB = 0.32 V, range = 0.0 to 81.6 V
pid	pid_kfp1_index_1ph	RW	7000_1C04h (pid0) 7000_2004h (pid1)	[5:0]	PID pre-filter coefficient index. Note that index settings greater than 55 are clamped to 55. kfp_exp = pid_kfp1_index_1ph[5:3] kfp_man = 8 + pid_kfp1_index_1ph[2:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(MHz) = [kfp / (1 - kfp)] * 50 MHz / 2*pi
pid	pid_kfp2_index_1ph	RW	7000_1C04h (pid0) 7000_2004h (pid1)	[11:6]	PID post-filter coefficient index. Note that index settings greater than 55 are clamped to 55. kfp_exp = pid_kfp2_index_1ph[5:3] kfp_man = 8 + pid_kfp2_index_1ph[2:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(MHz) = [kfp / (1 - kfp)] * 50 MHz / 2*pi
pid	pid_kp_index_1ph	RW	7000_1C04h (pid0) 7000_2004h (pid1)	[17:12]	PID proportional coefficient index. kp_exp = pid_kp_index_1ph[5:3] kp_man = 8 + pid_kp_index_1ph[2:0] kp = kp_man * 2^kp_exp * 2^-15
pid	pid_ki_index_1ph	RW	7000_1C04h (pid0) 7000_2004h (pid1)	[23:18]	PID integral coefficient index. ki_exp = pid_ki_index_1ph[5:3] ki_man = 8 + pid_ki_index_1ph[2:0] ki = ki_man * 2^ki_exp * 2^-25
pid	pid_kd_index_1ph	RW	7000_1C04h (pid0)	[30:24]	PID derivative coefficient index. Note that index settings greater

Peripheral	Field name	Access	Address	Bits	Description
			7000_2004h (pid1)		than 119 are clamped to 119. kd_exp = pid_kd_index_1ph[6:3] kd_man = 8 + pid_kd_index_1ph[2:0] kd = kd_man * 2^kd_exp * 2^-10
pid	pid_force_duty	RW	7000_1C08h (pid0) 7000_2008h (pid1)	[7:0]	Forced duty-cycle value overrides the PID output when selected by pid_force_duty_en. Since this force is applied at the PID output, downstream adjustments to the duty cycle such as current balance in an interleaved (multiphase) design are still applied. To also override the current balance adjust use pwm.ramp0_force_duty or pwm.ramp1_force_duty. LSB = 2^-8, range = 0.0 to 0.9961
pid	pid_force_duty_en	RW	7000_1C08h (pid0) 7000_2008h (pid1)	[8]	Forced duty-cycle select. 0: Use PID-computed duty cycle 1: Use pid_force_duty
pid	pid_osp_ff_thr	RW	7000_1C18h (pid0) 7000_2018h (pid1)	[6:0]	Defines the minimum PID FF value, above which to begin checking for an open-sense fault during soft-start. This fault will indicate a larger-than-expected impedance between V_{OUT} and V_{OUT_sen}. LSB = 2^-7, range = 0.0 to 0.9921875
pid	pid_osp_duty_thr	RW	7000_1C18h (pid0) 7000_2018h (pid1)	[13:7]	Defines the minimum PID duty cycle value, above which to begin checking for an open-sense fault during soft-start. This fault will indicate a larger-than-expected impedance between V_{OUT} and V_{OUT_sen}. LSB = 2^-7, range = 0.0 to 0.9921875
pid	pid_osp_ff_scale	RW	7000_1C18h (pid0) 7000_2018h (pid1)	[17:14]	Scale factor applied to PID FF term to detect an open-sense fault between V_{OUT} and V_{OUT_sen} during soft-start. A setting of 0 will disable this fault check. LSB = 0.5, range = 0.0 to 7.5
pid	pid_ff_vrect_override	RW	7000_1C1Ch (pid0) 7000_201Ch (pid1)	[11:0]	V_{RECT} override for PID FF computation in internal VSADC format (i.e., as sensed at VRSEN, BVRSEN). Used only when selected by pid_ff_vrect_sel. If computing

Peripheral	Field name	Access	Address	Bits	Description
					from V_{IN}, pid_ff_vrect_override(U12.0) = V_{IN}(V) * 800 * MFR_VRECT_SCALE(U0.12) * MFR_TRANSFORMER_SCALE(U0.12) / (2^12 * 2^12) LSB = 1.25 mV, range = 0.0 to 5.11875 V
pid	pid_ff_override	RW	7000_1C20h (pid0) 7000_2020h (pid1)	[9:0]	PID FF override value. Used when selected by pid_ff_override_sel. This parameter along with pid_ff_override_sel may be used to override the HW-computed FF with a FW computation appropriate for boost or buck-boost derived topologies. LSB = 2^-10, range = 0.0 to 0.9990
pid	pid_ff_override_sel	RW	7000_1C20h (pid0) 7000_2020h (pid1)	[10]	PID FF override select. 0: Use computed FF with V_{RECT} selected by pid_ff_vrect_sel 1: Use pid_ff_override
pid	pid_ff_one_div_vout_scale_loop	RW	7000_1C24h (pid0) 7000_2024h (pid1)	[8:0]	Conversion factor for computing V_{OUT} from the measured VSADC output. This parameter is computed automatically by the default FW and requires no action by the user. **Computed by FW from PMBus commands as follows:** pid_ff_one_div_vout_scale_loop(U6.3) = 2^19 / VOUT_SCALE_LOOP(U0.16), clamp to 2^9-1 LSB = 0.125, range = 0.0 to 63.875
pid	pid_ff_i82_div_trans_scale_loop	RW	7000_1C24h (pid0) 7000_2024h (pid1)	[27:9]	Conversion factor for computing 1/V_{RECT} from the TSADC V_{IN} measurement. This parameter is computed automatically by the default FW and requires no action by the user. **Computed by FW from PMBus commands as follows:** exp = $signed(MFR_TRANSFORMER_SCALE[15:11]), exp = -10, -11, -12 supported quot1(U5.7) = 2^(-3-exp) / MFR_TRANSFORMER_SCALE[9:0]

Peripheral	Field name	Access	Address	Bits	Description
					pid_ff_i82_div_trans_scale_loop(U-4.23) = quot1(U5.7) * 82 (U-9.16) LSB = 2^-23, range = 0.0 to 0.0625
pid	pid_kp_eff	R	7000_1C28h (pid0) 7000_2028h (pid1)	[5:0]	PID K_p coefficient value after V_{RECT} scaling. See pid_kp_index_1ph description for definition
Pid	pid_ki_eff	R	7000_1C28h (pid0) 7000_2028h (pid1)	[11:6]	PID K_i coefficient value after V_{RECT} scaling. See pid_ki_index_1ph description for definition
Pid	pid_kd_eff	R	7000_1C28h (pid0) 7000_2028h (pid1)	[18:12]	PID K_D coefficient value after V_{RECT} scaling. See pid_kd_index_1ph description for definition
Pid	pid_reset_accum	RW	7000_1C2Ch (pid0) 7000_202Ch (pid1)	[0]	PID integral term accumulator reset. Allows FW to reset the accumulator by first writing a 1 to reset the accumulator followed by writing a 0 to release the reset. 0: Accumulator operating normally 1: Accumulator reset to 0
pid	pid_freeze_accum	RW	7000_1C30h (pid0) 7000_2030h (pid1)	[0]	PID integral term accumulator freeze control. Allows FW to hold the accumulator at its current value by writing a 1 until it is released by writing a 0. 0: Accumulator operating normally 1: Accumulator value frozen
pid	pid_duty	R	7000_1C40h (pid0) 7000_2040h (pid1)	[15:0]	Unfiltered PID duty cycle output. Note that this is not the register field used for the PMBus command READ_DUTY_CYCLE. The PMBus command uses telem.tlm_duty_fsw. LSB = 2^-16, range = 0.0 to 0.999985
pid	pid_ff_duty_lpf	R	7000_1C44h (pid0) 7000_2044h (pid1)	[15:0]	PID FF computation after gain/offset adjust and LPF. LSB = 2^-16, range = 0.0 to 0.999985
pwm	mode_control_loop0	RW	7000_2C00	[13:12]	Loop 0 control mode select 0: VMC 1: PCMC on secondary 2: PCMC on primary 3: Reserved

Peripheral	Field name	Access	Address	Bits	Description
pwm	mode_control_loop1	RW	7000_2C00	[15:14]	Loop 1 control mode select 0: VMC 1: PCMC on secondary 2: PCMC on primary 3: Reserved
pwm	ramp0_force_duty	RW	7000_2C34	[7:0]	Forced duty-cycle value overrides ramp0 input when selected by ramp0_force_duty_en. Because this force is applied at the ramp input, upstream adjustments to the duty cycle such as current balance in an interleaved (multiphase) design are overwritten. To not override the current balance, adjust use pid0.pid_force_duty or pid1.pid_force_duty, which apply the force prior to downstream adjustments. LSB = 0.3906 percent, range = 0.0 to 99.6094 percent
pwm	ramp0_force_duty_en	RW	7000_2C34	[8]	PWM ramp0 forced duty-cycle select. 0: Use PID computed duty cycle 1: Use ramp0_force_duty
pwm	ramp1_force_duty	RW	7000_2C48	[7:0]	Forced duty-cycle value overrides ramp1 input when selected by ramp1_force_duty_en. Because this force is applied at the ramp input, upstream adjustments to the duty cycle such as current balance in an interleaved (multiphase) design are overwritten. To not override the current balance, adjust use pid0.pid_force_duty or pid1.pid_force_duty which apply the force prior to downstream adjustments. LSB = 0.3906 percent, range = 0.0 to 99.6094 percent
pwm	ramp1_force_duty_en	RW	7000_2C48	[8]	PWM ramp1 forced duty cycle select. 0: Use PID-computed duty cycle 1: Use ramp1_force_duty

7 Digital pulse width modulator

This chapter discusses the digital pulse width modulator (PWM) and its implementation in more detail. Applicable feedback control modes and BM, as well as the fast-transient operation, are described and the relevant register settings are provided. At the end of the chapter, all relevant PWM-related registers and PMBus commands are given.

The PWM converts the compensation filter output into one or more PWM pulses depending on the application. The XDPP1100 has up to 12 PWM output pins. The PWM consists of:

- Two ramp generators
- 12 pulse generators
- 12 interpolators

A simplified block diagram of the PWM is shown in Figure 54. Based on the compensation filter output, the ramp generators produce timing signals for pulse generators to create coarse resolution pulse outputs. These coarse pulses are fed to the interpolators, where fine resolution timing information is used to create output pulses with 78.125 ps resolution. The following sections discuss the PWM features and functions in more detail.

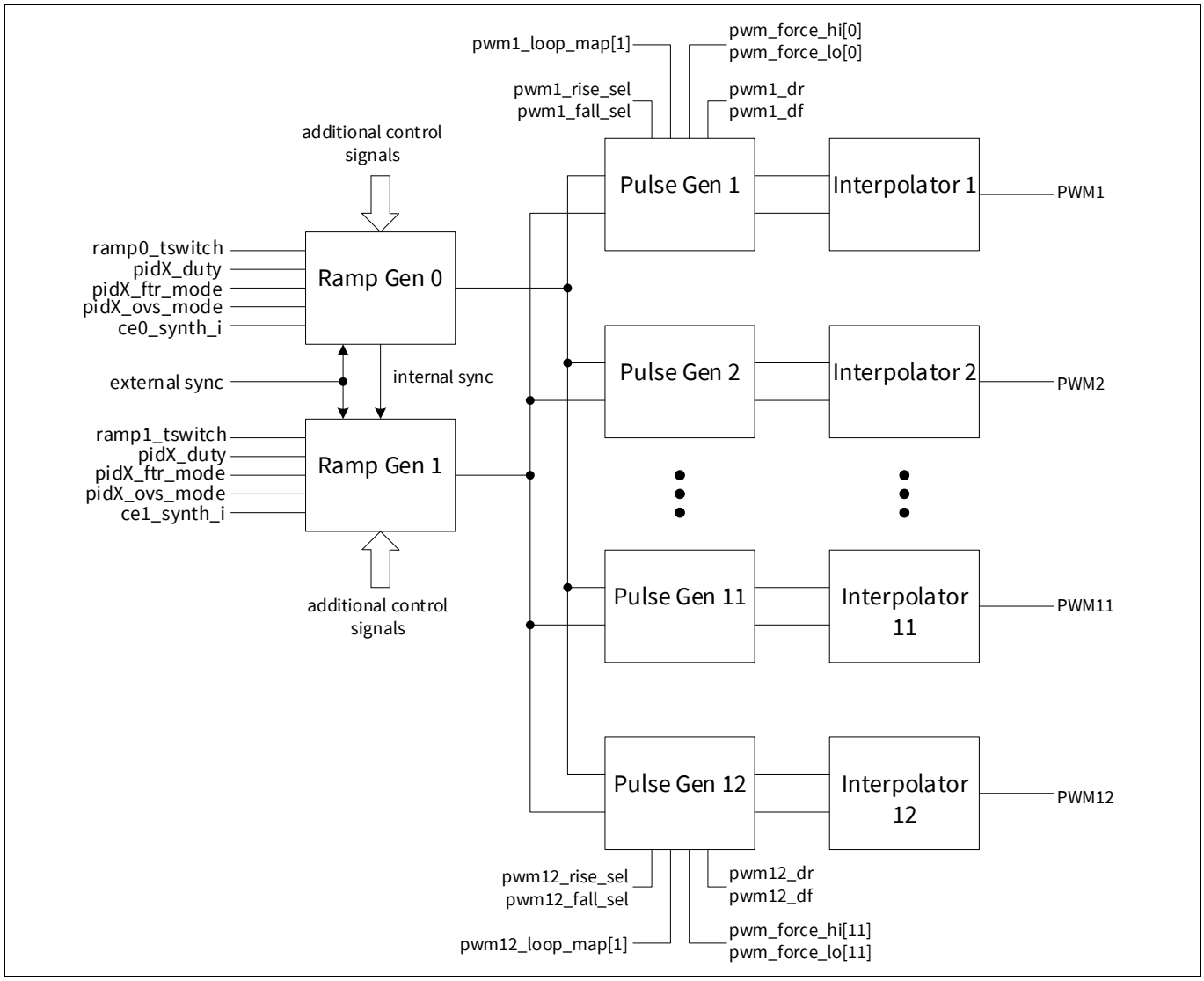

Figure 54 Simplified block diagram of the PWM

7.1 PWM ramp generator

The PWM consists of two ramp generators which support the operation of a single-loop system with up to two phases, or a dual-loop system with a single-phase per loop. The ramp generator produces timing information for the pulse generators for further processing. Each ramp generator receives as its input the compensation filter output. The filter used by each ramp is selected via register **rampX_pid_sel**, where X = 0, denotes ramp generator 0 and X = 1, denotes ramp generator 1. The compensation filter receives its error input from the VS pins. The corresponding PID source settings are:

- **rampX_pid_sel** = 0 selects PID0 (VSEN)
- **rampX_pid_sel** = 1 selects PID1 (BVSEN)

Typical settings of the register **rampX_pid_sel** for the supported system configurations are shown in Table 35.

Table 35 Typical settings of the register rampX_pid_sel

Topology	ramp0_pid_sel	ramp1_pid_sel
Single loop, single-phase	0	0
Single loop, interleaved	0	0
Dual loop	0	1

To generate the PWM pulses, the ramp generator produces the timing information based on a timing ramp. This timing ramp consists of a digital counter which counts from t = 0 up to the maximum ramp count, ramp_max, before returning to 0 and counting up again. The ramp counter functionality is illustrated in Figure 55.

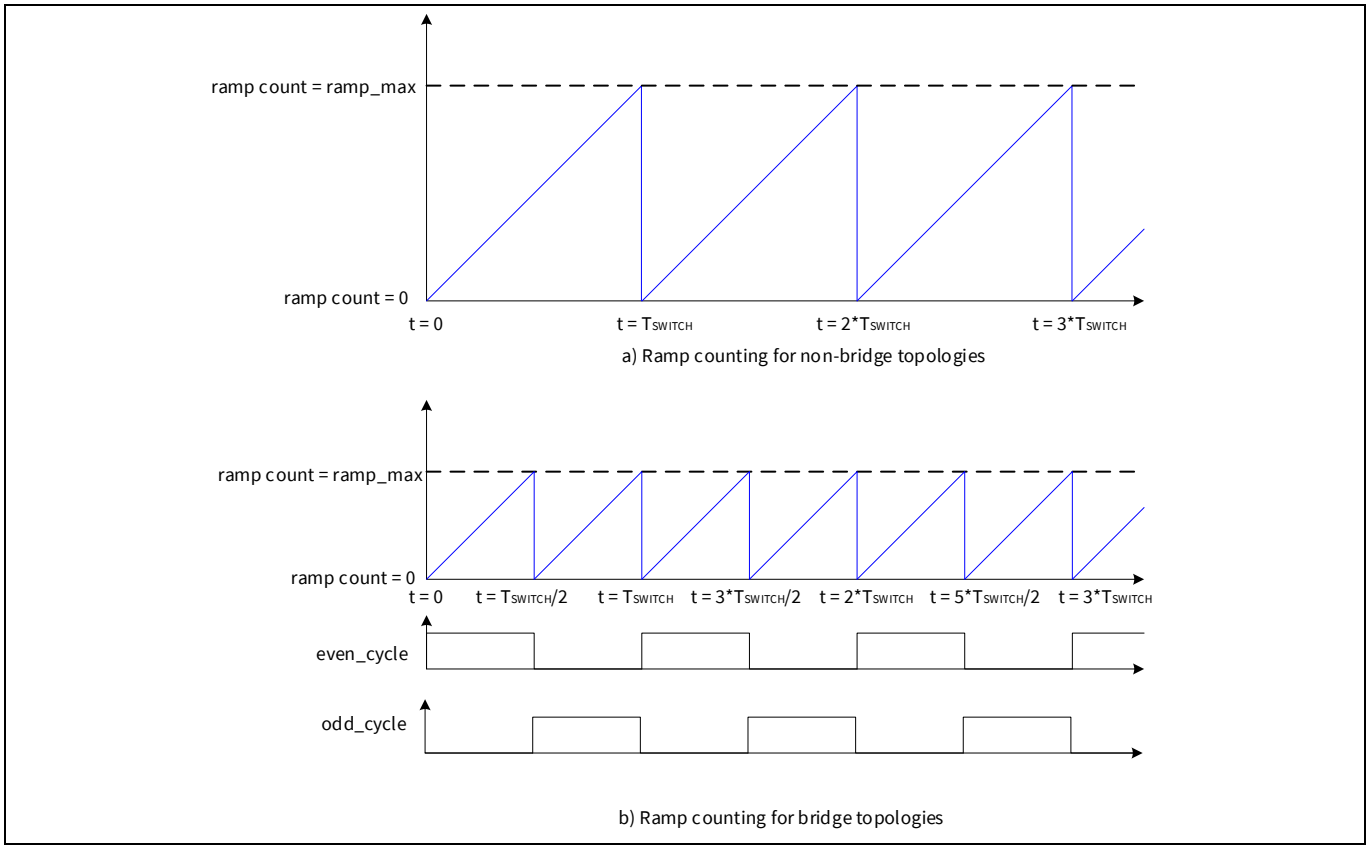

Figure 55 Ramp counter for a) non-bridge and b) bridge topologies

The ramp counter operation depends on the topology:

- For non-bridge topologies (buck, ACF, etc.) the ramp period is equal to the switching period (T_{switch}) and ramp_max is equal to a digital representation of T_{switch}, as shown in the upper part of Figure 55
- For bridge topologies (HB, FB) the ramp period equals half of the switching period (T_{switch} /2) and ramp_max equals a digital representation of T_{switch} /2, as shown in the lower part of Figure 55

In bridge mode, the two halves of the switching cycle are identified as either the even half-cycle or the odd half-cycle, as shown in the bottom part of Figure 55. This identification is used by the pulse generator to produce pulses only on the correct half-cycle.

Register **rampX_half_mode** (X = 0, 1) is used to select between bridge and non-bridge topologies. This register defines whether half-mode is enabled for the ramp. If the mode is enabled, the ramp count (ramp_max) equals half of the switching period. Correspondingly, if half-mode is disabled, the ramp count equals the switching period.

The switching period, T_{switch}, is defined by the register **tswitchX** (X = 0, 1). It is automatically programmed by the FW based on the PMBus command FREQUENCY_SWITCH, which sets the switching frequency in kHz. The register **tswitchX** has LSB weight of 20 ns and range 0.0 to 10.22 µs. Therefore, there are some considerations regarding FREQUENCY_SWITCH values:

- Only values with corresponding T_{switch} that are an exact multiple of 20 ns can be achieved
- If FREQUENCY_SWITCH is set to a value that cannot be achieved in **tswitchX**, the FW will choose the closest achievable setting

The closest achievable FREQUENCY_SWITCH to a target F_{switch} can be found using Equation (7.1).

$$FREQUENCY_SWITCH = \frac{50e6}{ROUND\left(\frac{50e6}{target\ F_{switch}}\right)} \quad (7.1)$$

Note: 50e6 = 1/20e-9.

7.1.1 PWM ramp modulation schemes

The timing information provided by the ramp generator is a pair of timing markers called t1 and t2. These timing markers are used by the pulse generator to define the rising and falling edges of the PWM pulses. The timing markers are discussed further in the pulse generator Section 7.2.

XDPP1100 supports the following modulation schemes for placing t1 and t2:

- Trailing edge (TE) modulation
- Leading edge (LE) modulation
- Dual edge (DE) modulation

These modulation schemes are shown in Figure 56. The first modulation waveform in Figure 56 a) shows the TE modulation case. The timing marker placement for TE modulation:

- t1 is fixed at ramp count = 0
- t2 is modulated based on the selected feedback control mode (Section 7.3)

The example PWM pulse shown in Figure 56 a) has a fixed LE and a modulated TE. The second modulation scheme is LE modulation, shown in Figure 56 b). The timing marker placement for LE modulation is the following:

- t1 is modulated based on the selected feedback control mode (Section 7.3)
- t2 is fixed at ramp count = ramp_max

The example PWM pulse shown in Figure 56 b) has a fixed TE and a modulated LE. The last modulation scheme is DE modulation, shown in Figure 56 c). In this modulation, the timing markers t1 and t2 are both modulated based on the selected feedback control mode (Section 7.3). The example PWM pulse shown in Figure 56 c) has modulated LE and TE.

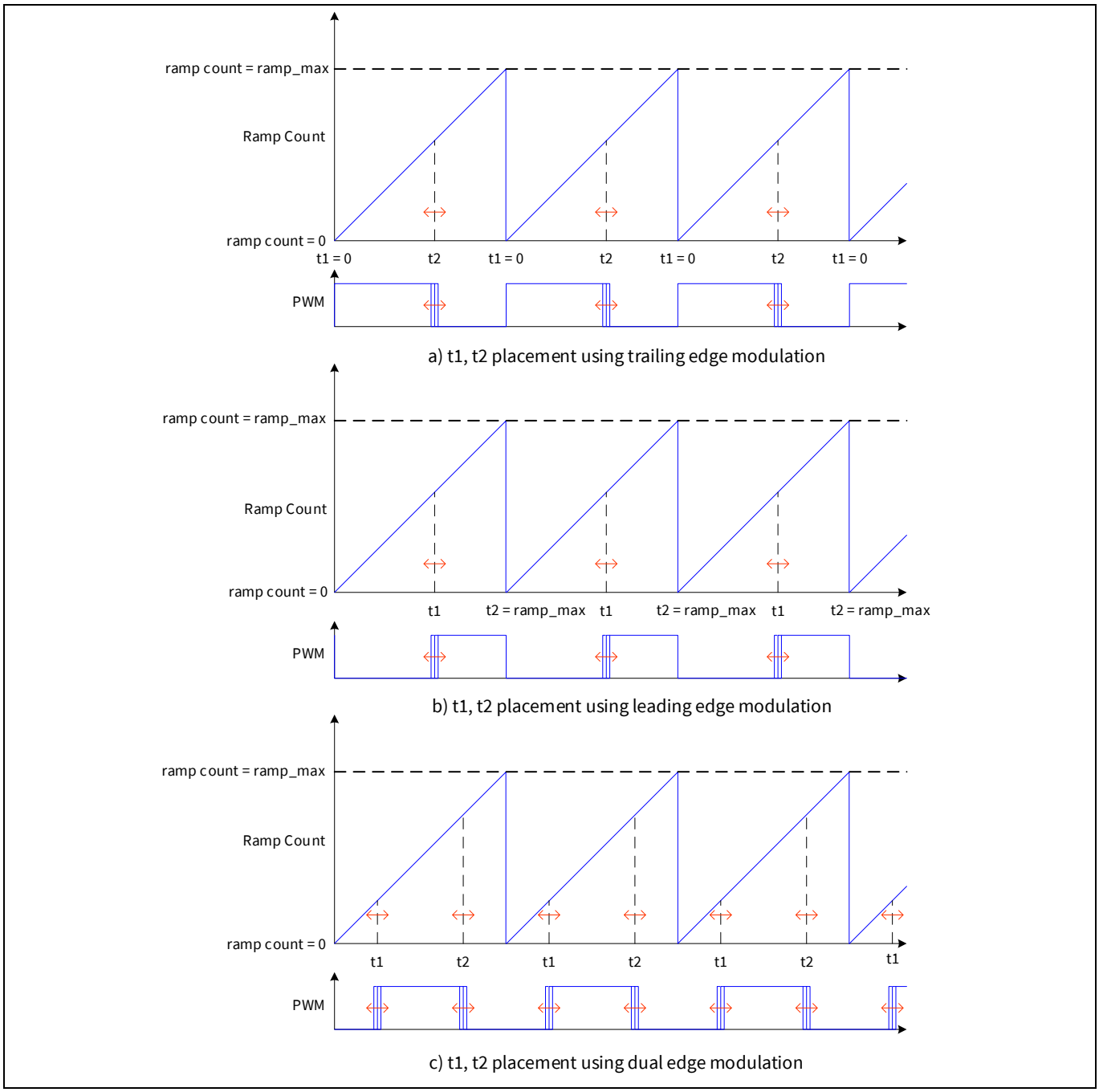

Figure 56 t1 and t2 placement for TE, LE, and DE modulation

The pulse width, t2 – t1, for all modulation schemes is given in Equation (7.2):

$$t_2 - t_1 = D * ramp_max \quad (7.2)$$

D is the duty-cycle output of the compensation filter in the case of VMC. It should be noted that the pulse generator is also capable of creating a pulse with LE at t2 and TE at t1. This pulse would have a pulse width equal to (1 – D) * ramp_max. The modulation scheme for both ramps is selected via register **rampX_m_flavor** (X = 0, 1) as shown in Table 36.

Table 36 Modulation type settings

rampX_m_flavor	Modulation type
0	Dual edge (DE)
1	Leading edge (LE)
2, 3	Trailing edge (TE)

7.1.2 PWM ramp synchronization

Both ramp generators, ramp0 and ramp1, can synchronize to an external sync signal. In addition, ramp1 synchronizes to ramp0 even without the presence of an external sync signal. The synchronization is programmed through the register **rampX_sync_sel** (X = 0, 1).

For ramp0 programming, the register **ramp0_sync_sel** should always be set to 1. Thus, ramp0 synchronizes to an external sync pulse if provided. Otherwise, its period is defined by register **tswitch0** (PAGE0.FREQUENCY_SWITCH). For ramp1 programming, the register **ramp1_sync_sel** should be programmed as follows:

- Set to 1, in order to synchronize to ramp0. If ramp0 is synchronized to an external sync pulse, ramp1 also synchronizes to this pulse. This setting should be chosen for a dual-loop system with $F_{switch1} = F_{switch0}$ or a single-loop system with an interleaved phase
- Set to 0, to use the period defined by register $\mathbf{t}_{switch1}$ (PAGE1.FREQUENCY_SWITCH). This setting should be chosen for a dual-loop system with $F_{switch1} \neq F_{switch0}$ or a single-loop system using only Loop 1 (not typical)

The ramp phase in relation to the external sync pulse, (or ramp1 to ramp0 for internal sync) is defined by the register **rampX_phase** (X = 0, 1). This is illustrated in Figure 57. The value for **rampX_phase** is automatically computed by the FW from the PMBus command PAGEX.INTERLEAVE as given in Equation (7.3).

$$rampX_phase = \frac{256 * PAGEX.INTERLEAVE[3:0]}{PAGEX.INTERLEAVE[7:4]} \quad (7.3)$$

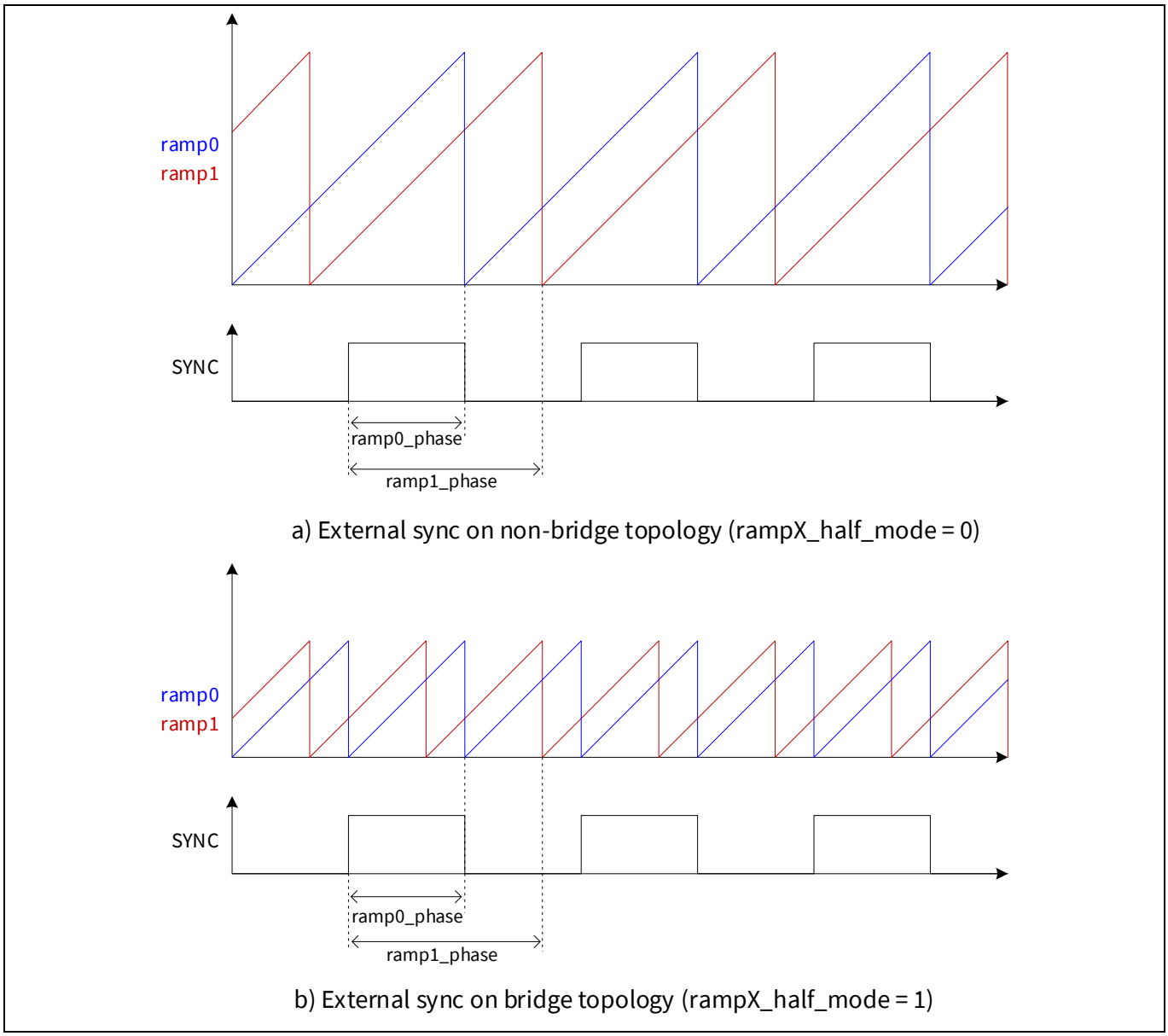

Figure 57 External sync phase programming for non-bridge and bridge topologies

To synchronize to the external signal, various IO pins of the XDPP1100 can be used. These pins are BEN, BPWRGD, EN, FAULT1, FAULT2, IMON, PWRGD, SMBALERT, SYNC and PWM1 through PWM12. Each pin has a corresponding register **<pin_name>_func** (such as **fault1_func**, or **smbalert_func**) which needs to be set to a value of 3 to map the corresponding pin to the external sync function. The direction of the external sync function, input, or output, is defined by register **sync_dir_out**:

- Set to 0 for sync input
- Set to 1 for sync output

Multiple IO pins may be mapped as sync output simultaneously. When the sync function is defined as an input, only one pin is used to receive the external sync pulse. If multiple pins are mapped to the sync function, priority is given to the input with the lowest pin number. This priority order is: FAULT1, FAULT2, IMON, PWM9, PWM10, PWM11, PWM12, PWM1, PWM2, PWM3, PWM4, PWM5, PWM6, PWM7, PWM8, SMBALERT, BEN, BPWRGD, SYNC, EN, and PWRGD.

When a selected IO pin is mapped as an input, the sync function synchronizes to the rising edge of the provided sync pulse. A 40 ns deglitcher is available on the sync input and it is enabled by setting register **sync_deglitch_en** to 1. It should be noted that enabling the deglitcher adds 40 ns of latency between the sync pulse rising edge and the internal ramps.

The following points should be considered while using external synchronization:

- To achieve initial synchronization lock to the incoming pulse, the input pulse frequency must be within 6.25 percent of the internal switching frequency that is programmed by PMBus command FREQUENCY_SWITCH
- Once initial lock is achieved, the sync function maintains lock over a range ±12.5 percent of the internal switching frequency defined by PMBus command FREQUENCY_SWITCH
- In case the sync pulse frequency is outside the lock range or if no pulse is provided, the sync function reverts to the frequency defined by FREQUENCY_SWITCH

To monitor the sync function state in real time, the following read-only registers are available:

- **sync_in_range**
- **sync_state**
- **sync_in_period**
- **sync_fly_period**

Section 7.7 provides a further description of these registers.

7.2 PWM pulse generator

XDPP1100 contains 12 pulse generators, as was shown in Figure 54. There is one dedicated pulse generator for each of the PWM outputs, PWM1 through PWM12. These pulse generators use the timing markers, t1 and t2, from the ramp generator (discussed in Section 7.1.1) to form a PWM pulse with coarse timing. The coarse pulses are then fed to the interpolators where fine resolution timing information is used to create output pulses with 78.125 ps resolution.

7.2.1 Pulse generator enable

Pulse generator, dedicated to certain PWM outputs, is enabled through a PMBus command FW_CONFIG_PWM. This command defines which PWM outputs are allocated for a given system and loop in the following way:

- PAGE0 command defines Loop 0
- PAGE1 command defines Loop 1

The enable programming of each pulse generator depends on the actual system and the selected topology. The command FW_CONFIG_PWM consists of two components defining which topology-specific FETs are assigned to certain PWM outputs and thus a pulse generator. These two components are:

- FW_CONFIG_PWM[11:0] = pwm_on_mask[11:0]
- FW_CONFIG_PWM[27:16] = pwm_srfet_mask[11:0]

Both of these masks are used by the FW to enable the HW pulse generators. The first component pwm_on_mask[11:0] defines which PWM outputs correspond to:

- Primary-side FETs in an isolated topology
- High-side FETs in a non-isolated buck topology
- Power FETs in non-isolated boost or buck-boost topologies

The other mask pwm_srfet_mask[11:0] defines which PWM outputs correspond to:

- Secondary-side FETs in an isolated topology
- Low-side FETs in a non-isolated buck topology
- SR FETs in non-isolated boost or buck-boost topologies

In both masks, bit [0] corresponds to PWM1 and bit [11] corresponds to PWM12. In addition, during the diode emulation operating mode (e.g., during diode emulation start-up), the mask pwm_srfet_mask[11:0] identifies for the FW which PWM outputs should be disabled.

7.2.2 Ramp selection (loop/phase)

For each pulse generator, the ramp can be chosen via register **pwmY_loop_map**, where Y = 1 to 12. This selection is handled by the parameter **pwmY_loop_map** (Y = 1 to 12), as shown in Table 37.

Table 37 Pulse generator ramp selection via pwmY_loop_map

pwmY_loop_map	Ramp assignment	Definition
0	Unassigned	PWM not in use
1	ramp0	Loop 0, Phase 1
2	ramp1	Loop 1, Phase 1
3	ramp1	Loop 0, Phase 2

For standard topologies (HB, FB, forward, buck, etc.) the following considerations apply:

- ramp0 is generally assigned to Loop 0, Phase 1
- ramp1 is assigned to either Loop 0, Phase 2 or Loop 1, Phase 1

The XDPP1100 also supports potential non-standard uses of the register **pwmY_loop_map**. Nevertheless, it is critical that the loop assignment is correct, as this is used for the HW-based fault-induced shutdown. For example, if both ramp0 and ramp1 are used by Loop 0, Phase 1, the **pwmY_loop_map** assignment to select ramp1 should be 3 since that corresponds to Loop 0.

7.2.3 Edge selection (t1/t2)

The ramp generator provides timing markers t1 and t2. The rising and falling edges of the pulse are mapped to the timing markers by the registers:

- **pwmY_rise_sel**
- **pwmY_fall_sel**

Y equals a number from 1 to 12. The options for the rising edge selection are shown in Table 38.

Table 38 Pulse generator rising edge selection

pwmY_rise_sel	PWM rising edge selection
0	t1
1	t2
2	t1 even
3	t2 even
4	t1 odd

pwmY_rise_sel	PWM rising edge selection
5	t2 odd
6	t1 delay
7	t2 delay
8	VRSEN falling edge
9 to 15	BVRSEN falling edge

The selection for the rising edge depends on the topology. Typical selections are for:

- **Non-bridge topologies:** t1 or t2, create an edge on every t1 or t2 event
- **Bridge topologies:** t1 or t2 even, t1 or t2 odd, creates an edge only on the even or odd half-cycle t1 or t2 event

The option t1 delay and t2 delay are used to create a short fixed-width pulse on t1 or t2. To create a short positive pulse, select for:

- Rising edge, t1 or t2
- Falling edge, t1 or t2 delay and adjust the falling edge dead-time (see Section 7.2.4) to set the pulse width

The VRSEN and BVRSEN options allow the definition of the rising edge based on the falling edge of V_{RECT}. This option may be used for example to turn on the secondary-side SR FET only after V_{RECT} has gone low. Correspondingly, the falling edge options are shown in Table 39.

Table 39 Pulse generator falling edge selection

pwmY_fall_sel	PWM falling edge selection
0	t1
1	t2
2	t1 even
3	t2 even
4	t1 odd
5	t2 odd
6	t1 delay
7	t2 delay

The selection of the falling edge is the same as defined for the rising edge. The only exception is that the options VRSEN or BVRSEN are not available for the falling edge.

The XDPP1100 GUI can be used to automatically program the registers **pwmY_loop_map**, **pwmY_rise_sel** and **pwmY_fall_sel** based on topology selection and PWM output mapping. Some example settings for these registers are shown in Figure 58 for the standard topologies.

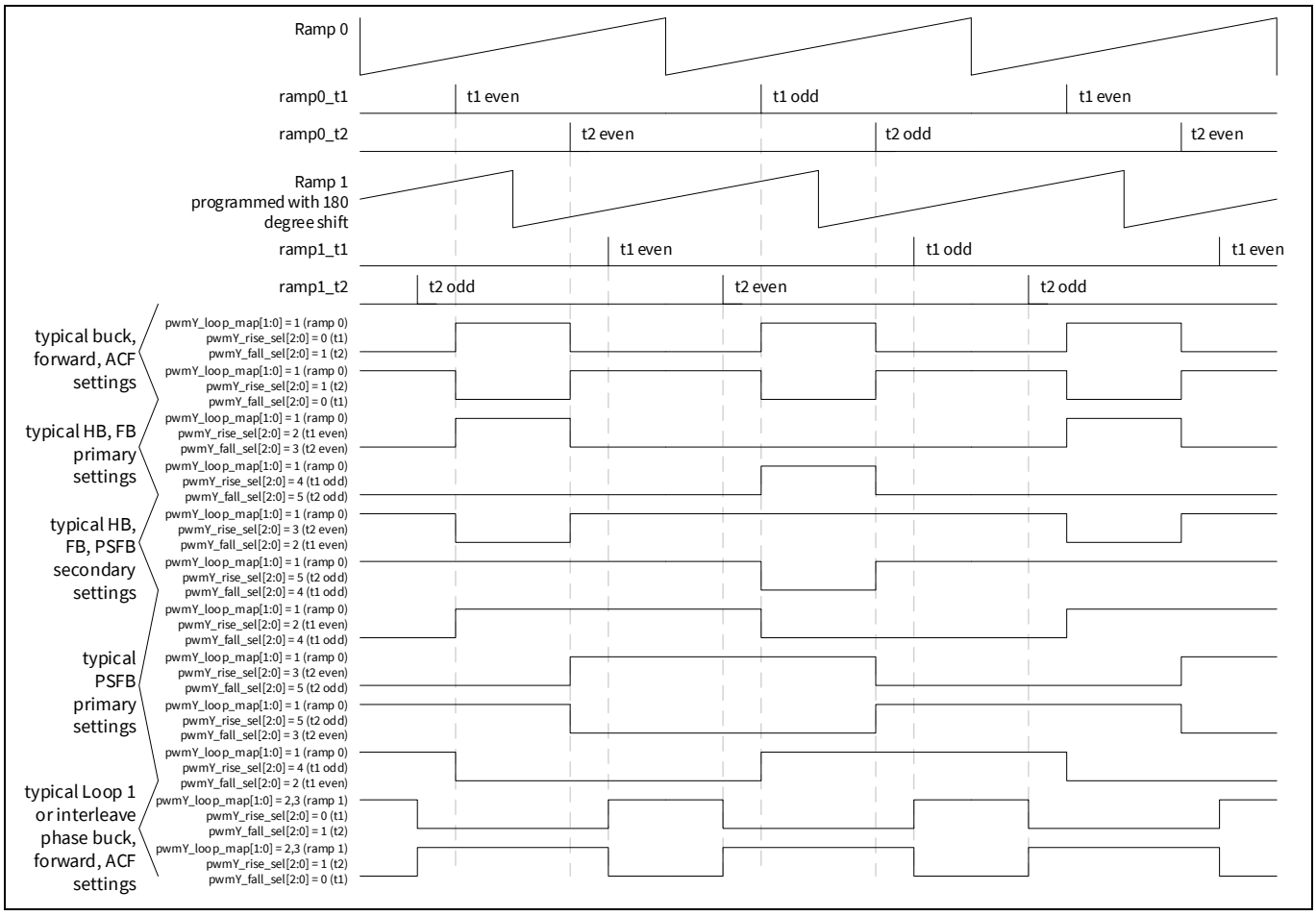

Figure 58 Typical pwmX_loop_map, pwmX_rise_sel and pwmX_fall_sel settings for various topologies

7.2.4 Dead-time programming

XDPP1100 can apply independent dead time (delay) to both the rising and the falling edges of each PWM output. The dead time programming for the edges is the following:

- Rising edge dead time, via register **pwmY_dr** (Y = 1 to 12)
- Falling edge dead time, via **pwmY_df**

Figure 59 shows the dead time programming, where both parameters have a range of 0.0 to 318.75 ns with the resolution of 1.25 ns. The settings for all **pwmY_dr** and **pwmY_df** are automatically computed based on the programming of PMBus command PWM_DEADTIME. The mapping from PWM_DEADTIME to each **pwmY_dr** and **pwmY_df** is shown in Table 40.

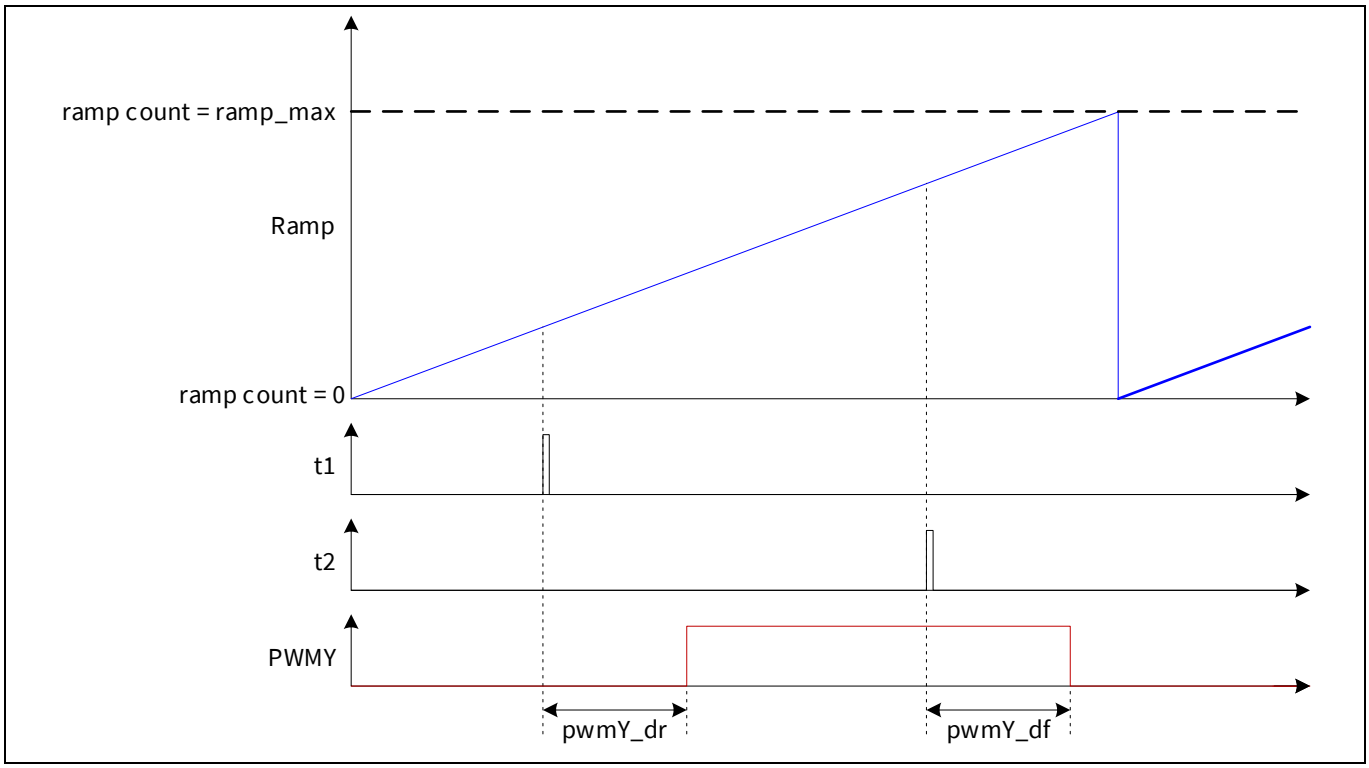

Figure 59 Dead-time programming

Table 40 PMBus command PWM_DEADTIME mapping to pwmY_dr and pwmY_df

Register bit range (PWM_DEADTIME)	Field name	Register bit range (PWM_DEADTIME)	Field name
PWM_DEADTIME[7:0]	pwm1_df[7:0]	PWM_DEADTIME[103:96]	pwm7_df[7:0]
PWM_DEADTIME[15:8]	pwm1_dr[7:0]	PWM_DEADTIME[111:104]	pwm7_dr[7:0]
PWM_DEADTIME[23:16]	pwm2_df[7:0]	PWM_DEADTIME[119:112]	pwm8_df[7:0]
PWM_DEADTIME[31:24]	pwm2_dr[7:0]	PWM_DEADTIME[127:120]	pwm8_dr[7:0]
PWM_DEADTIME[39:32]	pwm3_df[7:0]	PWM_DEADTIME[135:128]	pwm9_df[7:0]
PWM_DEADTIME[47:40]	pwm3_dr[7:0]	PWM_DEADTIME[143:136]	pwm9_dr[7:0]
PWM_DEADTIME[55:48]	pwm4_df[7:0]	PWM_DEADTIME[151:144]	pwm10_df[7:0]
PWM_DEADTIME[63:56]	pwm4_dr[7:0]	PWM_DEADTIME[159:152]	pwm10_dr[7:0]
PWM_DEADTIME[71:64]	pwm5_df[7:0]	PWM_DEADTIME[167:160]	pwm11_df[7:0]
PWM_DEADTIME[79:72]	pwm5_dr[7:0]	PWM_DEADTIME[175:168]	pwm11_dr[7:0]
PWM_DEADTIME[87:80]	pwm6_df[7:0]	PWM_DEADTIME[183:176]	pwm12_df[7:0]
PWM_DEADTIME[95:88]	pwm6_dr[7:0]	PWM_DEADTIME[191:184]	pwm12_dr[7:0]

Two additional PMBus commands are provided to allow automatic dead time adjustment based on the output current. These commands are:

- PWM_DEADTIME_ADJUST
- FW_CONFIG_DEADTIME_ADJUSTMENT_THRESHOLD

Under specific conditions, the dead times are defined by the command PWM_DEADTIME_ADJUST instead of those defined by PWM_DEADTIME. This condition occurs if:

- Command FW_CONFIG_DEADTIME_ADJUSTMENT_THRESHOLD is non-zero
- And I_{OUT} is greater than FW_CONFIG_DEADTIME_ADJUSTMENT_THRESHOLD

It should be noted that I_{OUT} polling for this FW-based function occurs at a 1 ms rate.

More complex dead time adjustment algorithms are additionally possible using a user-written FW patch. It is important to note that to synchronously and simultaneously update all the dead time changes, an update to any of the dead time parameters (**pwmY_dr** or **pwmY_df**) become effective only after the register containing **pwm12_dr** and **pwm12_df** is written. This means that even if the PWM12 output is not in use, a dummy write to register 7000_2C80h by FW is required.

7.2.5 Force high and force low

XDPP1100 has two registers which allow the PWM output state to be forced. By setting appropriate bits in either of the following register values to 1, the corresponding PWM output can be forced either high or low:

- Register **pwm_force_hi[11:0]** forces the corresponding PWM output high
- Register **pwm_force_lo[11:0]** forces the corresponding PWM output low

In both registers, bit 0 corresponds to PWM1 and bit 11 corresponds to PWM12. In the case that corresponding bits are set in both parameters, **pwm_force_lo** has higher priority and the PWM output is set low.

7.3 Feedback control modes

The XDPP1100 supports VMC and PCMC. In the case of PCMC, both primary-side and secondary-side control are possible. The feedback control mode is selected via register **mode_control_loopX**, where X denotes zero or one depending on which loop is being used. The control method programming is shown in Table 41.

Table 41 Feedback control mode programming

mode_control_loopX	Feedback control mode
0	VMC
1	PCMC on secondary
2	PCMC on primary
3	Reserved

7.3.1 Voltage mode control

VMC is the simplest control method, and it can be configured by setting the register value **mode_control_loopX** to 0, as was described in Table 41. The functional block diagram shown in Figure 60 illustrates the internal configuration while the XDPP1100 applies VMC.

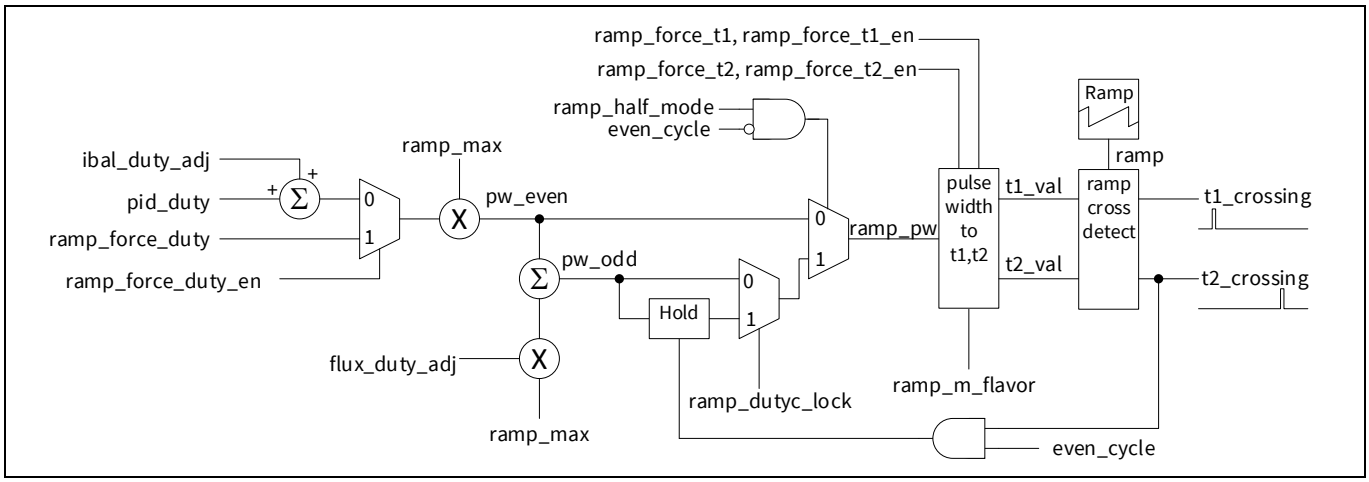

Figure 60 Functional block diagram of VMC

In VMC, the compensator output, pid_duty, is directly interpreted as the duty cycle with a range of 0.0 to 0.9999. In addition, two other sources contribute to the final duty cycle used to compute the PWM pulse width (PW). These sources are as follows:

- Current balance duty-cycle adjustment (ibal_duty_adj). This signal comes from the current balance function (which will be described in Section 11). It corrects for difference in the phase current in a dual-phase (interleave) loop and it is applied only to ramp0. This is because the phase associated with ISEN contains the current balance duty-cycle adjustment while the phase associated with BISEN does not
- Flux balance duty-cycle adjustment (fbal_duty_adj). This signal comes from the flux balance function (which will be described in Section 12). It corrects for the transformer flux (Volt-second) differences between the even and odd half-cycles in the FB topology and it is applied only to the odd half-cycle

As shown in Figure 60, the duty cycle components are multiplied by ramp_max to convert from duty cycle to pulse width, ramp_pw. The pulse width is then converted to target t1 and t2 values, t1_val and t2_val, based on the modulation type as show in Table 42.

Table 42 t1 and t2 computation by edge modulation type

Modulation type	t1_val	t2_val
Dual edge (DE)	(ramp_max - ramp_pw) / 2	(ramp_max + ramp_pw) / 2
Leading edge (LE)	ramp_max - ramp_pw	ramp_max
Trailing edge (TE)	0	ramp_pw

The target t1 and t2 values are then compared against the ramp waveform to create the t1_crossing and t2_crossing signals used by the pulse generators to define the PWM edges, as described in Section 7.2.3.

Duty-cycle lock mode is intended for use in FB topologies where it is important to maintain the flux balance between the two half-cycles. In this mode, the odd half-cycle PW is sampled and held on the even half-cycle t2 crossing detection. In the case of TE modulation t1 = 0 and this leads to equal odd and even cycle PWs based on the contributions from the compensator (PID) and current balance. The duty-cycle lock mode is enabled by setting the register **rampX_dutyc_lock** to 1.

It should be noted that the flux balance adjustment still contributes to the odd cycle PW even when duty-cycle lock is enabled. This means that the PWs will not be identical if compensation is required to correct some external deviation (e.g., differences in driver PW propagation).

7.3.2 Peak current mode control

PCMC is based on current information, and it can be selected by setting the register value **mode_control_loopX** to 1 for secondary-side PCMC or 2 for primary-side PCMC. In the case of PCMC the compensator output represents the control current, whereas for VMC it was directly the duty cycle. Figure 61 shows the functional block diagram of the XDPP1100 internal configuration when PCMC is applied.

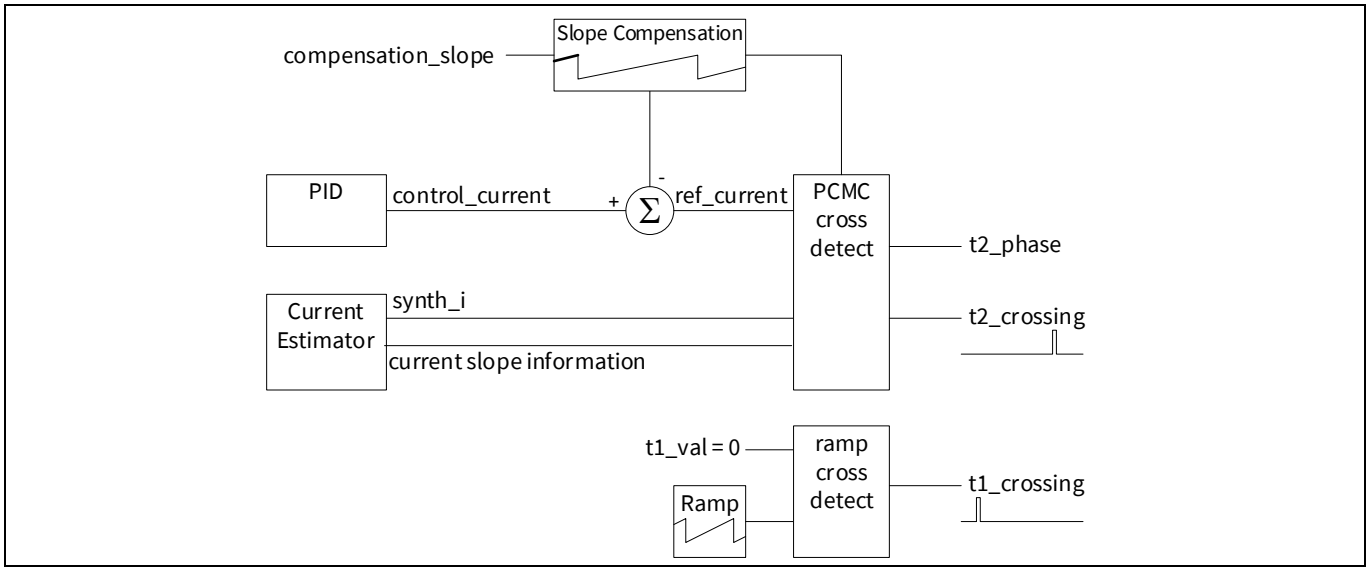

Figure 61 Functional block diagram of PCMC

PID selection for ramp0 and ramp1 is programmable, whereas the CE selection is hardwired. This is because:

- Current estimator CE0 (ISEN) is connected to ramp0
- Current estimator CE1 (BISEN) is connected to ramp1
- Current used for control (either primary or secondary) should be sensed by ISEN/IREF in a single-loop system using ramp0

The PID output represents “normalized control current”, where the normalization is to the maximum range of the IADC. This leads to a signed output range of -1.0 to +1.0. PCMC supports only TE modulation, and therefore the register **rampX_m_flavor** should be set to value 2. This means that the t1 crossing event always occurs at ramp = 0.

PCMC requires slope compensation to prevent subharmonic oscillation when the duty cycle is higher than 0.5. A slope compensation ramp is provided, and its value is programmable through register **compensation_slope**, as shown in Table 43.

Table 43 PCMC slope compensation values

Compensation_slope	Slope of compensation ramp
0	V_{OUT}/L
1	$V_{OUT}/2L$
2	$V_{OUT}/4L$
3	Reserved

As shown in Figure 61, the output of the slope compensation ramp is subtracted from the control current to obtain the reference current. The reconstructed current output from the CE, synth_i, is then compared against this reference current. Both currents are digital and discrete in time. Therefore, to determine the convergence rate of these two signals, the PCMC cross-detect function uses the slope information from both the slope compensation ramp and the CE. This convergence rate information is used to:

- Predict the t2 crossing in the next clock cycle
- Determine the phase of the t2 crossing within the clock cycle

The phase is used by the interpolators for fine timing resolution. It should be noted that in the PCMC the PWM PW resolution is limited to 625 ps compared to the 78.125 ps available in VMC.

7.3.3 Maximum and minimum pulse width enforcement

The maximum PW is possible to enforce. This can be performed by using either a fixed or variable method. These methods are adjusted in the following ways:

- Fixed method: Register **rampX_dc_max** defines the fixed maximum duty cycle for the ramp X, where the register value is computed by FW from PMBus command MAX_DUTY, as shown in Equation (7.4)
- Variable method: Register **rampX_dc_max_nom** defines the variable maximum duty-cycle limit for the ramp X, where the variable limit scales the maximum duty cycle with the measured V_{RECT} to limit the transformer flux at high V_{IN}. The scaling is with respect to register **pid_vrect_ref** as given in Equation (7.5)

$$fixed_\max_duty = rampX_dc_max = MAX_DUTY \tag{7.4}$$

$$variable_max_duty = rampX_dc_\max_nom * \frac{pid_vrect_ref}{V_{RECT}} \tag{7.5}$$

Setting **rampX_dc_max_nom** to 0 disables the variable maximum duty-cycle limit. The applied maximum duty limit is the minimum of the fixed and variable limits assuming the variable limit is enabled, as shown below.

```
if (rampX_dc_max_nom>0)
      max_duty = MIN(fixed_max_duty, variable_max_duty)
else
      max_duty = fixed_max_duty
```

The maximum PW limit is computed as the product of the maximum duty cycle limit and the maximum ramp value according to Equation (7.6).

$$pw_max = \max_duty * rampX_max \tag{7.6}$$

The minimum PW is defined by register **rampX_pw_min** and its value is automatically computed by the FW based on the PMBus command MFR_MIN_PW according to Equation (7.7).

$$rampX_pw_min = MFR_MIN_PW \tag{7.7}$$

Setting this register value to 0 disables the minimum PW enforcement. Two methods of minimum PW enforcement are supported in VMC. They can be defined through register **rampX_min_pw_state** in the following way:

- register value 0, sets PW to 0 (blank pulse)
- register value 1, sets PW to **rampX_pw_min** value (clamp pulse to min.)

In PCMC, only the clamp to min. method is supported.

7.3.4 Forced duty cycle T1, T2

XDPP1100 provides several methods through different registers for overriding the HW computed duty cycle or PW. The registers used for overriding the duty cycle are:

- **pid_force_duty** and **pid_force_duty_en**
- **rampX_force_duty** and **rampX_force_duty_en**
- **rampX_force_t1** and **rampX_force_t1_en**
- **rampX_force_t2** and **rampX_force_t2_en**

The first method was already discussed in the compensator chapter, in Section 6.1.5 where the duty cycle can be forced at the PID output using the registers **pid_force_duty** and **pid_force_duty_en**. Regarding this method, it should be emphasized that:

- In a dual-phase system these registers can be used to apply the same duty cycle to both ramps simultaneously
- These settings are applied before the current balance adjustment, and if current balance function is required, this is the only method to override the duty cycle
- These PID override parameters are stored in the OTP so they can be used to configure the XDPP1100 to apply permanent fixed duty cycle in an open-loop control mode without applying a FW patch

The second method overrides the duty using registers **rampX_force_duty** and **rampX_force_duty_en** and it is applied subsequent to the current balance adjustment, as was shown in Figure 60. The operation of this duty override is the following:

- **rampX_force_duty** is applied at the beginning of the next ramp cycle (i.e., ramp = 0) after **rampX_force_duty_en** is set to 1
- Subsequent changes to **rampX_force_duty** are applied at the beginning of the next ramp cycle after the change

It is possible to monitor when the new duty cycle force has been applied on the ramp. This can be done by observing the register **rampX_dutyc_force_status** bit. This register bit is:

- Cleared to 0, when either **rampX_force_duty** or **rampX_force_duty_en** change
- Set to 1, when new duty-cycle force has been applied on the ramp at the start of the next ramp cycle

The ramp-based override parameters are preferred when each ramp requires a different duty cycle or when the current balance adjustment must be temporarily or permanently overridden. The ramp override parameters are not stored in the OTP; therefore, the FW-control is required to apply these parameters.

For the last two methods, registers **rampX_force_t1** and **rampX_force_t1_en** may be used to override the HW computed t1_val, and registers **rampX_force_t2** and **rampX_force_t2_en** may be used to override the HW computed t2_val. In both cases the parameters are not stored in the OTP, so FW control is required to apply these parameters.

Both parameters **rampX_force_t1** and **rampX_force_t2** have an LSB resolution of 5 ns. It is possible to provide additional time resolution to both t1 and t2 via **rampX_force_duty[5:0]** according to Equations (7.8) and (7.9) when t1 and t2 are forced.

$$t1_val = \{rampX_force_t1[10:0], rampX_force_duty[5:0]\} \quad (7.8)$$

$$t2_val = \{rampX_force_t2[10:0], rampX_force_duty[5:0]\} \quad (7.9)$$

This method is useful under TE or LE modulation when either t1 or t2 is set to a fixed value, because both t1 and t2 use the same **rampX_force_duty** parameter for the LSBs.

It should be noted that for DE modulation (if the pulse on and off times allow), it is possible to use interrupts on t1 and t2 (see Section 7.6) in order to change **rampX_force_duty** between edges.

7.4 Burst mode

Burst mode (BM) is a light-load operating state that is only supported in bridge topologies. During this state, the controller provides a burst of PWM pulses upon the error voltage dropping below a programmed threshold. The BM is enabled by setting PMBus command POWER_MODE = 0 and disabled by setting POWER_MODE = 3.

The BM is entered when the output current falls below the level defined by the register **pid_burst_mode_ithr**, which is computed according to Equation (7.10).

$$pid_burst_mode_ithr = \frac{2*Iout_burst_entry_threshold}{MFR_IOUT_APC} \quad (7.10)$$

Where Iout_burst_entry_threshold is the output current value in amps at which to enter the BM and MFR_IOUT_APC is the PMBus command defining the APC setting of the IADC. BM entry and operation are illustrated in Figure 62.

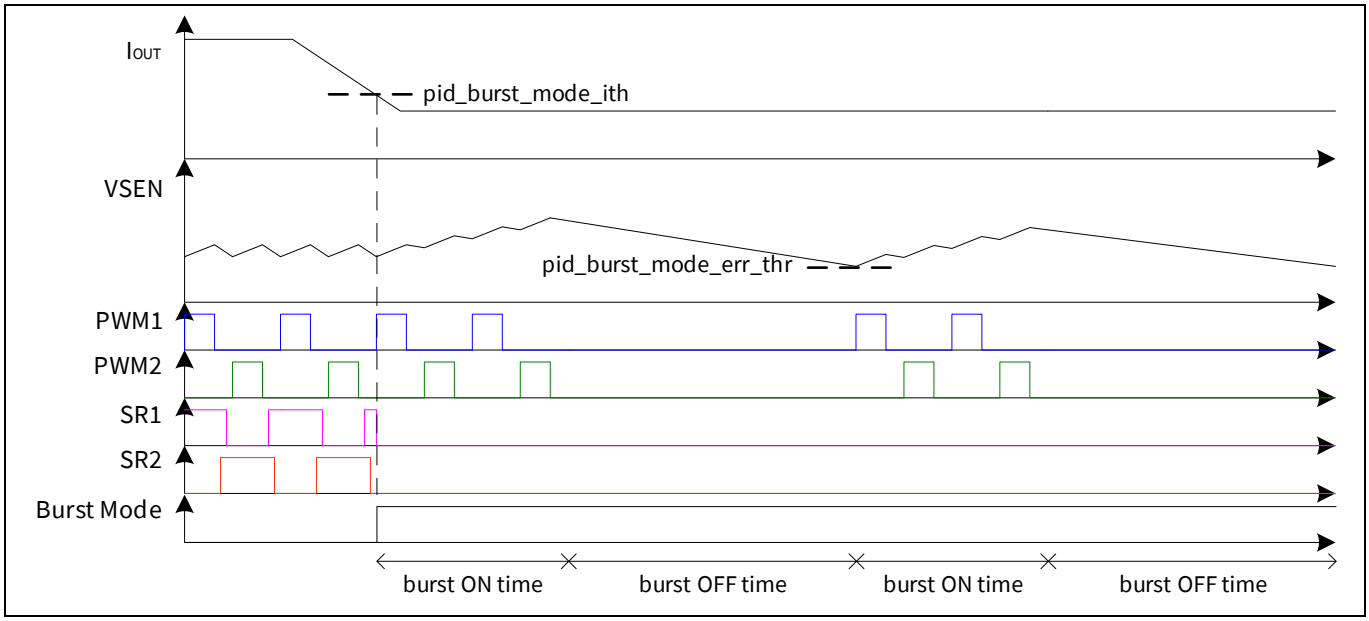

Figure 62 BM operation with pid_burst_reps = 1

Upon entering BM, XDPP1100 disables the SR FETs and outputs a burst of PWM pulses on each half-cycle corresponding to a number programmed via register **pid_burst_reps** as defined in Table 44.

Table 44 Burst count programming at BM

pid_burst_reps	Number of PWM pulses per burst
0	1
1	2
2	4
3	8

Subsequent to the initial burst, the PWM outputs are disabled until the output voltage drop produces an error voltage greater than the value defined in register **pid_burst_mode_err_thr**. This parameter is computed according to Equation (7.11).

$$pid_burst_mode_err_thr = \frac{Vout_burst_err * VOUT_SCALE_LOOP * 2^{-16}}{1.25mV} \quad (7.11)$$

Where Vout_burst_err is the amount (in volts) that the output voltage must drop to trigger the start of the next burst and VOUT_SCALE_LOOP is the PMBus command defining the V_{OUT} resistor divider ratio.

As soon as V_{OUT} drops below the threshold, another burst of PWM pulses is initiated. This process continues until the burst off-time (defined as the time between the end of one burst and the start of the next one) is less than $T_{switch}/2$. The off time in any given system is based on the load current, the output capacitance, the inductor value and the number of PWM pulses per burst.

Increasing the number of PWM pulses per burst increases the peak inductor current, resulting in a longer off-time for a given load current and output capacitor at the expense of increased voltage ripple. Upon exiting the BM, the SR FETs are re-enabled, and the PWM outputs resume normal operation under compensator control.

7.5 Fast-transient response

The system response to load transients depends on the applied PID compensator and the resulting BW of the feedback loop. The BW is typically 1/6 to 1/10 of the switching frequency. A slow response results in increased V_{OUT} undershoot or overshoot. To improve the transient performance, the XDPP1100 supports two fast response modes for transient improvement. These are:

- Fast-transient response for load steps
- Fast-overshoot response for load release

Both modes are supported only for bridge topologies. The following sections describe these methods in more detail.

7.5.1 Fast-transient response – load step

The controller maximizes the duty cycle when a positive load step is detected. This is obtained by reducing the switching period from the programmed T_{switch} to 2*T_{ON}, where T_{ON} is given in Equation (7.12).

$$T_{ON} = \left(\frac{T_{SWITCH}}{2}\right) * \left(\frac{V_{OUT}}{V_{RECT}}\right) \quad (7.12)$$

An example of the fast-transient response event is shown in Figure 63. As a result of the fast transient:

- The duty cycle is nearly 100 percent as observed at the PWM outputs, and only the dead time contributes to The PWM off-time. It should be noted that the dead time effect is not shown in the figure
- V_{OUT} undershoot is reduced due to the provided maximum current increase in the inductor
- Transformer flux (B in the figure) remains bounded since T_{ON} is roughly the same as in normal regulation

At high V_{RECT} where the on-time inductor current slope is higher, it is possible to reduce T_{ON} in proportion to ($1/V_{IN}$). This is performed by programming a threshold value in the register **lpX_ftr_vin_thresh**, where:

- Below the threshold, the fast-transient T_{ON} is based on the FF duty cycle from the PID (i.e., V_{OUT}/V_{RECT}) and T_{ON} is as defined in Equation (7.12)
- Above the threshold, the fast-transient T_{ON} is reduced in proportion to V_{IN} and T_{ON} is as defined in Equation (7.13)

$$T_{ON} = \left(\frac{T_{SWITCH}}{2}\right) * \left(\frac{V_{OUT}}{V_{RECT}}\right) * \left(\frac{lp0_ftr_vin_thresh}{V_{IN}}\right) \quad (7.13)$$

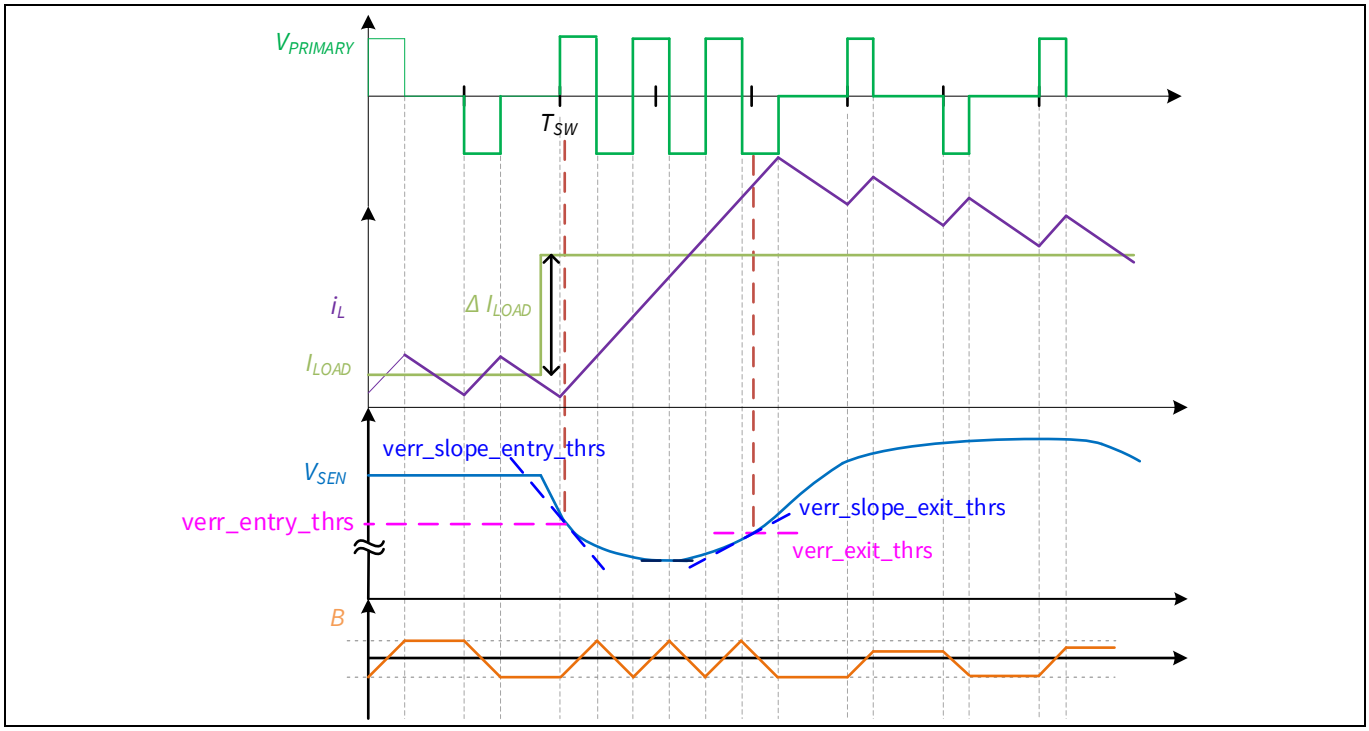

Figure 63 Fast-transient response waveform

Two registers control the entry into the fast-transient response mode. These are:

- **pid_verr_entry_thrs**, which defines the error voltage (V_{err} = target voltage – sensed voltage) entry threshold as observed at VSEN (i.e., after scaling by the V_{OUT} resistor divider)
- **pid_verr_slope_entry_thrs**, which defines the error voltage slope entry threshold

The control loop enters fast-transient response mode when both of the following conditions are met:

- V_{err} is greater than **pid_verr_entry_thrs**
- V_{err} slope is greater than **pid_verr_slope_entry_thrs**

Note that Figure 63 shows these conditions occurring simultaneously. When this is not the case, the realization of the second condition triggers the entry. Correspondingly, two registers control the exit from the fast-transient response mode. These are:

- **pid_verr_exit_thrs**, which defines the error voltage (V_{err} = target voltage – sensed voltage) exit threshold as observed at VSEN (i.e., after scaling by the V_{OUT} resistor divider)
- **pid_verr_slope_exit_thrs**, which defines the error voltage slope exit threshold

The control loop exits the fast-transient response mode when both of the following conditions are met:

- V_{err} is less than **pid_verr_exit_thrs**
- V_{err} slope is less than **pid_verr_slope_exit_thrs**

Note that Figure 63 shows these conditions occurring simultaneously. When this is not the case, the realization of the second condition triggers the exit.

The fast-transient response mode is enabled by setting the register **pid_verr_entry_thrs** to any non-zero value and saving to the OTP. While programming this register, it should be noted that:

- At the system boot the FW reads and saves the setting of **pid_verr_entry_thrs** and then sets it to 0
- On converter enable, the saved setting is restored to the register upon reaching the initial target voltage
- On converter disable it is reset again to 0

The register value does not need to be stored in OTP to be effective and, like most other registers, changes while regulating are permitted (e.g., to fine-tune response).

The likelihood of triggering fast-transient response due to the switching noise at the VSEN pin can be reduced. For this purpose, an LPF may be applied to the error voltage used in the above comparisons. The filter BW is controlled by register **pid_ftr_lpf** as shown in Table 45. This register is applicable for both fast-transient and fast-overshoot response.

Table 45 Low-pass filter BW programming

pid_ftr_lpf	Fast transient/overshoot V_{err} BW
0	1 MHz
1	2 MHz
2	4 MHz
3	8 MHz
4 to 7	Filter bypassed

Refer to the Application Guide – Digital power controller XDPP1100 for additional information regarding system tuning with respect to fast-transient response.

7.5.2 Fast-transient response – load release

The controller minimizes the duty cycle when a negative load step is detected. This is performed by immediately reducing the PWM PW to the programmed minimum (which may be 0). This reduces V_{OUT} overshoot by providing maximum current decrease in the inductor. Figure 64 shows an example of the fast-overshoot response event.

Two registers control the entry into the fast-overshoot response mode. These are:

- **pid_ovs_entry_thrs**, which defines the error voltage entry threshold as observed at VSEN (i.e., after scaling by the V_{OUT} resistor divider)
- **pid_ovs_slope_entry_thrs**, which defines the error voltage slope entry threshold

The control loop enters fast-overshoot response mode when both of the following conditions are met:

- V_{err} is less than **pid_ovs_entry_thrs**
- V_{err} slope is less than **pid_ovs_slope_entry_thrs**

Note that Figure 64 shows these conditions occurring simultaneously. When this is not the case, the realization of the second condition triggers the entry.

Correspondingly, two registers control the exit from the fast-overshoot response mode. These are:

- **pid_ovs_exit_thrs**, which defines the V_{err} exit threshold as observed at VSEN (i.e., after scaling by the V_{OUT} resistor divider)
- **pid_ovs_slope_exit_thrs**, which defines the error voltage slope exit threshold

The control loop exits the fast-overshoot response mode when both of the following conditions are met:

- V_{err} is greater than **pid_ovs_exit_thrs**
- V_{err} slope is greater than **pid_ovs_slope_exit_thrs**

Note that Figure 64 shows these conditions occurring simultaneously. When this is not the case, the realization of the second condition triggers the exit.

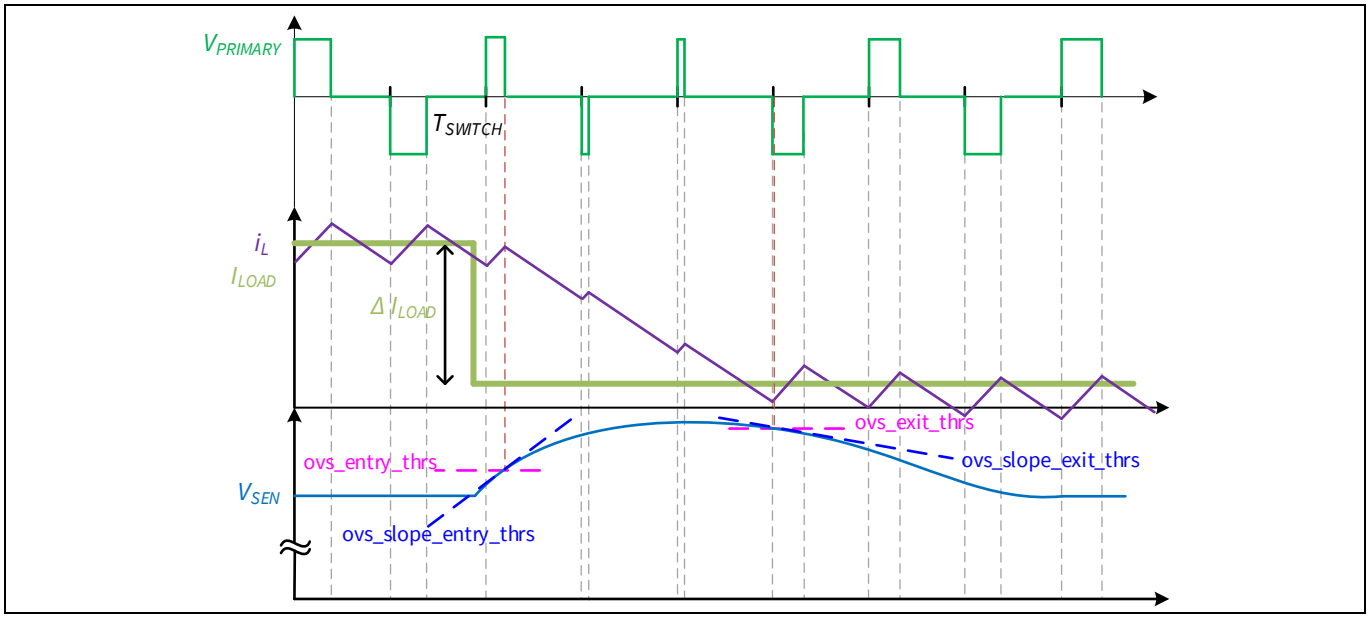

Figure 64 Fast-overshoot response waveform

The fast-overshoot response mode is enabled by setting the register **pid_ovs_entry_thrs** to any non-zero value. While programming this register, it should be noted that:

- At the system boot the FW reads and saves the setting of **pid_ovs_entry_thrs** and then sets it to 0
- On converter enable, the saved setting is restored to the register upon reaching the initial target voltage
- On converter disable it is reset again to 0

The value does not need to be stored in OTP to be effective, and like most other registers changes while regulating are permitted (e.g., to fine-tune response). The likelihood of triggering fast-transient response due to the switching noise at the VSEN pin can be reduced in a similar way as discussed in the previous section by using the LPF. The filter BW settings were summarized in Table 45.

Refer to the Application Guide – Digital power controller XDPP1100 for additional information regarding system tuning with respect to fast-overshoot response.

7.6 PWM interrupts

Both ramp generators (ramp0 and ramp1) are capable of setting two interrupts (IRQs). The following registers define the interrupts triggered on timing markers t1 and t2:

- **rampX_t1_irq_sel**, for t1
- **rampX_t2_irq_sel**, for t2

The X denotes either 0 or 1. The programming of these registers is defined in Table 46.

Table 46 Ramp generator interrupt programming

[2:0]	rampX_t1_irq_sel	rampX_t2_irq_sel
0	Disabled	Disabled
1	t1	t2
2	t1 even	t2 even
3	t1 odd	t2 odd
4 to 7	Set by rampX_irq_phase	V_{RECT} falling edge

In addition to setting an interrupt on t1 and t2:

- Register **rampX_t1_irq_sel** may also select an arbitrary phase location within the ramp for an interrupt. The phase location is defined by the register **rampX_irq_phase**. Its value is defined according to Equation (7.14)
- Register **rampX_t2_irq_sel** may also select the falling edge of V_{RECT} to generate the interrupt

$$IRQ\ Phase = \left(\frac{rampX_irq_phase[3:0]}{16}\right) * 360° \quad (7.14)$$

The frequency of the interrupts can be controlled by the register **rampX_irq_rate**. It is possible to program the interrupts to occur from once per switching period to once every 64 switching periods, as shown in Table 47.

Table 47 Ramp generator interrupt rate selection

rampX_irq_rate	Interrupt frequency
0	Every T_{switch}
1	Every 2 T_{switch}
2	Every 4 T_{switch}
3	Every 8 T_{switch}
4	Every 16 T_{switch}
5	Every 32 T_{switch}
6, 7	Every 64 T_{switch}

It should be noted that **rampX_irq_rate** applies to both the t1 and t2 interrupts if they are both enabled.

7.7 PWM registers

The relevant PWM registers and their descriptions are provided in Table 48.

Table 48 PWM-related registers and their descriptions

Peripheral	Field name	Access	Address	Bits	Description
pwm	ramp0_pid_sel	RW	7000_2C00h	[0]	PID source select for PWM ramp0. PID0 receives its error input from the VSEN input. PID1 receives its error input from the BVSEN input. Generally, ramp0 should always use PID0. 0: PID0 (typical setting) 1: PID1

Peripheral	Field name	Access	Address	Bits	Description
pwm	ramp0_sync_sel	RW	7000_2C00h	[1]	Sync select for ramp0. Should always be set to 1
pwm	ramp0_m_flavor	RW	7000_2C00h	[3:2]	Edge modulation type for ramp0. 0: DE 1: LE 2 to 3: TE
pwm	ramp0_half_mode	RW	7000_2C00h	[4]	Half-mode enable for ramp0. When half-mode is enabled, the maximum ramp count is equal to half of T_{switch}, otherwise it is equal to T_{switch}. Half-mode should be enabled for bridge-type primary-side topologies and disabled otherwise. 0: Half-mode disabled (non-bridge topology) 1: Half-mode enabled (bridge topology)
pwm	ramp0_min_pw_state	RW	7000_2C00h	[5]	Selects pulse generator response when PW computed from PID duty cycle is less than ramp0_pw_min. 0: Set pulse width to 0 (i.e., blank pulse) 1: Set pulse width to ramp0_pw_min (i.e., clamp to min.)
pwm	ramp1_pid_sel	RW	7000_2C00h	[6]	PID source select for ramp1. PID0 receives its error input from the VSEN input. PID1 receives its error input from the BVSEN input. ramp1 is used on interleaved (dual-phase) or dual-loop designs. PID0 should be selected on interleaved designs due to the shared V_{OUT} sense source (VSEN) on both phases. PID1 should be selected on dual-loop designs due to the different V_{OUT} sense sources on both loops. 0: PID0 (interleaved) 1: PID1 (dual-loop)
pwm	ramp1_sync_sel	RW	7000_2C00h	[7]	Sync select for ramp1. ramp1 is used on interleaved (dual-phase) or dual-loop designs. 0: Sync to $F_{switch1}$ without external sync, select for dual-loop

Peripheral	Field name	Access	Address	Bits	Description
					topology with $F_{switch1}$ = $F_{switch0}$ or a Loop 1 only topology 1: Sync to $F_{switch0}$ or external sync, select for dual-loop topology with $F_{switch1}$ = $F_{switch0}$ or single-loop interleaved phase
pwm	ramp1_m_flavor	RW	7000_2C00h	[9:8]	Edge modulation type for ramp1. ramp1 is used on interleaved (dual-phase) or dual-loop designs. In an interleaved design, this register should match the setting of ramp0_m_flavor. 0: DE 1: LE 2 to 3: TE
pwm	ramp1_half_mode	RW	7000_2C00h	[10]	Half-mode enable for ramp1. When half-mode is enabled, the maximum ramp count is equal to half of T_{switch}, otherwise it is equal to T_{switch}. Half-mode should be enabled for bridge type primary-side topologies and disabled otherwise. 0: Half-mode disabled (non-bridge topology) 1: Half-mode enabled (bridge topology)
pwm	ramp1_min_pw_state	RW	7000_2C00h	[11]	Selects pulse generator response when PW computed from PID duty cycle is less than ramp1_pw_min. 0: Set pulse width to 0 (i.e., blank pulse) 1: Set pulse width to ramp1_pw_min (i.e., clamp to min.)
pwm	mode_control_loop0	RW	7000_2C00h	[13:12]	Loop 0 control mode select 0: VMC 1: PCMC on secondary 2: PCMC on primary 3: Reserved
pwm	mode_control_loop1	RW	7000_2C00h	[15:14]	Loop 1 control mode select 0: VMC 1: PCMC on secondary 2: PCMC on primary 3: Reserved

Peripheral	Field name	Access	Address	Bits	Description
pwm	ramp0_dutyc_lock	RW	7000_2C00h	[16]	ramp0 duty-cycle lock enable. When enabled on a bridge topology (ramp0_half_mode = 1), the odd half-cycle duty cycle is locked to the even half-cycle duty cycle prior to applying any flux balance correction. Duty-cycle lock is required when using flux balancing but may also be used without flux balance. 0: Duty lock disabled 1: Duty lock enabled
pwm	ramp1_dutyc_lock	RW	7000_2C00h	[17]	ramp1 duty-cycle lock enable. When enabled on a bridge topology (ramp1_half_mode = 1), the odd half-cycle duty cycle is locked to the even half-cycle duty cycle prior to applying any flux balance correction. Duty-cycle lock is required when using flux balancing but may also be used without flux balance. 0: Duty lock disabled 1: Duty lock enabled
pwm	compensation_slope	RW	7000_2C00h	[19:18]	Defines the compensation ramp slope when PCMC selected as modulation type by mode_control_loop0 or mode_control_loop1. This single register applies to both loops in the case of a dual-loop system. 0: V_{OUT}/L 1: $V_{OUT}/2L$ 2: $V_{OUT}/4L$ 3: Reserved
pwm	pwm1_fall_sel	RW	7000_2C04h	[2:0]	Topology-driven PWM1 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using dual-edge modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for

Peripheral	Field name	Access	Address	Bits	Description
					bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm1_rise_sel	RW	7000_2C04h	[6:3]	Topology-driven PWM1 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using dual-edge modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm2_fall_sel	RW	7000_2C04h	[9:7]	Topology-driven PWM2 falling edge select. t1 and t2 refer to the modulated edges created by the

Peripheral	Field name	Access	Address	Bits	Description
					ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using dual0edge modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm2_rise_sel	RW	7000_2C04h	[13:10]	Topology-driven PWM2 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle

Peripheral	Field name	Access	Address	Bits	Description
					3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm3_fall_sel	RW	7000_2C04h	[16:14]	Topology-driven PWM3 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm3_rise_sel	RW	7000_2C04h	[20:17]	Topology-driven PWM3 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to

Peripheral	Field name	Access	Address	Bits	Description
					distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm4_fall_sel	RW	7000_2C04h	[23:21]	Topology-driven PWM4 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm4_rise_sel	RW	7000_2C04h	[27:24]	Topology-driven PWM4 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or

Peripheral	Field name	Access	Address	Bits	Description
					0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm5_fall_sel	RW	7000_2C08h	[2:0]	Topology-driven PWM5 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle

Peripheral	Field name	Access	Address	Bits	Description
					6: t1 delay 7: t2 delay
pwm	pwm5_rise_sel	RW	7000_2C08h	[6:3]	Topology-driven PWM5 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm6_fall_sel	RW	7000_2C08h	[9:7]	Topology-driven PWM6 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and

Peripheral	Field name	Access	Address	Bits	Description
					even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm6_rise_sel	RW	7000_2C08h	[13:10]	Topology-driven PWM6 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm7_fall_sel	RW	7000_2C08h	[16:14]	Topology-driven PWM7 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is

Peripheral	Field name	Access	Address	Bits	Description
					modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm7_rise_sel	RW	7000_2C08h	[20:17]	Topology-driven PWM7 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle

Peripheral	Field name	Access	Address	Bits	Description
					6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm8_fall_sel	RW	7000_2C08h	[23:21]	Topology-driven PWM8 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm8_rise_sel	RW	7000_2C08h	[27:24]	Topology-driven PWM8 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high

Peripheral	Field name	Access	Address	Bits	Description
					after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm9_fall_sel	RW	7000_2C0Ch	[2:0]	Topology-driven PWM9 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm9_rise_sel	RW	7000_2C0Ch	[6:3]	Topology-driven PWM9 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1

Peripheral	Field name	Access	Address	Bits	Description
					and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm10_fall_sel	RW	7000_2C0Ch	[9:7]	Topology-driven PWM10 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay

Peripheral	Field name	Access	Address	Bits	Description
pwm	pwm10_rise_sel	RW	7000_2C0Ch	[13:10]	Topology-driven PWM10 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm11_fall_sel	RW	7000_2C0Ch	[16:14]	Topology-driven PWM11 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to

Peripheral	Field name	Access	Address	Bits	Description
					distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm11_rise_sel	RW	7000_2C0Ch	[20:17]	Topology-driven PWM11 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay 8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm12_fall_sel	RW	7000_2C0Ch	[23:21]	Topology-driven PWM12 falling edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or

Peripheral	Field name	Access	Address	Bits	Description
					0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay
pwm	pwm12_rise_sel	RW	7000_2C0Ch	[27:24]	Topology-driven PWM12 rising edge select. t1 and t2 refer to the modulated edges created by the ramp. When using TE modulation, t1 is fixed at time 0 and t2 is modulated. When using LE modulation, t2 is fixed at T_{switch} (or 0.5 * T_{switch} and T_{switch} for bridge topologies) and t1 is modulated. When using DE modulation, t1 and t2 are centered around 0.5 * T_{switch} (or 0.25 * T_{switch} and 0.75 * T_{switch} for bridge topologies) and both are modulated. Odd and even cycle designations are for use with bridge topologies to distinguish between half-cycles. The VRSEN and BVRSEN options allow a PWM output to be set high after detection of the falling transition of the rectification voltage. 0: t1 1: t2 2: t1 even cycle 3: t2 even cycle 4: t1 odd cycle 5: t2 odd cycle 6: t1 delay 7: t2 delay

Peripheral	Field name	Access	Address	Bits	Description
					8: VRSEN neg. edge 9 to 15: BVRSEN neg. edge
pwm	pwm1_loop_map	RW	7000_2C10h	[1:0]	Defines the loop and phase mapping of the PWM1 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm2_loop_map	RW	7000_2C10h	[3:2]	Defines the loop and phase mapping of the PWM2 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm3_loop_map	RW	7000_2C10h	[5:4]	Defines the loop and phase mapping of the PWM3 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm4_loop_map	RW	7000_2C10h	[7:6]	Defines the loop and phase mapping of the PWM4 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm5_loop_map	RW	7000_2C10h	[9:8]	Defines the loop and phase mapping of the PWM5 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm6_loop_map	RW	7000_2C10h	[11:10]	Defines the loop and phase mapping of the PWM6 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm7_loop_map	RW	7000_2C10h	[13:12]	Defines the loop and phase mapping of the PWM7 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm8_loop_map	RW	7000_2C10h	[15:14]	Defines the loop and phase mapping of the PWM8 output.

Peripheral	Field name	Access	Address	Bits	Description
					0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm9_loop_map	RW	7000_2C10h	[17:16]	Defines the loop and phase mapping of the PWM9 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm10_loop_map	RW	7000_2C10h	[19:18]	Defines the loop and phase mapping of the PWM10 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm11_loop_map	RW	7000_2C10h	[21:20]	Defines the loop and phase mapping of the PWM11 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	pwm12_loop_map	RW	7000_2C10h	[23:22]	Defines the loop and phase mapping of the PWM12 output. 0: PWM not in use 1: Loop 0, phase 0 2: Loop 1, phase 0 3: Loop 0, phase 1
pwm	ramp0_dc_max_nom	RW	7000_2C14h	[7:0]	ramp0 nominal max. duty cycle scaled by the rectification voltage (V_{RECT}). The PMBus command MAX_DUTY defines a fixed max. duty-cycle limit. This parameter defines a variable max. duty-cycle limit that scales with the rectification voltage. The lower of the two limits will be enforced. This max. duty cycle limit is scaled with respect to the V_{ref} reference defined by parameter pid0.pid_vrect_ref as shown below. Max. duty = ramp0_dc_max_nom * pid_vrect_ref/V_{RECT} A setting of 0 will disable the scaled max. duty-cycle limit and only MAX_DUTY will apply.

Peripheral	Field name	Access	Address	Bits	Description
					LSB = 0.5 percent, range = 0.0 to 99.5 percent
pwm	ramp1_dc_max_nom	RW	7000_2C14h	[15:8]	ramp1 nominal max. duty cycle scaled by the rectification voltage (V_{RECT}). The PMBus command MAX_DUTY defines a fixed max. duty-cycle limit. This parameter defines a variable max. duty-cycle limit that scales with the rectification voltage. The lower of the two limits will be enforced. This max. duty-cycle limit is scaled with respect to the V_{ref} reference defined by parameter pid1.pid_vrect_ref as shown below. Max. duty = ramp1_dc_max_nom * pid_vrect_ref/V_{RECT} A setting of 0 will disable the scaled max. duty-cycle limit and only MAX_DUTY will apply. LSB = 0.5 percent, range = 0.0 to 99.5 percent

pwm	lp0_ftr_vin_thresh	RW	7000_2C18h	[6:0]	Loop 0 fast-transient response (FTR) input voltage threshold. Below this threshold, the FTR PW is based on the FF duty cycle from the PID. Above this threshold, the FTR PW is reduced in proportion to V_{IN}.
					
					if (V_{IN} is less than lp0_ftr_vin_thresh)
					FTR PW = FF_Duty * (T_{switch}/2)
					else
					FTR PW = FF_Duty * (T_{switch} /2) * (lp0_ftr_vin_thresh/V_{IN})
					
					LSB = 1 V, range = 0 to 127 V

pwm	lp1_ftr_vin_thresh	RW	7000_2C18h	[13:7]	Loop 1 FTR input voltage threshold. Below this threshold, the FTR PW is based on the FF duty cycle from the PID. Above this threshold, the FTR PW is reduced in proportion to V_{IN}. if (V_{IN} is less than lp1_ftr_vin_thresh)

Peripheral	Field name	Access	Address	Bits	Description
					FTR PW = FF_Duty * (T_{switch} /2) else FTR PW = FF_Duty * (T_{switch} /2) * (lp1_ftr_vin_thresh/V_{IN}) LSB = 1 V, range = 0 to 127 V
pwm	ramp0_phase	RW	7000_2C1Ch	[7:0]	ramp0 phase alignment with respect to sync signal selected with ramp0_sync_sel. **Computed by FW from PMBus command as follows:** ramp0_phase = 2^8 * PAGE0.INTERLEAVE[3:0] / PAGE0.INTERLEAVE[7:4] LSB = 1.40625 degrees, range = 0.0 to 358.59375 degrees
pwm	ramp1_phase	RW	7000_2C20h	[7:0]	ramp1 phase alignment with respect to sync signal selected with ramp1_sync_sel. **Computed by FW from PMBus command as follows:** ramp1_phase = 2^8 * PAGE1.INTERLEAVE[3:0]/ PAGE1.INTERLEAVE[7:4] LSB = 1.40625 degrees, range = 0.0 to 358.59375 degrees
pwm	pwm_force_hi	RW	7000_2C24h	[11:0]	Force PWM output high, [0] corresponds to PWM1, [11] corresponds to PWM12. This register has lower priority than pwm_force_lo.
pwm	pwm_force_lo	RW	7000_2C24h	[23:12]	Force PWM output low, [0] corresponds to PWM1, [11] corresponds to PWM12. This register has higher priority than pwm_force_hi.
pwm	pwm_on		7000_2C28h	[11:0]	PWM channel enabled for pulse generation when corresponding bit position high, [0] corresponds to PWM1, [11] corresponds to PWM12. **Note:** Intended to be driven by FW only
pwm	tswitch0	RW	7000_2C2Ch	[8:0]	PWM ramp0 switching period. This register defines the switching period of ramp0 when ramp0 is not synced to an external sync

Peripheral	Field name	Access	Address	Bits	Description
					signal. **Computed by FW from PMBus command as follows:** $T_{switch0}$ = 1/ (FREQUENCY_SWITCH(Hz) * 20 ns) LSB = 20 ns, range = 0.0 to 10.22 µs
pwm	ramp0_dc_max	RW	7000_2C30h	[7:0]	PWM ramp0 max. duty cycle (fixed). See ramp0_dc_max_nom description for scaled max. duty cycle information. **Computed by FW from PMBus command as follows:** ramp0_dc_max = MAX_DUTY LSB = 0.5 percent, range = 0.0 to 99.5 percent
pwm	ramp0_pw_min	RW	7000_2C30h	[15:8]	PWM ramp0 min. pulse width. When the duty cycle from the PID is less than this value the duty cycle input to the ramp is either clamped to this value or set to 0 as determined by the setting of ramp0_pw_min_state. **Computed by FW from PMBus as follows:** ramp0_pw_min = MFR_MIN_PW LSB = 5 ns, range = 0 to 1275 ns
pwm	ramp0_force_duty	RW	7000_2C34h	[7:0]	This forced duty-cycle value overrides the ramp0 duty-cycle input when selected by ramp0_force_duty_en. Since this force is applied at the ramp input, upstream adjustments to the duty cycle such as current balance in an interleaved (multiphase) design are overwritten. To not override the current balance adjustment, use pid_force_duty, which is applied prior to the current balance adjustments. LSB = 0.3906 percent, range = 0.0 to 99.6094 percent
pwm	ramp0_force_duty_en	RW	7000_2C34h	[8]	PWM ramp0 forced duty-cycle select. 0: Use PID computed duty cycle 1: Use ramp0_force_duty

Peripheral	Field name	Access	Address	Bits	Description
pwm	ramp0_force_t1	RW	7000_2C38h	[10:0]	PWM ramp0 forced t1 setting selected by ramp0_force_t1_en. t1 is the time of the first PWM “edge” in a ramp cycle. In a TE modulation scheme t1 is fixed at 0. In the leading and DE modulation schemes t1 is determined based on the duty cycle. LSB = 5 ns, range = 0 to 10235 ns
pwm	ramp0_force_t1_en	RW	7000_2C38h	[11]	PWM ramp0 t1 force enable. 0: t1 determined by modulation scheme and duty cycle 1: t1 set by ramp0_force_t1
pwm	ramp0_force_t2	RW	7000_2C3Ch	[10:0]	PWM ramp0 forced t2 setting selected by ramp0_force_t2_en. t2 is the time of the second PWM “edge” in a ramp cycle. In a LE modulation scheme t2 is fixed at T_{switch} (or $T_{switch}/2$ in bridge topologies). In the trailing and DE modulation schemes t2 is determined based on the duty cycle. LSB = 5 ns, range = 0 to 10235 ns
pwm	ramp0_force_t2_en	RW	7000_2C3Ch	[11]	PWM ramp0 t2 force enable. 0: t2 determined by modulation scheme and duty cycle 1: t2 set by ramp0_force_t2
pwm	tswitch1	RW	7000_2C40h	[8:0]	PWM ramp1 switching period. This register defines the switching period of ramp1 when ramp1 is not synced to an external sync signal. **Computed by FW from PMBus command as follows:** $T_{switch1}$ = 1/ (FREQUENCY_SWITCH(Hz) * 20ns) LSB = 20 ns, range = 0.0 to 10.22 µs
pwm	ramp1_dc_max	RW	7000_2C44h	[7:0]	PWM ramp1 max. duty cycle (fixed). See ramp1_dc_max_nom description for scaled max. duty-cycle information. **Computed by FW from PMBus command as follows:** ramp1_dc_max = MAX_DUTY

Peripheral	Field name	Access	Address	Bits	Description
					LSB = 0.5 percent, range = 0.0 to 99.5 percent
pwm	ramp1_pw_min	RW	7000_2C44h	[15:8]	PWM ramp1 min. pulse width. When the duty cycle from the PID is less than this value the duty-cycle input to the ramp is either clamped to this value or set to 0 as determined by the setting of ramp0_pw_min_state. **Computed by FW from PMBus command as follows:** ramp1_pw_min = MFR_MIN_PW LSB = 5 ns, range = 0 to 1275 ns
pwm	ramp1_force_duty	RW	7000_2C48h	[7:0]	This forced duty-cycle value overrides the ramp1 duty-cycle input when selected by ramp1_force_duty_en. Since this force is applied at the ramp input, upstream adjustments to the duty cycle such as current balance in an interleaved (multiphase) design are overwritten. To not override the current balance adjustment, use pid_force_duty, which is applied prior to the current balance adjustments. LSB = 0.3906 percent, range = 0.0 to 99.6094 percent
pwm	ramp1_force_duty_en	RW	7000_2C48h	[8]	PWM ramp1 forced duty-cycle select. 0: Use PID computed duty cycle 1: Use ramp1_force_duty
pwm	ramp1_force_t1	RW	7000_2C4Ch	[10:0]	PWM ramp1 forced t1 setting selected by ramp1_force_t1_en. t1 is the time of the first PWM “edge” in a ramp cycle. In a TE modulation scheme t1 is fixed at 0. In the leading and DE modulation schemes t1 is determined based on the duty cycle. LSB = 5 ns, range = 0 to 10235 ns
pwm	ramp1_force_t1_en	RW	7000_2C4Ch	[11]	PWM ramp1 t1 force enable. 0: t1 determined by modulation scheme and duty cycle 1: t1 set by ramp1_force_t1

Peripheral	Field name	Access	Address	Bits	Description
pwm	ramp1_force_t2	RW	7000_2C50h	[10:0]	PWM ramp1 forced t2 setting selected by ramp1_force_t2_en. t2 is the time of the second PWM “edge” in a ramp cycle. In a LE modulation scheme t2 is fixed at T_{switch} (or T_{switch} /2 in bridge topologies). In the trailing and DE modulation schemes t2 is determined based on the duty cycle. LSB = 5 ns, range = 0 to 10235 ns
pwm	ramp1_force_t2_en	RW	7000_2C50h	[11]	PWM ramp1 t2 force enable. 0: t2 determined by modulation scheme and duty cycle 1: t2 set by ramp1_force_t2
pwm	pwm1_dr	RW	7000_2C54h	[7:0]	PWM1 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm1_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm1_dr[7:0] = PWM_DEADTIME[15:8] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm1_df	RW	7000_2C54h	[15:8]	PWM1 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm1_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm1_df[7:0] =

Peripheral	Field name	Access	Address	Bits	Description
					PWM_DEADTIME[7:0] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm2_dr	RW	7000_2C58h	[7:0]	PWM2 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm2_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm2_dr[7:0] = PWM_DEADTIME[31:24] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm2_df	RW	7000_2C58h	[15:8]	PWM2 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm2_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm2_df[7:0] = PWM_DEADTIME[23:16] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm3_dr	RW	7000_2C5Ch	[7:0]	PWM3 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm3_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus**

Peripheral	Field name	Access	Address	Bits	Description
					command as follows: pwm3_dr[7:0] = PWM_DEADTIME[47:40] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm3_df	RW	7000_2C5Ch	[15:8]	PWM3 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm3_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm3_df[7:0] = PWM_DEADTIME[39:32] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm4_dr	RW	7000_2C60h	[7:0]	PWM4 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm4_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm4_dr[7:0] = PWM_DEADTIME[63:56] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm4_df	RW	7000_2C60h	[15:8]	PWM4 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm4_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead-time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written.

Peripheral	Field name	Access	Address	Bits	Description
					Computed by FW from PMBus command as follows: pwm4_df[7:0] = PWM_DEADTIME[55:48] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm5_dr	RW	7000_2C64h	[7:0]	PWM5 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm5_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm5_dr[7:0] = PWM_DEADTIME[79:72] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm5_df	RW	7000_2C64h	[15:8]	PWM5 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm5_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm5_df[7:0] = PWM_DEADTIME[71:64] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm6_dr	RW	7000_2C68h	[7:0]	PWM6 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm6_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h

Peripheral	Field name	Access	Address	Bits	Description
					(pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm6_dr[7:0] = PWM_DEADTIME[95:88] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm6_df	RW	7000_2C68h	[15:8]	PWM6 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm6_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm6_df[7:0] = PWM_DEADTIME[87:80] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm7_dr	RW	7000_2C6Ch	[7:0]	PWM7 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm7_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm7_dr[7:0] = PWM_DEADTIME[111:104] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm7_df	RW	7000_2C6Ch	[15:8]	PWM7 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm7_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only

Peripheral	Field name	Access	Address	Bits	Description
					becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm7_df[7:0] = PWM_DEADTIME[103:96] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm8_dr	RW	7000_2C70h	[7:0]	PWM8 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm8_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm8_dr[7:0] = PWM_DEADTIME[127:120] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm8_df	RW	7000_2C70h	[15:8]	PWM8 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm8_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm8_df[7:0] = PWM_DEADTIME[119:112] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm9_dr	RW	7000_2C74h	[7:0]	PWM9 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm9_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times

Peripheral	Field name	Access	Address	Bits	Description
					simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm9_dr[7:0] = PWM_DEADTIME[143:136] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm9_df	RW	7000_2C74h	[15:8]	PWM9 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm9_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm9_df[7:0] = PWM_DEADTIME[135:128] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm10_dr	RW	7000_2C78h	[7:0]	PWM10 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm10_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm10_dr[7:0] = PWM_DEADTIME[159:152] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm10_df	RW	7000_2C78h	[15:8]	PWM10 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm10_fall_sel. In order to

Peripheral	Field name	Access	Address	Bits	Description
					synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm10_df[7:0] = PWM_DEADTIME[151:144] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm11_dr	RW	7000_2C7Ch	[7:0]	PWM11 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by pwm11_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm11_dr[7:0] = PWM_DEADTIME[175:168] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm11_df	RW	7000_2C7Ch	[15:8]	PWM11 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm11_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm11_df[7:0] = PWM_DEADTIME[167:160] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm12_dr	RW	7000_2C80h	[7:0]	PWM12 rising edge delay (dead) time from t1 or t2. Mapping of the rising edge to t1 or t2 defined by

Peripheral	Field name	Access	Address	Bits	Description
					pwm12_rise_sel. In order to synchronously update all pwmX_dr and pwmX_df times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm12_dr[7:0] = PWM_DEADTIME[191:184] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	pwm12_df	RW	7000_2C80h	[15:8]	PWM12 falling edge delay (dead) time from t1 or t2. Mapping of the falling edge to t1 or t2 defined by pwm12_fall_sel. In order to synchronously update all dead times simultaneously, an update to any dead time register only becomes effective after 7000_2C80h (pwm12_dr, pwm12_df) is written. **Computed by FW from PMBus command as follows:** pwm12_df[7:0] = PWM_DEADTIME[183:176] LSB = 1.25 ns, range = 0.0 to 318.75 ns
pwm	ramp0_dutyc_force_status	R	7000_2C84h	[0]	This register bit is set to 1 when a new duty-cycle force has been applied on ramp0. It is cleared to 0 when either ramp0_force_duty_en or ramp0_force_duty changes.
pwm	ramp1_dutyc_force_status	R	7000_2C84h	[1]	This register bit is set to 1 when a new duty-cycle force has been applied on ramp1. It is cleared to 0 when either ramp1_force_duty_en or ramp1_force_duty changes.
pwm	ramp0_t1_irq_sel	RW	7000_2C8Ch	[2:0]	PWM ramp0 t1 IRQ source select. 0: t1 IRQ disabled 1: t1 2: t1 even 3: t1 odd

Peripheral	Field name	Access	Address	Bits	Description
					4 to 7: IRQ set by ramp0_irq_phase[3:0]
pwm	ramp0_t2_irq_sel	RW	7000_2C8Ch	[5:3]	PWM ramp0 t2 IRQ source select. 0: t2 IRQ disabled 1: t2 2: t2 even 3: t2 odd 4 to 7: V_{RECT} falling edge
pwm	ramp0_irq_phase	RW	7000_2C8Ch	[9:6]	Defines the t1 interrupt with respect to the internal PWM ramp phase when ramp0_t1__irq_sel=4. IRQ phase = (ramp0_irq_phase[3:0]/ 16) * 360 degrees LSB = 22.5 degrees, range = 0.0 to 337.5 degrees
pwm	ramp0_irq_rate	RW	7000_2C8Ch	[12:10]	Defines frequency of IRQs selected by ramp0_t1_irq_sel and ramp0_t2_irq_sel. 0: Every T_{switch} 1: Every 2 T_{switch} 2: Every 4 T_{switch} 3: Every 8 T_{switch} 4: Every 16 T_{switch} 5: Every 32 T_{switch} 6: Every 64 T_{switch} 7: Every 64 T_{switch}
pwm	ramp1_t1_irq_sel	RW	7000_2C8Ch	[15:13]	PWM ramp1 t1 IRQ source select. 0: t1 IRQ disabled 1: t1 2: t1 even 3: t1 odd 4 to 7: IRQ set by ramp1_irq_phase[3:0]
pwm	ramp1_t2_irq_sel	RW	7000_2C8Ch	[18:16]	PWM ramp1 t2 IRQ source select. 0: t2 IRQ disabled 1: t2 2: t2 even 3: t2 odd 4 to 7: V_{RECT} falling edge
pwm	ramp1_irq_phase	RW	7000_2C8Ch	[22:19]	Defines the t1 interrupt with respect to the internal PWM ramp phase when ramp1_t1__irq_sel=4. IRQ phase = (ramp1_irq_phase[3:0]/

Peripheral	Field name	Access	Address	Bits	Description
					16) * 360 degrees LSB = 22.5 degrees, range = 0.0 to 337.5 degrees
pwm	ramp1_irq_rate	RW	7000_2C8Ch	[25:23]	Defines frequency of interrupts selected by ramp1_t1_irq_sel and ramp1_t2_irq_sel. 0: Every T_{switch} 1: Every 2 T_{switch} 2: Every 4 T_{switch} 3: Every 8 T_{switch} 4: Every 16 T_{switch} 5: Every 32 T_{switch} 6: Every 64 T_{switch} 7: Every 64 T_{switch}
pwm	pwm_in_en_mask	RW	7000_2C90h	[11:0]	When bit X is low, enables input buffer on PWM[X+1] input pin. The input buffer can then be used for polling the state of the PWM output via the common.io_inputs status register. Setting low also asserts interrupt PWM_IN_IRQ when the enabled PWM inputs are high.
pid	pid_vrect_ref	RW	7000_1C00h (pid0) 7000_2000h (pid1)	[27:20]	Reference V_{RECT} voltage for PID coefficient scaling and variable max. duty-cycle limit. PID coefficients are scaled with V_{RECT} to maintain a constant loop gain. This parameter defines the reference V_{RECT} voltage at which the gain scale is 1.0. This parameter should be set to the expected nominal V_{RECT} voltage and it should be set before optimization of PID coefficients K_P, K_I and K_D. **Example:** V_{IN_nom} = 48 V, FB topology, N_{turn} = 3 pid_vrect_ref = V_{RECT_nom} = 48 V/3 = 16 V LSB = 0.32 V, range = 0.0 to 81.6 V
pid	pid_verr_exit_thrs	RW	7000_1C0Ch (pid0) 7000_200Ch (pid1)	[6:0]	FTR mode error voltage (V_{err}) exit threshold where the error voltage is defined as V_{err} = (target voltage - sense voltage) When (V_{err} is less than

Peripheral	Field name	Access	Address	Bits	Description
					pid_verr_exit_thrs) AND (V_{err} slope < pid_verr_slope_exit_thrs) the control loop exits FTR mode. **Note:** This threshold is always positive, indicating that the controller exits FTR mode prior to the sensed voltage overshooting the target. LSB = 1.25 mV, range = 0.0 to 158.75 mV at VSEN
pid	pid_verr_slope_exit_thrs	RW	7000_1C0Ch (pid0) 7000_200Ch (pid1)	[13:7]	FTR mode error voltage (V_{err}) slope exit threshold where the error voltage is defined as V_{err} = (target voltage - sense voltage) When (V_{err} is less than pid_verr_exit_thrs) AND (V_{err} slope < pid_verr_slope_exit_thrs) the control loop exits FTR mode. **Notes:** 1. This threshold is always negative, indicating that the controller does not exit FTR mode until the sensed voltage has hit its maximum undershoot and is approaching the target voltage 2. There is a -1.25 mV offset (i.e., code 0 = -1.25 mV, 1 = -2.5 mV ...) LSB = -1.25 mV/clk, range = -1.25 to -160 mV/clk at VSEN
pid	pid_verr_entry_thrs	RW	7000_1C0Ch (pid0) 7000_200Ch (pid1)	[20:14]	FTR mode V_{err} entry threshold where the error voltage is defined as V_{err} = (target voltage - sense voltage) When (V_{err} is greater than pid_verr_entry_thrs) AND (V_{err} slope is greater than pid_verr_slope_entry_thrs) the control loop enters FTR mode. **Notes:** 1. This threshold is always positive, indicating that the controller enters FTR mode only when the sensed voltage is below the target 2. Setting pid_verr_entry_thrs = 0 disables FTR mode.

Peripheral	Field name	Access	Address	Bits	Description
					LSB = 1.25 mV, range = 0.0 to 158.75 mV at VSEN
pid	pid_verr_slope_entry_thrs	RW	7000_1C0Ch (pid0) 7000_200Ch (pid1)	[27:21]	FTR mode V_{err} slope entry threshold where the error voltage is defined as V_{err} = (target voltage - sense voltage) When (V_{err} is greater than pid_verr_entry_thrs) AND (verr slope > pid_verr_slope_entry_thrs) the control loop enters FTR mode. **Note:** This threshold is always positive, indicating that the controller enters FTR mode as the sensed voltage is decreasing toward maximum undershoot. LSB = 1.25 mV/clk, range = 0.0 to 158.75 mV/clk at VSEN
pid	pid_ftr_lpf	RW	7000_1C0Ch (pid0) 7000_200Ch (pid1)	[30:28]	FTR mode LPF BW applied to the V_{err} prior to comparing against the FTR entry and exit thresholds. 0: 1 MHz 1: 2 MHz 2: 4 MHz 3: 8 MHz 4 to 7: Filter bypassed
pid	pid_ovs_exit_thrs	RW	7000_1C10h (pid0) 7000_2010h (pid1)	[6:0]	Overshoot transient (OVS) mode V_{err} exit threshold where the error voltage is defined as V_{err} = (target voltage - sense voltage) When (V_{err} is greater than pid_ovs_exit_thrs) AND (V_{err} slope is greater than pid_ovs_slope_exit_thrs) the control loop exits OVS mode. **Notes:** 1. This threshold is always negative, indicating that the controller exits OVS mode prior to the sensed voltage undershooting the target 2. There is a -1.25 mV offset (i.e., code 0 = -1.25 mV, 1 = -2.5 mV ...)

Peripheral	Field name	Access	Address	Bits	Description
					LSB = -1.25 mV, range = -1.25 to -160 mV at VSEN
pid	pid_ovs_slope_exit_thrs	RW	7000_1C10h (pid0) 7000_2010h (pid1)	[13:7]	OVS mode error voltage (V_{err}) slope exit threshold where the error voltage is defined as V_{err} = (target voltage - sense voltage) When (V_{err} is greater than pid_ovs_exit_thrs) AND (V_{err} slope is greater than pid_ovs_slope_exit_thrs) the control loop exits OVS mode. **Note:** This threshold is always positive, indicating that the controller does not exit OVS mode until the sensed voltage has hit its maximum overshoot and is approaching the target voltage. LSB = 1.25 mV/clk, range = 0.0 to 158.75 mV/clk at VSEN
pid	pid_ovs_entry_thrs	RW	7000_1C10h (pid0) 7000_2010h (pid1)	[20:14]	OVS mode V_{err} entry threshold where the error voltage is defined as V_{err} = (target voltage - sense voltage) When (V_{err} is less than pid_ovs_entry_thrs) AND (V_{err} slope is less than pid_ovs_slope_entry_thrs) the control loop enters OVS mode. **Notes:** 1. This threshold is always negative, indicating that the controller enters OVS mode only when the sensed voltage is above the target 2. Setting pid_ovs_entry_thrs = 0 disables OVS mode 3. There is a -1.25 mV offset (i.e., code 0 = -1.25 mV, 1 = -2.5 mV ...) LSB = -1.25 mV, range = -1.25 to -160 mV at VSEN
pid	pid_ovs_slope_entry_thrs	RW	7000_1C10h (pid0) 7000_2010h (pid1)	[27:21]	OVS mode error voltage (V_{err}) slope entry threshold where the error voltage is defined as V_{err} = (target voltage - sense

Peripheral	Field name	Access	Address	Bits	Description
					voltage) When (V_{err} is less than pid_ovs_entry_thrs) AND (V_{err} slope is less than pid_ovs_slope_entry_thrs) the control loop enters OVS mode. **Notes:** 1. This threshold is always negative, indicating that the controller enters OVS mode as the sensed voltage is increasing toward maximum overshoot 2. There is a -1.25 mV offset (i.e., code 0 = -1.25 mV, 1 = -2.5 mV ...) LSB = 1.25 mV/clk, range = 0.0 to 158.75 mV/clk at VSEN
pid	pid_burst_mode_err_thr	RW	7000_1C14h (pid0) 7000_2014h (pid1)	[3:0]	BM V_{err} threshold where the V_{err} is defined as V_{err} = (target voltage - sense voltage) When the controller is in BM (V_{err} is less than pid_burst_mode_err_thr), it will trigger the start of a new burst sequence. **Note:** This threshold is always positive, indicating that the controller triggers the start of a new burst sequence at or below the target voltage. LSB = 1.25 mV, range = 0.0 to 18.75 mV at VSEN
pid	pid_burst_mode_ith	RW	7000_1C14h (pid0) 7000_2014h (pid1)	[9:4]	BM entry current threshold. When BM is enabled (POWER_MODE[7:0]=0x00), the controller will enter BM when the sensed current drops below pid_burst_mode_ith. LSB = (Q_{ADC}/2) where Q_{ADC} is the value of MFR_IOUT_APC in amps, range = 0.0 to 31.5 Q_{ADC}
pid	pid_burst_reps	RW	7000_1C14h (pid0) 7000_2014h (pid1)	[11:10]	BM cycle count. In BM, 1 cycle corresponds to 1 even half-cycle pulse followed by 1 odd half-cycle pulse. This register defines the number of burst cycles in each

Peripheral	Field name	Access	Address	Bits	Description
					burst event. A higher cycle count can be used to increase the inductor peak current in a burst event, which will increase the time between burst events at a given load current. 0: 1 cycle 1: 2 cycles 2: 4 cycles 3: 8 cycles
common	sync_deglitch_en	RW	7000_3008h	[28]	Deglitch enable for digital sync function when used as an input. 0: Sync input deglitch disabled 1: Sync input deglitch enabled
common	sync_dir_out	RW	7000_3014h	[13]	Defines direction of pin mapped to PWM ramp sync feature. 0: Sync mapped pin is input 1: Sync mapped pin is output
common	sync_fly_period	R	7000_30A8h	[10:0]	Current sync flywheel period, will match incoming period when lock achieved. LSB = 5 ns, range = 0 to 10.235 µs
common	sync_in_period	R	7000_30A8h	[21:11]	Measured incoming period on sync mapped input. LSB = 5 ns, range = 0 to 10.235 µs
common	sync_state	R	7000_30A8h	[23:22]	Digital sync state. 0: Using internal sync clock 1: Phase locking to external sync clock 2: Using external sync clock 3: Phase locking to internal sync clock
common	sync_in_range	R	7000_30A8h	[24]	Digital sync “in range” status. The sync input can make initial lock to an incoming signal within +/-6.25 percent of the internal clock frequency. Once locked, the sync function can maintain lock within a +/-12.5 percent deviation of the internal clock frequency. 0: Sync not locked 1: Sync locked

7.8 PWM PMBus commands

The relevant PWM-related PMBus commands are given in Table 49.

Table 49 PWM-related PMBus commands

Command name	Access	Length	Address	Bits	Description
MAX_DUTY	RW	Word	32h	[15:0]	The MAX_DUTY command sets the maximum duty cycle, in percent, of the unit's power conversion stage. Format is in LINEAR11 with exponent = -2
FREQUENCY_SWITCH	RW	Word	33h	[15:0]	The FREQUENCY_SWITCH command sets the switching frequency in kHz. Format is LINEAR11 with exponents 0 and +1 supported
POWER_MODE	RW	Byte	34h	[7:0]	Defines the power conversion mode of operation. 0: Max. efficiency (BM enabled) 3: Max. power (BM disabled)
INTERLEAVE	RW	Word	37h	[15:0]	The INTERLEAVE command is used to arrange multiple units so that their switching periods can be distributed in time. This may be used to facilitate paralleling of multiple units or to reduce AC currents injected into the power bus. On the XDPP1100 it is also used to set the relative phase alignment of the two phases in a dual-phase topology. **Dual-phase example:** PAGE0.INTERLEAVE[11:8] = 1d (or any non-zero group ID) PAGE0.INTERLEAVE[7:4] = 2d (two phases in group) PAGE0.INTERLEAVE[3:0] = 0d (position 0 in group 0 degrees) PAGE1.INTERLEAVE[11:8] = 1d (same group ID as PAGE0) PAGE1.INTERLEAVE[7:4] = 2d (two phases in group) PAGE1.INTERLEAVE[3:0] = 1d (position 1 in group 180 degrees)
FW_CONFIG_PWM	RW	Block 4 bytes	C4h	[31:0]	The FW_CONFIG_PWM command maps PWM outputs as either primary-side or secondary-side FETs assigned to the loop corresponding to the PMBus PAGE. Set bits corresponding to primary-side PWMs to 1 in pwm_on_mask. Set bits corresponding to

Command name	Access	Length	Address	Bits	Description
					secondary-side PWMs to 1 in pwm_srfet_mask. Set all other bits to 0. PMBus PAGE0 = Loop 0 (VSEN V_{OUT} sense) PMBus PAGE1 = Loop 1 (BVSEN V_{OUT} sense) [31:28]: Reserved, set to 0h [27:16]: pwm_srfet_mask[11:0], [0] = PWM1, [11] = PWM12 [15:12]: Reserved, set to 0h [11:0]: pwm_on_mask[11:0], [0] = PWM1, [11] = PWM12
PWM_DEADTIME	RW	Block 24 bytes	CFh	[191:0]	Defines rising and falling edge dead times of each PWM output when I_{OUT} is less than FW_CONFIG_DEADTIME_ADJUSTMENT_THRESHOLD or FW_CONFIG_DEADTIME_ADJUSTMENT_THRESHOLD = 000h. For each byte below, LSB = 1.25 ns, range = 0.0 to 318.75 ns. [7:0]: PWM1 falling edge dead time, pwm1_df [15:8]: PWM1 rising edge dead time, pwm1_dr [23:16]: PWM2 falling edge dead time, pwm2_df [31:24]: PWM2 rising edge dead time, pwm2_dr [39:32]: PWM3 falling edge dead time, pwm3_df [47:40]: PWM3 rising edge dead time, pwm3_dr [55:48]: PWM4 falling edge dead time, pwm4_df [63:56]: PWM4 rising edge dead time, pwm4_dr [71:64]: PWM5 falling edge dead time, pwm5_df [79:72]: PWM5 rising edge dead time, pwm5_dr [87:80]: PWM6 falling edge dead time, pwm6_df [95:88]: PWM6 rising edge dead time, pwm6_dr [103:96]: PWM7 falling edge dead time, pwm7_df

Command name	Access	Length	Address	Bits	Description
					[111:104]: PWM7 rising edge dead time, pwm7_dr [119:112]: PWM8 falling edge dead time, pwm8_df [127:120]: PWM8 rising edge dead time, pwm8_dr [135:128]: PWM9 falling edge dead time, pwm9_df [143:136]: PWM9 rising edge dead time, pwm9_dr [151:144]: PWM10 falling edge dead time, pwm10_df [159:152]: PWM10 rising edge dead time, pwm10_dr [167:160]: PWM11 falling edge dead time, pwm11_df [175:168]: PWM11 rising edge dead time, pwm11_dr [183:176]: PWM12 falling edge dead time, pwm12_df [191:184]: PWM12 rising edge dead time, pwm12_dr
PWM_DEADTIME_ADJUSTMENT	RW	Block 24 bytes	D6h	[191:0]	Defines rising and falling edge dead times of each PWM output when I_{OUT} is greater than or equal to FW_CONFIG_DEADTIME_ADJUSTMENT_THRESHOLD. For each byte below, LSB = 1.25 ns, range = 0.0 to 318.75 ns. [7:0]: PWM1 falling edge dead time, pwm1_df [15:8]: PWM1 rising edge dead time, pwm1_dr [23:16]: PWM2 falling edge dead time, pwm2_df [31:24]: PWM2 rising edge dead time, pwm2_dr [39:32]: PWM3 falling edge dead time, pwm3_df [47:40]: PWM3 rising edge dead time, pwm3_dr [55:48]: PWM4 falling edge dead time, pwm4_df [63:56]: PWM4 rising edge dead-time, pwm4_dr [71:64]: PWM5 falling edge dead-time, pwm5_df [79:72]: PWM5 rising edge dead-

Command name	Access	Length	Address	Bits	Description
					time, pwm5_dr [87:80]: PWM6 falling edge dead-time, pwm6_df [95:88]: PWM6 rising edge dead-time, pwm6_dr [103:96]: PWM7 falling edge dead-time, pwm7_df [111:104]: PWM7 rising edge dead-time, pwm7_dr [119:112]: PWM8 falling edge dead-time, pwm8_df [127:120]: PWM8 rising edge dead-time, pwm8_dr [135:128]: PWM9 falling edge dead-time, pwm9_df [143:136]: PWM9 rising edge dead-time, pwm9_dr [151:144]: PWM10 falling edge dead-time, pwm10_df [159:152]: PWM10 rising edge dead-time, pwm10_dr [167:160]: PWM11 falling edge dead-time, pwm11_df [175:168]: PWM11 rising edge dead-time, pwm11_dr [183:176]: PWM12 falling edge dead-time, pwm12_df [191:184]: PWM12 rising edge dead-time, pwm12_dr
FW_CONFIG_DEADTIME_ADJUSTMENT_THRESHOLD	RW	Word	E4h	[15:0]	Current threshold amps, above which we use PWM_DEADTIME_ADJUSTMENT instead of PWM_DEADTIME. Format is LINEAR11 with exponent = -2. Set to 0000h to disable
MFR_IOUT_APC	RW	Word	EAh	[15:0]	Current sense APC. Linear11 format with suggested exponent -8 or -9 depending on magnitude. Unit = amps
MFR_MIN_PW	RW	Byte	EBh	[7:0]	PWM ramp minimum PW, LSB = 5 ns, range = 0 to 1275 ns **Note:** PAGE 0 MFR_MIN_PW associated with PWM ramp0 and PAGE 1 MFR_MIN_PW associated with PWM ramp1 even in the case that both PWM ramps are used by Loop 0

8 Telemetry

This chapter discusses the telemetry module and its submodules and their operation. The relevant register settings and PMBus commands are also provided.

The main function of the telemetry module is to provide source selection and low-pass filtering of HW-sensed parameters (V_{OUT}, I_{OUT}, V_{IN}, I_{IN}, temperature, duty cycle and frequency) for use by the FW in generating the equivalent PMBus READ commands (READ_VOUT, READ_IOUT, etc.).

In addition, the telemetry module supports programmable interrupt generation and watermark detection. The following sections describe the various telemetry submodules.

8.1 Output voltage (V_{OUT}) telemetry

The V_{OUT} telemetry submodule block diagram is shown in Figure 65. Its input is received from the VSP module. This input is low-pass filtered at a programmable BW, determined by register **tlmX_kfp_vout** (where X = 0, 1 for Loop 0, 1). The BW programming via **tlmX_kfp_vout** is given in Table 50.

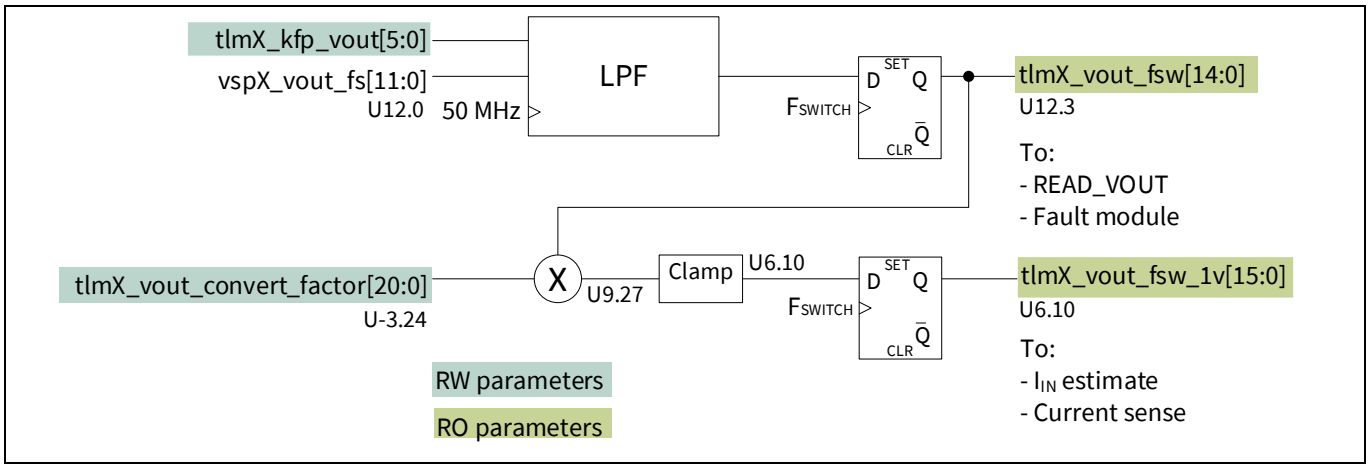

Figure 65 Output voltage telemetry block diagram

Table 50 V_{OUT} LPF BW programming

tlm_kfp_vout	kfp	kfp_real	F3db (kHz)	tlm_kfp_vout	kfp	kfp_real	F3db (kHz)
0	4	0.0001	0.97	16	64	0.0020	15.57
1	5	0.0002	1.21	17	80	0.0024	19.48
2	6	0.0002	1.46	18	96	0.0029	23.38
3	7	0.0002	1.70	19	112	0.0034	27.29
4	8	0.0002	1.94	20	128	0.0039	31.21
5	10	0.0003	2.43	21	160	0.0049	39.05
6	12	0.0004	2.92	22	192	0.0059	46.90
7	14	0.0004	3.40	23	224	0.0068	54.77
8	16	0.0005	3.89	24	256	0.0078	62.66
9	20	0.0006	4.86	25	320	0.0098	78.48
10	24	0.0007	5.83	26	384	0.0117	94.36
11	28	0.0009	6.81	27	448	0.0137	110.31
12	32	0.0010	7.78	28	512	0.0156	126.31

tlm_kfp_vout	kfp	kfp_real	F3db (kHz)	tlm_kfp_vout	kfp	kfp_real	F3db (kHz)
13	40	0.0012	9.73	29	640	0.0195	158.52
14	48	0.0015	11.67	30	768	0.0234	190.99
15	56	0.0017	13.62	31	896	0.0273	223.71

The LPF output is subsampled to the F_{switch} rate, and it is available as read-only register **tlmX_vout_fsw**. This version of V_{OUT} is in VSADC format with binary point at 1.25 mV and is used by the V_{OUT} fault functions as well as the READ_VOUT PMBus command. Register **tlmX_vout_convert_factor** converts the binary point to 1 V and creates an output tlmX_vout_vsw_1v, which is used by the current sense and I_{IN} estimate modules. The register value for **tlmX_vout_convert_factor** is computed by FW based on PMBus command VOUT_SCALE_LOOP.

8.2 Input voltage (V_{IN}) telemetry

The V_{IN} telemetry submodule block diagram is shown in Figure 66. Multiple input sources are capable of providing the input voltage telemetry. These are:

- VRSEN and BVRSEN, which can be selected using either secondary-side (V_{RECT}) or primary-side (V_{IN}) sensing
- PRISEN, which can be selected for primary-side (V_{IN}) sensing
- Loop 0 V_{OUT}, in case this V_{OUT} supplies downstream converter Loop 1 V_{IN} (e.g., post-buck topology)
- Register **tlmX_vin_force**, which forces the V_{IN} voltage value; this can be used by the FW to override the HW V_{IN} computation options
- Register **tlmX_vin_src_sel**, which allows the selection of the V_{IN} input source according to Table 51

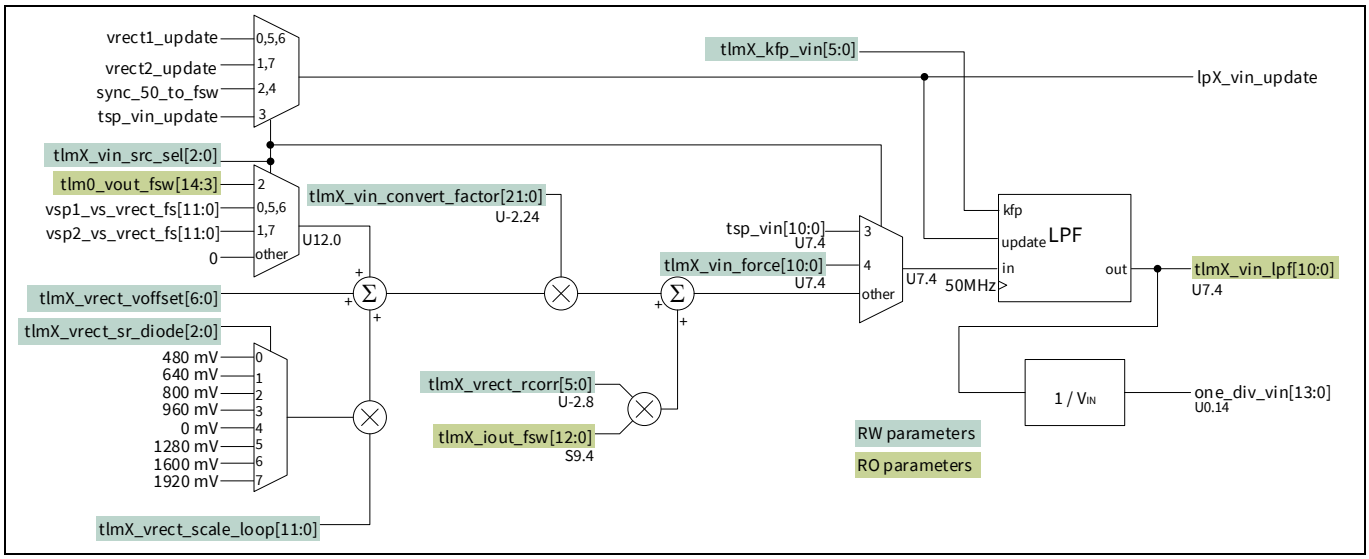

Figure 66 V_{IN} telemetry block diagram

Table 51 V_{IN} telemetry source selection

tlm_vin_src_sel	Source	Comments
0	VRSEN	Secondary V_{RECT} sense, ***vrs_init*** prior to start-up
1	BVRSEN	Secondary V_{RECT} sense, ***vrs_init*** prior to start-up
2	Loop 0 V_{OUT}	Select on Loop 1 when Loop 1 V_{IN} provided by Loop 0 V_{OUT}
3	PRISEN	Non-pulsed/primary V_{IN} sense via TSADC

tlm_vin_src_sel	Source	Comments
4	tlm_vin_force	Forced V_{IN} via FW (e.g., FW override of HW computation)
5	VRSEN	Secondary V_{RECT} sense, 0 V prior to start-up, select on Loop 1 when sharing Loop 0 V_{RECT} sense (e.g., interleaved topologies)
6	VRSEN	Non-pulsed/primary V_{IN} sense via VSADC
7	BVRSEN	Non-pulsed/primary V_{IN} sense via VSADC

When Loop 0 V_{OUT} is selected as the V_{IN} source for Loop 1, it is important to note that scaling by register **tlm1_vin_convert_factor** will apply. In order for this parameter to be computed correctly by the FW, Loop 1 (PMBus PAGE 1) commands should be programmed as follows:

- PAGE 1 MFR_VRECT_SCALE = PAGE 0 (Loop 0) VOUT_SCALE_LOOP converted to LINEAR11 format
- PAGE 1 MFR_TRANSFORMER_SCALE = B3FFh (0.999)

The selected input source is low-pass filtered at a programmable BW determined by register **tlmX_kfp_vin**. BW programming is shown in Table 52. The output of the LPF is available as read-only register **tlmX_vin_lpf**. It is used for the READ_VIN PMBus command as well as the V_{IN} fault functions. The V_{IN} telemetry submodule also provides a $1/V_{IN}$ output for use by the input CE.

Table 52 V_{IN}, I_{OUT}, I_{IN} and duty-cycle LPF BW programming, where LSB = 0.0001221 and F_{switch} = 1 MHz

tlm_kfp_vin tlm_kfp_iout tlm_kfp_iin tlm_kfp_duty	kfp	kfp_real	F3db/F_{switch}	tlm_kfp_vin tlm_kfp_iout tlm_kfp_iin tlm_kfp_duty	kfp	kfp_real	F3db/F_{switch}
0	4	0.0005	7.78E-05	20	128	0.0156	2.53E-03
1	5	0.0006	9.72E-05	21	160	0.0195	3.17E-03
2	6	0.0007	1.17E-04	22	192	0.0234	3.82E-03
3	7	0.0009	1.36E-04	23	224	0.0273	4.47E-03
4	8	0.0010	1.56E-04	24	256	0.0313	5.13E-03
5	10	0.0012	1.95E-04	25	320	0.0391	6.47E-03
6	12	0.0015	2.33E-04	26	384	0.0469	7.83E-03
7	14	0.0017	2.72E-04	27	448	0.0547	9.21E-03
8	16	0.0020	3.11E-04	28	512	0.0625	1.06E-02
9	20	0.0024	3.90E-04	29	640	0.0781	1.35E-02
10	24	0.0029	4.68E-04	30	768	0.0938	1.65E-02
11	28	0.0034	5.46E-04	31	896	0.1094	1.95E-02
12	32	0.0039	6.24E-04	32	1024	0.1250	2.27E-02
13	40	0.0049	7.81E-04	33	1280	0.1563	2.95E-02
14	48	0.0059	9.38E-04	34	1536	0.1875	3.67E-02
15	56	0.0068	1.10E-03	35	1792	0.2188	4.46E-02
16	64	0.0078	1.25E-03	36	2048	0.2500	5.31E-02
17	80	0.0098	1.57E-03	37	2560	0.3125	7.23E-02
18	96	0.0117	1.89E-03	38	3072	0.3750	9.55E-02
19	112	0.0137	2.21E-03	39	3584	0.4375	1.24E-01

Register **tlmX_vin_convert_factor** scales the VSADC formatted inputs for the transformer turns ratio and V_{RECT} sense resistor divider and converts the binary point to 1 V. This register value is computed by FW based on PMBus commands MFR_VRECT_SCALE and MFR_TRANSFORMER_SCALE.

If secondary-side sense is selected as the input source, several registers are provided to fine-tune the V_{RECT} to V_{IN} computation. There registers are:

- **tlmX_vrect_voffset**, which is a constant offset correction term added directly to V_{RECT} voltage coming from the VSP. The voltage should be programmed as seen at the (B)VRSEN input subsequent to the V_{RECT} sense resistor divider
- **tlmX_vrect_rcorr**, which provides an output current-dependent offset correction term applied subsequent to the scaling for transformer turns and resistor divider
- **tlmX_vrect_sr_diode**, which provides a voltage correction term on the V_{RECT} computation when SR FETs are off. It intends to compensate for one or two series body diodes, and the diode drop should be entered as seen at the rectification voltage prior to the sense resistor divider

8.3 Output current (I_{OUT}) telemetry

The I_{OUT} telemetry submodule block diagram is shown in Figure 67. Multiple input sources are capable of providing the output current telemetry. These are:

- ISEN and BISEN, which can be selected individually to provide I_{OUT} or, in the case of a dual-phase or interleaved topology, the sum of the ISEN and BISEN currents can be selected
- Register **tlmX_iout_src_sel**, which allows the selection of the I_{OUT} input source according to Table 53

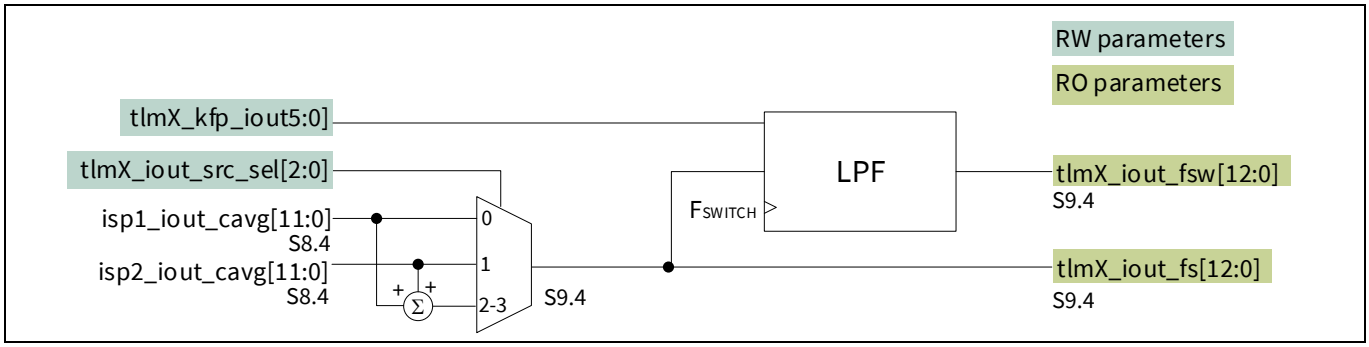

Figure 67 I_{OUT} telemetry block diagram

Table 53 I_{OUT} telemetry source selection

tlm_iout_src_sel	Source
0	ISEN
1	BISEN
2	ISEN + BISEN
3	ISEN + BISEN

The selected input source is low-pass filtered at a programmable BW determined by register **tlmX_kfp_iout**. This BW programming is shown in Table 52. The output of the LPF is available as read-only register **tlmX_iout_fsw**. It is used for the READ_IOUT PMBus command as well as the I_{OUT} fault functions. The input of the LPF is also available as read-only register **tlmX_iout_fs**, and it is used for the fast I_{OUT} fault functions.

8.4 Input current (I_{IN}) telemetry

The I_{IN} telemetry submodule block diagram is shown in Figure 68. Multiple input sources are capable of providing the input current telemetry. These are:

- ISEN and BISEN, which can be selected individually to provide I_{IN}, or input CE (Section 8.5) is provided to estimate I_{IN} based on measured V_{IN}, V_{OUT}, I_{OUT} and duty cycle if I_{IN} is not directly sensed
- Register **tlmX_iin_src_sel**, which allows the selection of the I_{IN} input source according to Table 54

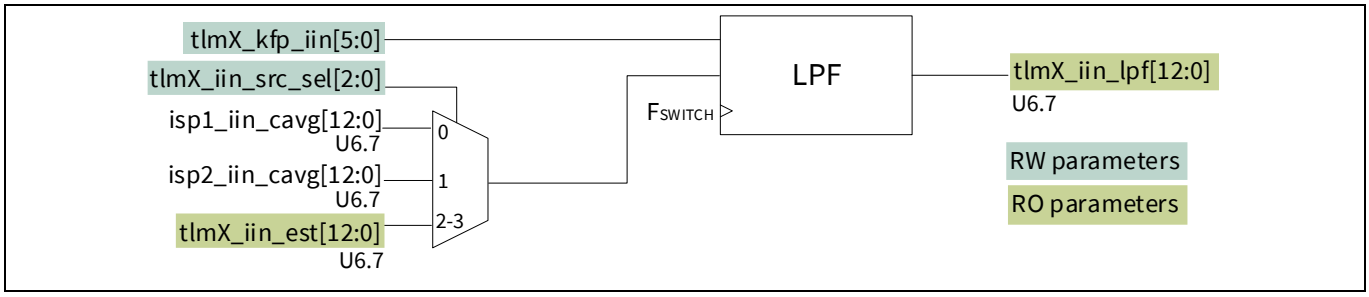

Figure 68 I_{IN} telemetry block diagram

Table 54 I_{IN} telemetry source selection

tlm_iin_src_sel	Source
0	ISEN
1	BISEN
2	Estimated I_{IN}
3	Estimated I_{IN}

The selected input source is low-pass filtered at a programmable BW determined by register **tlmX_kfp_iin**. The BW programming is shown in Table 52. The output of the LPF is available as read-only register **tlmX_iin_lpf**. It is used for the READ_IIN PMBus command as well as the I_{IN} fault functions.

8.5 Input CE

The I_{IN} submodule is shown in Figure 69. The estimated input current is obtained according to Equation (8.1).

$$I_{IN} = I_{OUT} * \left[\frac{V_{OUT}}{V_{IN}} + alpha * \left(Nx * Duty - \frac{V_{OUT}}{V_{IN}}\right)\right] \quad (8.1)$$

Where:

- Nx is defined according to Equation (8.2) in case of HB topology and according to Equation (8.3) for other topologies
- Alpha is defined via register **tlmX_iin_est_alpha**, and this value is used to fine-tune the estimate shifting of the relative percentage of V_{OUT}/V_{IN} versus duty cycle used to estimate I_{IN} from I_{OUT}

$$Nx = 2 * \frac{Nturn_sec}{Nturn_pri} \quad (8.2)$$

$$Nx = \frac{Nturn_sec}{Nturn_pri} \quad (8.3)$$

Register **tlmX_transformer_scale_loop** defines Nx and is automatically computed by FW based on PMBus command MFR_TRANSFORMER_SCALE and the selected topology.

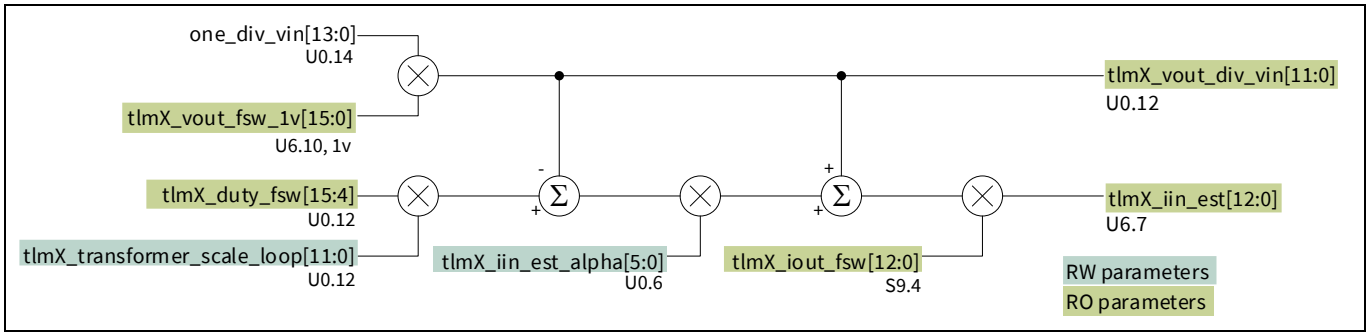

Figure 69 Input current estimation block diagram

8.6 Output and input power telemetry

Output and input power telemetry are computed in FW based on the HW-computed voltages and currents. The output power is obtained according to Equation (8.4) and correspondingly the input power is computed based on Equation (8.5).

$$P_{OUT} = tlmX_vout_fsw * tlmX_iout_fsw \quad (8.4)$$

$$P_{IN} = tlmX_vin_lpf * tlmX_iin_lpf \quad (8.5)$$

Registers **tlmX_vout_fsw** and **tlmX_iout_fsw** are the outputs of the output voltage and output current telemetry submodules, and correspondingly the registers **tlmX_vin_lpf** and **tlmX_iin_lpf** are the outputs of the input voltage and input current telemetry submodules.

8.7 Temperature telemetry

The temperature telemetry HW module block diagram is shown in Figure 70. The HW temperature telemetry module provides low-pass filtering of the following TSADC temperature outputs:

- ATSEN
- BTSEN
- ITSEN

The LPF BW is defined via register **tlm_kfp_tsen**, as shown in Table 55.

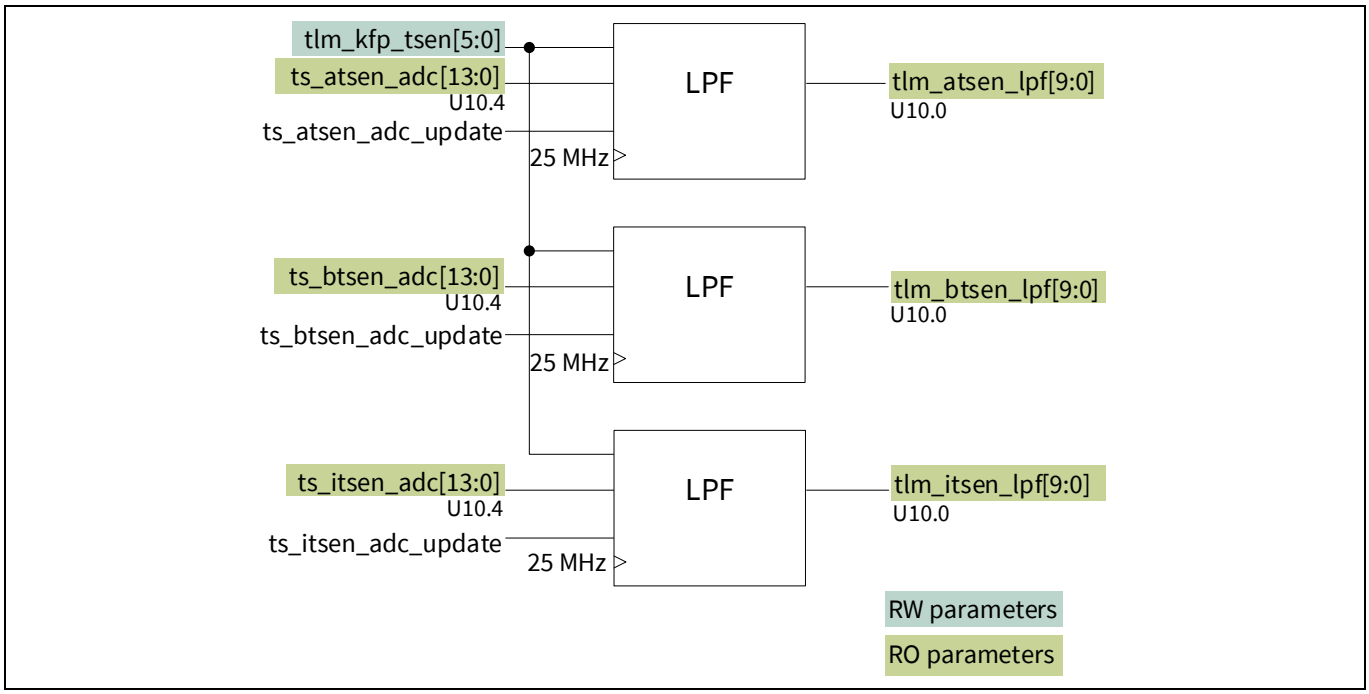

Figure 70 Temperature telemetry block diagram

Table 55 Temperature and general-purpose ADC LPF BW programming, where LSB = 0.0001221 and F_{SAMPLE} = 0.125 MHz

tlm_kfp_tsen tlm_kfp_imon tlm_kfp_xaddr	K_{FP}	kfp_real	F3dB (kHz)	tlm_kfp_tsen tlm_kfp_imon tlm_kfp_xaddr	K_{FP}	kfp_real	F3db (kHz)
0	4	0.0005	0.010	20	128	0.0156	0.316
1	5	0.0006	0.012	21	160	0.0195	0.396
2	6	0.0007	0.015	22	192	0.0234	0.477
3	7	0.0009	0.017	23	224	0.0273	0.559
4	8	0.0010	0.019	24	256	0.0313	0.642
5	10	0.0012	0.024	25	320	0.0391	0.809
6	12	0.0015	0.029	26	384	0.0469	0.978
7	14	0.0017	0.034	27	448	0.0547	1.151
8	16	0.0020	0.039	28	512	0.0625	1.326
9	20	0.0024	0.049	29	640	0.0781	1.686
10	24	0.0029	0.058	30	768	0.0938	2.058
11	28	0.0034	0.068	31	896	0.1094	2.443
12	32	0.0039	0.078	32	1024	0.1250	2.842
13	40	0.0049	0.098	33	1280	0.1563	3.684
14	48	0.0059	0.117	34	1536	0.1875	4.591
15	56	0.0068	0.137	35	1792	0.2188	5.570
16	64	0.0078	0.157	36	2048	0.2500	6.631
17	80	0.0098	0.196	37	2560	0.3125	9.043

tlm_kfp_tsen tlm_kfp_imon tlm_kfp_xaddr	K_{FP}	kfp_real	F3dB (kHz)	tlm_kfp_tsen tlm_kfp_imon tlm_kfp_xaddr	K_{FP}	kfp_real	F3db (kHz)
18	96	0.0117	0.236	38	3072	0.3750	11.937
19	112	0.0137	0.276	39	3584	0.4375	15.473

The conversion from ADC codes to temperature is performed in FW using a user-defined LUT. A default LUT is provided in ROM based on a 47 kΩ NTC sense element. Both NTC and PTC LUTs are allowed.

The following PMBus commands are supported for both loops with mapping from ATSEN, BTSEN, and ITSEN:

- READ_TEMPERATURE_1
- READ_TEMPERATURE_2

These mappings are defined by PMBus command MFR_SELECT_TEMPERATURE_SENSOR.Read_Temperature_1_Read_Temperature_2_source_select, as given in Table 56.

Table 56 READ_TEMPERATURE source selection

MFR_SELECT_TEMPERATURE_SENSOR. Read_Temperature_1_Read_Temperature_2_source_select	READ_TEMPERATURE_1 Source	READ_TEMPERATURE_2 Source
0, 6, 7	ATSEN	BTSEN
1	ATSEN	ITSEN
2	BTSEN	ATSEN
3	BTSEN	ITSEN
4	ITSEN	ATSEN
5	ITSEN	BTSEN

8.8 Duty-cycle telemetry

The duty-cycle telemetry submodule is shown in Figure 71. The ramp on-time (t2 – t1) is low-pass filtered at a programmable BW defined by register **tlmX_kfp_duty**, and the BW programming is shown in Table 52.

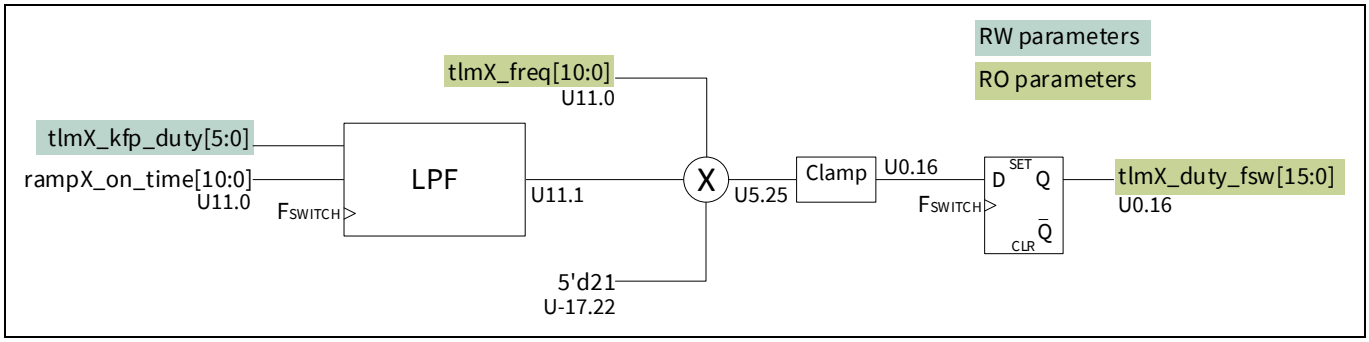

Figure 71 Duty-cycle telemetry block diagram

The filtered on-time is multiplied by the switching frequency (equivalent to dividing by the period) and further scaled to place the binary point at 1.0. The output of the duty-cycle telemetry is available on read-only register **tlmX_duty_fsw**. It is used for the PMBus command READ_DUTY_CYCLE as well as in the input CE.

8.9 Switching frequency (F_{switch}) telemetry

The F_{switch} telemetry submodule is shown in Figure 72. The measured switching period from the digital sync module is inverted to compute the frequency. This signal varies relatively slowly, thus no filtering is applied. The frequency output is available as read-only register **tlmX_freq** and is used for the READ_FREQUENCY PMBus command as well as by the duty-cycle telemetry module.

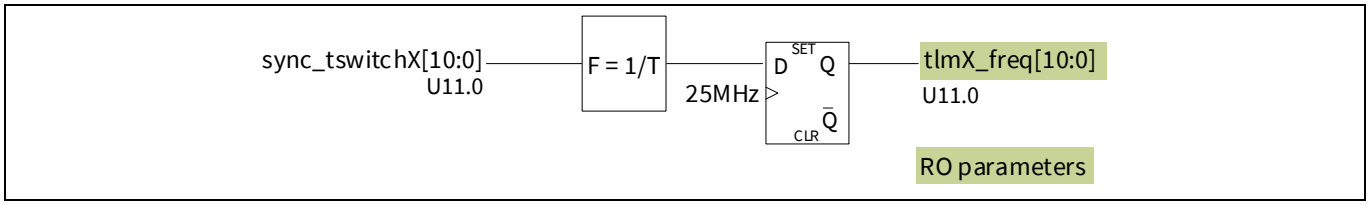

Figure 72 Switching frequency telemetry block diagram

8.10 General-purpose ADC telemetry

The general-purpose TSADC outputs are low-pass filtered as illustrated in the block diagram in Figure 73. The IMON and XADDR BWs are programmed via registers **tlm_kfp_imon** and **tlm_kfp_xaddr** according to the following definition:

- **tlm_kfp_xaddr** follows the BW programming given in Table 55
- For **ts_muxmode** = 5, 7; **tlm_kfp_imon** follows the BW programming given in Table 55
- For **ts_muxmode** = 4, 6; **tlm_kfp_imon** follows the BW programming given in Table 57

The **tlm_kfp_imon** BW programming depends on the **ts_muxmode** selection due to the different sample rates.

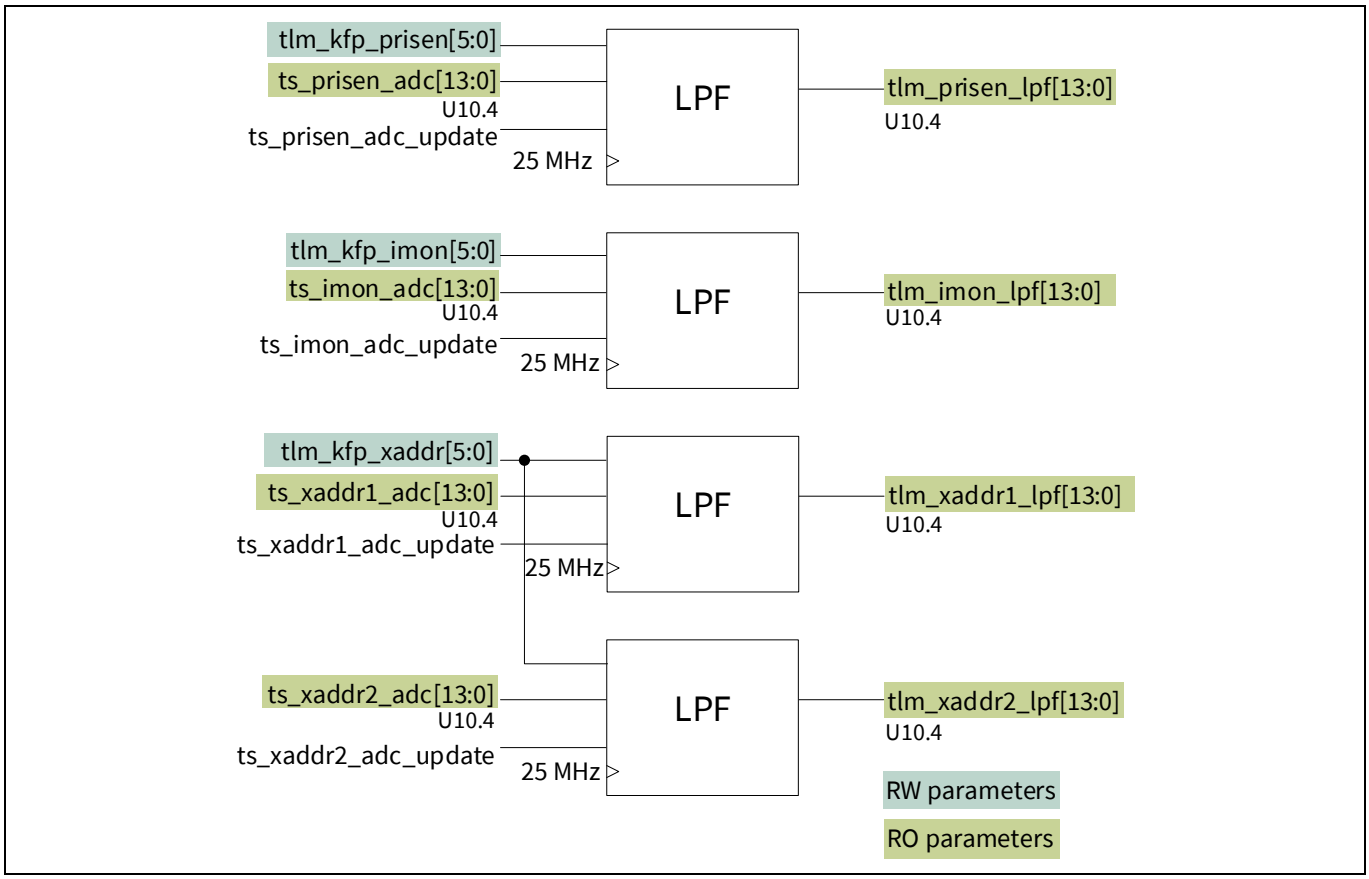

Figure 73 General-purpose ADC telemetry block diagram

The PRISEN BW is programmed through register **tlm_kfp_prisen** and follows a slightly different BW programming table, since it is sampled twice in the TSADC sampling cycle. The PRISEN BW programming is defined in Table 57 for **ts_muxmode** = 5, 7 and in Table 58 for **ts_muxmode** = 4, 5.

Table 57 PRISEN ADC LPF BW programming, where LSB = 0.0001221 and F_{sample} = 0.25 MHz. Applicable to PRISEN when ts_muxmode = 6, 7 and IMON when ts_muxmode = 4, 6

tlm_kfp_prisen	K_{FP}	kfp_real	F3dB (kHz)	tlm_kfp_prisen	K_{FP}	kfp_real	F3dB (kHz)
0	4	0.0005	0.019	20	128	0.0156	0.632
1	5	0.0006	0.024	21	160	0.0195	0.793
2	6	0.0007	0.029	22	192	0.0234	0.955
3	7	0.0009	0.034	23	224	0.0273	1.119
4	8	0.0010	0.039	24	256	0.0313	1.284
5	10	0.0012	0.049	25	320	0.0391	1.617
6	12	0.0015	0.058	26	384	0.0469	1.957
7	14	0.0017	0.068	27	448	0.0547	2.302
8	16	0.0020	0.078	28	512	0.0625	2.653
9	20	0.0024	0.097	29	640	0.0781	3.372
10	24	0.0029	0.117	30	768	0.0938	4.116
11	28	0.0034	0.136	31	896	0.1094	4.886
12	32	0.0039	0.156	32	1024	0.1250	5.684
13	40	0.0049	0.195	33	1280	0.1563	7.368
14	48	0.0059	0.235	34	1536	0.1875	9.182
15	56	0.0068	0.274	35	1792	0.2188	11.141
16	64	0.0078	0.313	36	2048	0.2500	13.263
17	80	0.0098	0.392	37	2560	0.3125	18.086
18	96	0.0117	0.472	38	3072	0.3750	23.873
19	112	0.0137	0.552	39	3584	0.4375	30.947

Table 58 PRISEN ADC LPF BW programming, where LSB = 0.0001221 and F_{SAMPLE} = 0.5 MHz. Applicable to PRISEN when ts_muxmode = 4, 5

tlm_kfp_prisen (ts_muxmode = 4, 5)	K_{FP}	kfp_real	F3dB (kHz)	tlm_kfp_prisen (ts_muxmode = 4, 5)	K_{FP}	kfp_real	F3dB (kHz)
0	4	0.0005	0.039	20	128	0.0156	1.263
1	5	0.0006	0.049	21	160	0.0195	1.585
2	6	0.0007	0.058	22	192	0.0234	1.910
3	7	0.0009	0.068	23	224	0.0273	2.237
4	8	0.0010	0.078	24	256	0.0313	2.567
5	10	0.0012	0.097	25	320	0.0391	3.235
6	12	0.0015	0.117	26	384	0.0469	3.914
7	14	0.0017	0.136	27	448	0.0547	4.604

tlm_kfp_prisen (ts_muxmode = 4, 5)	K_{FP}	kfp_real	F3dB (kHz)	tlm_kfp_prisen (ts_muxmode = 4, 5)	K_{FP}	kfp_real	F3dB (kHz)
8	16	0.0020	0.156	28	512	0.0625	5.305
9	20	0.0024	0.195	29	640	0.0781	6.744
10	24	0.0029	0.234	30	768	0.0938	8.232
11	28	0.0034	0.273	31	896	0.1094	9.773
12	32	0.0039	0.312	32	1024	0.1250	11.368
13	40	0.0049	0.390	33	1280	0.1563	14.737
14	48	0.0059	0.469	34	1536	0.1875	18.364
15	56	0.0068	0.548	35	1792	0.2188	22.282
16	64	0.0078	0.627	36	2048	0.2500	26.526
17	80	0.0098	0.785	37	2560	0.3125	36.172
18	96	0.0117	0.944	38	3072	0.3750	47.746
19	112	0.0137	1.103	39	3584	0.4375	61.894

8.11 Telemetry interrupts

The XDPP1100 provides 16 independently programmable telemetry IRQs. Figure 74 shows a block diagram of one IRQ along with the ORing of the other IRQs to generate signal tlm_irq, which is sent to the CPU. Register **tlm_irq_thr_src_sel_Y** (Y = 0 to 15) defines the input source as shown in Table 59.

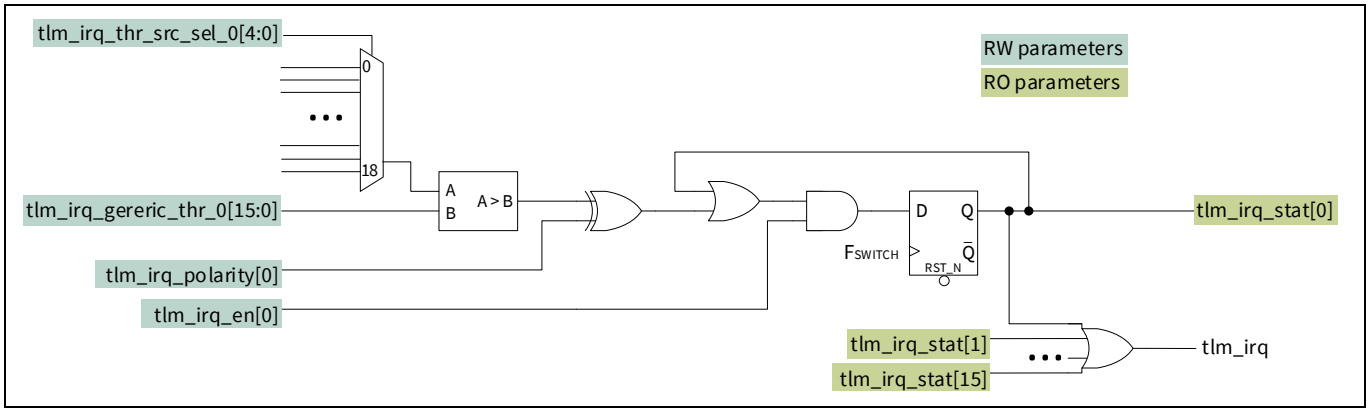

Figure 74 Telemetry IRQ block diagram

Table 59 Telemetry IRQ input source selection

tlm_irq_thr_src_sel	Source
0	Loop 0 I_{OUT}
1	Loop 1 I_{OUT}
2	Loop 0 V_{OUT}
3	Loop 1 V_{OUT}
4	Loop 0 I_{IN}
5	Loop 1 I_{IN}
6	Loop 0 V_{IN}

tlm_irq_thr_src_sel	Source
7	Loop 1 V_{IN}
8	Loop 0 duty cycle
9	Loop 1 duty cycle
10	Loop 0 F_{switch}
11	Loop 1 F_{switch}
12	ATSEN_ADC
13	BTSEN_ADC
14	Internal temperature
15	IMON_ADC
16	PRISEN_ADC
17	XADDR1_ADC
18	XADDR2_ADC
19 - 31	Unused

The following registers can be used for IRQ definition:

- **tlm_irq_gereric_thr_Y** defines the threshold against which the input source is compared; for register formatting based on the selected input, see Section 8.13
- **tlm_irq_polarity[Y]** defines the IRQ polarity (i.e., the direction of the comparison)
- Register **tlm_irq_en[Y]** is used to both enable and clear the IRQ

The telemetry IRQ status is available on read-only register **tlm_irq_stat**.

8.12 Telemetry high/low watermark detect

The XDPP1100 contains two watermark detectors per loop, designated as A and B. Figure 75 shows a block diagram of a single detector. Each watermark detector can detect and capture the high and low input value since the last clear. The input source is selected from V_{IN}, V_{OUT}, I_{IN} or I_{OUT} via register **tlmX_hilo_mark_Y_sel**, where X = 0, 1 for Loop 0, 1 and Y = A, B. Table 60 shows the input source selection.

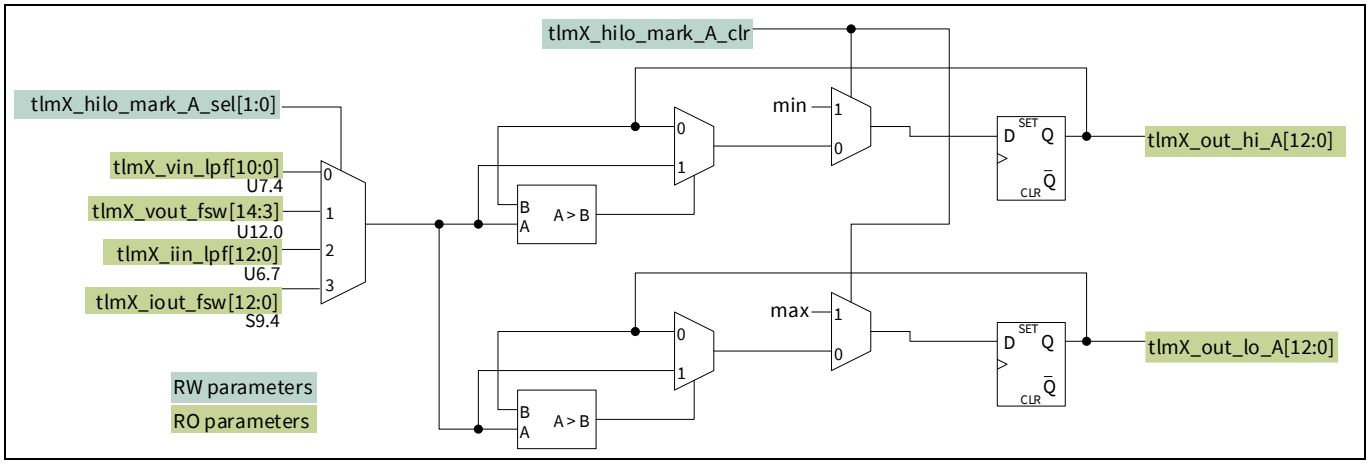

Figure 75 High/low watermark block diagram

Table 60 High/low watermark input source selection

tlm_hilo_mark_sel	Source
0	V_{IN}
1	V_{OUT}
2	I_{IN}
3	I_{OUT}

Register **tlmX_hilo_mark_Y_clr** is used to clear current high and low stored values. Upon a clear:

- The high register is reset to its minimum setting based on the format of the selected input; 0000h for unsigned inputs and 1000h for signed inputs
- The low register is reset to the most positive setting based on the format of the selected input; 1FFFh for unsigned inputs and 0FFFh for signed inputs

The current high and low stored values are available on read-only registers **tlmX_out_hi_Y** and **tlmX_out_lo_Y** respectively.

8.13 Telemetry registers

The relevant telemetry registers and their descriptions are provided in Table 61.

Table 61 Telemetry-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
telem	tlm_kfp_duty	RW	7000_3400h (Loop 0) 7000_3800h (Loop 1)	[5:0]	Duty cycle telemetry LPF coefficient index. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_duty[5:2] kfp_man = 4 + tlm_kfp_duty[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(kHz) = [kfp / (1 - kfp)] * F_{switch}(kHz) / 2*pi
telem	tlm_iin_est_alpha	RW	7000_3400h (Loop 0) 7000_3800h (Loop 1)	[11:0]	Input current estimate alpha coefficient. Defines the relative contributions of V_{OUT}/V_{IN} and duty cycle in the computation of the input current estimate. $I_{IN_est} = I_{OUT} * [(1-alpha)*V_{OUT}/V_{IN} +$ alpha*Duty*tlm_transformer_scale_loop] where tlm_transformer_scale_loop represents the V_{RECT}/V_{IN} ratio and is defined below LSB = 0.015625, range = 0.0 to 0.984375
telem	tlm_iin_src_sel	RW	7000_3400h (Loop 0)	[13:12]	Input current telemetry source select. 0: Measured on ISEN input

Peripheral	Field name	Access	Address	Bits	Description
			7000_3800h (Loop 1)		1: Measured on BISEN input 2 to 3: Estimated input current
telem	tlm_kfp_iin	RW	7000_3400h (Loop 0) 7000_3800h (Loop 1)	[19:14]	Input current telemetry LPF coefficient index. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_iin[5:2]" kfp_man = 4 + tlm_kfp_iin[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(kHz) = [kfp / (1 - kfp)] * F_{switch}(kHz) / 2*pi
telem	tlm_iout_src_sel	RW	7000_3400h (Loop 0) 7000_3800h (Loop 1)	[21:20]	Output current telemetry source select. 0: Measured on ISEN input 1: Measured on BISEN input 2 to 3: Sum of ISEN and BISEN inputs (dual-phase)
telem	tlm_kfp_iout	RW	7000_3400h (Loop 0) 7000_3800h (Loop 1)	[27:22]	Output current telemetry LPF coefficient index. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_iout[5:2] kfp_man = 4 + tlm_kfp_iout[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13""F3db(kHz) = [kfp / (1 - kfp)] * F_{switch}(kHz) / 2*pi
telem	tlm_vin_src_sel	RW	7000_3400h (Loop 0) 7000_3800h (Loop 1)	[30:28]	Input voltage telemetry source select. 0: VRSEN, secondary V_{RECT} sense, vrs_init prior to start-up 1: BVSEN_BVRSEN, secondary V_{RECT} sense, vrs_init prior to start-up 2: Loop 0 V_{OUT}, select on Loop 1 when Loop 1 V_{IN} provided by Loop 0 V_{OUT} (e.g., post-buck) 3: PRISEN, non-pulsed/primary V_{IN} sense via TSADC 4: tlm_vin_force, forced V_{IN} via FW (e.g., FW override of HW computation) 5: VRSEN, secondary V_{RECT} sense, 0 V prior to start-up. Select on Loop 1 when sharing Loop 0 V_{RECT} sense 6: VRSEN, non-pulsed/primary V_{IN} sense 7: BVSEN_BVRSEN, non-pulsed/primary V_{IN} sense

Peripheral	Field name	Access	Address	Bits	Description
					Note: When **tlm1_vin_src_sel** = 2 (Loop 0 V_{OUT}), set PAGE 1 MFR_TRANSFORMER_SCALE = B3FFh (0.999) and PAGE 1 MFR_VRECT_SCALE = PAGE 0 VOUT_SCALE_LOOP converted to LINEAR11 format for correct scaling of V_{IN} telemetry
telem	tlm_kfp_vin	RW	7000_3404h (Loop 0) 7000_3804h (Loop 1)	[5:0]	Input voltage telemetry LPF coefficient index. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_vin[5:2] kfp_man = 4 + tlm_kfp_vin[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(kHz) = [kfp / (1 - kfp)] * F_{switch}(kHz) / 2*pi
telem	tlm_kfp_vout	RW	7000_3404h (Loop 0) 7000_3804h (Loop 1)	[11:6]	Output voltage telemetry LPF coefficient index. Note that exp. settings greater than 7 are clamped to 7. Set to 63 to bypass filter. kfp_exp = tlm_kfp_vout[5:2] kfp_man = 4 + tlm_kfp_vout[1:0] kfp = kfp_man * 2^kfp_exp * 2^-15 F3db(kHz) = [kfp / (1 - kfp)] * 50e3 / 2*pi
telem	tlm_vrect_rcorr	RW	7000_3404h (Loop 0) 7000_3804h (Loop 1)	[17:12]	Resistive correction term applied to V_{RECT} based V_{IN} telemetry computation when V_{IN} is sensed via the VRSEN or BVRSEN input pin. This term is intended to correct current dependent offsets when using V_{RECT} sense. This term is multiplied by the output current and added to the V_{RECT} value after it has been converted to V_{IN}. The user should enter the effective resistance as seen at the VRSEN or BVRSEN input after accounting for the transformer turns ratio and the V_{RECT} resistor divider. LSB = 3.90625 mΩ, range = 0.0 to 246.09375 mΩ
telem	tlm_vrect_voffset	RW	7000_3404h (Loop 0) 7000_3804h (Loop 1)	[24:18]	Voltage offset correction term applied to the V_{RECT} computation. This term is added to the V_{RECT} value from the ADC. The user should enter

Peripheral	Field name	Access	Address	Bits	Description
					the effective offset as seen at the VRSEN input after the V_{RECT} sense resistor divider. LSB = 1.25 mV, -80 to +78.75 mV
telem	tlm_vrect_sr_diode	RW	7000_3404h (Loop 0) 7000_3804h (Loop 1)	[27:25]	Voltage correction term on V_{RECT} computation when SR FETs are off. Intended to compensate for one or two series body diodes. The user should enter the diode drop as seen at the V_{RECT} prior to the sense resistor divider. 0: 480 mV 1: 640 mV 2: 800 mV 3: 960 mV 4: 0 mV 5: 1280 mV 6: 1600 mV 7: 1920 mV
telem	tlm_duty_fsw	R	7000_3408h (Loop 0) 7000_3808h (Loop 1)	[15:0]	Low-pass filtered duty-cycle telemetry updated at F_{switch} rate. **PMBus command computed by FW as follows:** READ_DUTY_CYCLE(U8.2) = 400 * tlm_duty_fsw[15:0] / 2^16 LSB = 2^-16, range = 0.0 to 0.99998

telem	tlm_freq	R	7000_340Ch (Loop 0)
			7000_380Ch (Loop 1)
				[10:0]	Switching frequency telemetry including possible adjustment due to synchronization with external SYNC signal.
					PMBus command computed by FW as follows:
					if (tlm_freq[10])
					READ_FREQUENCY[15:0] = {5'd1, 1'd0, tlm_freq[10:1]}
					else
					READ_FREQUENCY[15:0] = {5'd0, 1'd0, tlm_freq[9:0]}
					LSB = 1 kHz, range = 0 to 2047 kHz

telem	tlm_iin_lpf	R	7000_3410h (Loop 0) 7000_3810h (Loop 1)	[12:0]	Low-pass filtered input current telemetry updated at F_{switch} rate. **PMBus command computed by FW as follows:** if (tlm_iin_lpf[12]) READ_IIN(U6.4) = {5'b11100, 1'b0, tlm_iin_lpd[12:3]} else if (tlm_iin_lpf[11])

Peripheral	Field name	Access	Address	Bits	Description
					READ_IIN(U5.5) = {5'b11011, 1'b0, tlm_iin_lpd[11:2]} else if (tlm_iin_lpf[10]) READ_IIN(U4.6) = {5'b11010, 1'b0, tlm_iin_lpd[10:1]} else READ_IIN(U3.7) = {5'b11001, 1'b0, tlm_iin_lpd[9:0]} LSB = 7.8125 mA, range = 0.0 to 63.9922 A
telem	tlm_iin_est	R	7000_3414h (Loop 0) 7000_3814h (Loop 1)	[12:0]	Unfiltered result of estimated input current computation based on I_{OUT}, V_{OUT}, V_{IN} and duty-cycle values. LSB = 7.8125 mA, range = 0.0 to 63.9922 A
telem	tlm_vout_div_vin	R	7000_3418h (Loop 0) 7000_3818h (Loop 1)	[11:0]	Unfiltered result of equation (V_{OUT} / V_{IN}) computed during input current estimation. LSB = 2^-12 V/V, range = 0.0 to 0.9998 V/V
telem	tlm_iout_fs	R	7000_341Ch (Loop 0) 7000_381Ch (Loop 1)	[12:0]	Unfiltered, cycle-averaged output current updated at F_{switch} rate. LSB = 62.5 mA, range = -256 to +255.9375 A
telem	tlm_iout_fsw	R	7000_3420h (Loop 0) 7000_3820h (Loop 1)	[12:0]	Low-pass filtered output current telemetry updated at F_{switch} rate. **PMBus command computed by FW as follows:** READ_IOUT(S9.2) = tlm_iout_fsw[12:2] LSB = 62.5 mA, range = -256 to +255.9375 A
telem	tlm_vin_force	RW	7000_3424h (Loop 0) 7000_3824h (Loop 1)	[10:0]	Forced V_{IN} input value. Selected as source for V_{IN} by tlm_vin_src_sel. Can be used by FW to bypass the HW-based V_{IN} computation. LSB = 62.5 mV, range = 0.0 to 127.9375 V
telem	tlm_vin_convert_factor	RW	7000_3428h (Loop 0) 7000_3828h (Loop 1)	[21:0]	Conversion factor for computing V_{IN} from the measured VSADC output. It consists of three components: scaling for the V_{RECT} resistor divider, scaling for the transformer/topology and shifting the binary point from the ADC representation (1.25 mV) to 1 V. **Computed by FW from PMBus**

Peripheral	Field name	Access	Address	Bits	Description
					commands as follows: exp1 = $signed(MFR_VRECT_SCALE [15:11]), exp=-10, -11, -12 supported exp2 = $signed(MFR_TRANSFORMER_ SCALE[15:11]), exp=-10, -11, -12 supported prod1(U0.14) = MFR_VRECT_SCALE[9:0] * MFR_TRANSFORMER_SCALE[9:0] / 2^(-14-exp1-exp2) quot1(U7.8) = 2^22 / prod1(U0.14), clamp to 2^15-1 tlm_vin_convert_factor(U-2.24) = quot1(U7.8) * 82(U-9.16) LSB = 2^-24, range = 0.0 to 0.2500
telem	tlm_vin_lpf	R	7000_342C h (Loop 0) 7000_382C h (Loop 1)	[10:0]	Low-pass filtered input voltage telemetry. **PMBus command computed by FW as follows:** READ_VIN(U7.2) = tlm_vin_lpf[10:2] LSB = 62.5 mV, range = 0.0 to 127.9375 V
telem	tlm_vout_convert_fact or	RW	7000_3430 h (Loop 0) 7000_3830 h (Loop 1)	[20:0]	Conversion factor for computing V_{OUT} from the measured VSADC output. It consists of two components: scaling for the V_{OUT} resistor divider and shifting the binary point from the ADC representation (1.25 mV) to 1 V. **Computed by FW from PMBus commands as follows:** Quot1(U6.8) = 2^24 / VOUT_SCALE_LOOP(U0.16), clamp to 2^14-1 tlm_vout_convert_factor(U-3.24) = Quot1(U6.8) * 82(U-9.16) LSB = 2^-24, range = 0.0 to 0.1250
telem	tlm_vout_fs	R	7000_3434 h (Loop 0) 7000_3834 h (Loop 1)	[11:0]	Output voltage (V_{OUT}) telemetry at the sample clock rate (50 MHz). **Note:** This field is in VSADC code format (i.e., the voltage at the VSEN pin after the sense resistor divider) and must be scaled by VOUT_SCALE_LOOP for output-referenced voltage. See tlm_vout_fsw description for scale

Peripheral	Field name	Access	Address	Bits	Description
					equation. LSB = 1.25 mV, range = 0.0 to 5.11875 V
telem	tlm_vout_fsw	R	7000_3438h (Loop 0) 7000_3838h (Loop 1)	[14:0]	Low-pass filtered output voltage (V_{OUT}) telemetry updated at the F_{switch} clock rate. **Note:** This field is in VSADC code format (i.e., the voltage at the VSEN pin after the sense resistor divider) and must be scaled by VOUT_SCALE_LOOP for output-referenced voltage. **PMBus command computed by FW as follows:** READ_VOUT(U16-X.X) = (tlm_vout_fsw(U12.3) / 800) * (2^16 / VOUT_SCALE_LOOP(U0.16)) * 2^(X-3) where X = negative of VOUT_MODE exponent LSB = 0.15625 mV, range = 0.0 to 5.11875 V
telem	tlm_transformer_scale_loop	RW	7000_343Ch (Loop 0) 7000_383Ch (Loop 1)	[11:0]	Ratio of V_{RECT} to V_{IN}. For HB equal to N_{turn_sec} / (2 * N_{turn_pri}), otherwise equal to N_{turn_sec} / N_{turn_pri}. **Computed by FW from PMBus as follows:** exp = $signed(MFR_TRANSFORMER_SCALE[15:11]), exp = -10, -11, -12 supported hb = 1 if HB topology and 0 otherwise if (hb==0 && exp==-10) tlm_vrect_scale_loop[11:0] = {MFR_TRANSFORMER_SCALE[9:0], 00} else if (hb==0 && exp==-11) tlm_vrect_scale_loop[11:0] = {0, MFR_TRANSFORMER_SCALE[9:0], 0} else if (hb==0 && exp==-12) tlm_vrect_scale_loop[11:0] = {00, MFR_TRANSFORMER_SCALE[9:0]} else if (hb==1 && exp==-10) tlm_vrect_scale_loop[11:0] = {0, MFR_TRANSFORMER_SCALE[9:0], 0} else if (hb==1 && exp==-11)

Peripheral	Field name	Access	Address	Bits	Description
					tlm_vrect_scale_loop[11:0] = {00, MFR_TRANSFORMER_SCALE[9:0]} else if (hb==1 && exp==-12) tlm_vrect_scale_loop[11:0] = {000, MFR_TRANSFORMER_SCALE[9:1]} LSB = 2^-12, range = 0.0 to 0.99976
telem	tlm_vrect_scale_loop	RW	7000_3440h (Loop 0) 7000_3840h (Loop 1)	[11:0]	V_{RECT} sense resistor scale value. **Computed by FW from PMBus as follows:** exp = $signed(MFR_VRECT_SCALE [15:11]), exp=-10, -11, -12 supported tlm_vrect_scale_loop[11:0] = MFR_VRECT_SCALE[9:0] << (exp + 12) LSB = 2^-12, range = 0.0 to 0.99976
telem	tlm_vout_scale_loop	RW	7000_3444h (Loop 0) 7000_3844h (Loop 1)	[15:0]	V_{OUT} sense resistor scale value. **Computed by FW from PMBus as follows:** tlm_vout_scale_loop(U0.16) = VOUT_SCALE_LOOP(U0.16) LSB = 2^-16, range = 0.0 to 0.9998
telem	tlm_hilo_mark_A_sel	RW	7000_3448h (Loop 0) 7000_3848h (Loop 1)	[1:0]	Input select for high/low watermark detector A. 0: V_{IN} 1: V_{OUT} 2: I_{IN} 3: I_{OUT}
telem	tlm_hilo_mark_B_sel	RW	7000_3448h (Loop 0) 7000_3848h (Loop 1)	[3:2]	Input select for high/low watermark detector B. 0: V_{IN} 1: V_{OUT} 2: I_{IN} 3: I_{OUT}
telem	tlm_hilo_mark_A_clr	W	7000_344Ch (Loop 0) 7000_384Ch (Loop 1)	[0]	Clear for high/low watermark detector A. Write first to 1 then to 0 to clear detector
telem	tlm_hilo_mark_B_clr	W	7000_3450h (Loop 0) 7000_3850h (Loop 1)	[0]	Clear for high/low watermark detector B. Write first to 1 then to 0 to clear detector
telem	tlm_out_hi_A	R	7000_3454h (Loop 0) 7000_3854h (Loop 1)	[12:0]	High watermark detector A output. Format depends on input selected by tlm_hilo_mark_A_sel. V_{IN}: Unsigned, LSB = 62.5 mV, range = 0.0 to 127.9375 V V_{OUT}: Unsigned, LSB = 1.25 mV,

Peripheral	Field name	Access	Address	Bits	Description
					range = 0.0 to 5.11875 V I_{IN}: Unsigned, LSB = 7.8125 mA, range = 0.0 to 63.9922 A I_{OUT}: Signed, LSB = 62.5 mA, range = -256 to +255.9375 A
telem	tlm_out_lo_A	R	7000_3454h (Loop 0) 7000_3854h (Loop 1)	[25:13]	Low watermark detector A output. Format depends on input selected by tlm_hilo_mark_A_sel. V_{IN}: Unsigned, LSB = 62.5 mV, range = 0.0 to 127.9375 V V_{OUT}: Unsigned, LSB = 1.25 mV, range = 0.0 to 5.11875 V I_{IN}: Unsigned, LSB = 7.8125 mA, range = 0.0 to 63.9922 A I_{OUT}: Signed, LSB = 62.5 mA, range = -256 to +255.9375 A
telem	tlm_out_hi_B	R	7000_3458h (Loop 0) 7000_3858h (Loop 1)	[12:0]	High watermark detector B output. Format depends on input selected by tlm_hilo_mark_B_sel. V_{IN}: Unsigned, LSB = 62.5 mV, range = 0.0 to 127.9375 V V_{OUT}: Unsigned, LSB = 1.25 mV, range = 0.0 to 5.11875 V I_{IN}: Unsigned, LSB = 7.8125 mA, range = 0.0 to 63.9922 A I_{OUT}: Signed, LSB = 62.5 mA, range = -256 to +255.9375 A
telem	tlm_out_lo_B	R	7000_3458h (Loop 0) 7000_3858h (Loop 1)	[25:13]	Low watermark detector B output. Format depends on input selected by tlm_hilo_mark_B_sel. V_{IN}: Unsigned, LSB = 62.5 mV, range = 0.0 to 127.9375 V V_{OUT}: Unsigned, LSB = 1.25 mV, range = 0.0 to 5.11875 V I_{IN}: Unsigned, LSB = 7.8125 mA, range = 0.0 to 63.9922 A I_{OUT}: Signed, LSB = 62.5 mA, range = -256 to +255.9375 A
tlmcom	tlm_kfp_tsen	RW	7000_5000h	[5:0]	ATSEN, BTSEN and ITSEN telemetry low-pass filter coefficient index. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_tsen[5:2] kfp_man = 4 + tlm_kfp_tsen[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13

Peripheral	Field name	Access	Address	Bits	Description
					F3db(kHz) = [kfp / (1 - kfp)] * 125 kHz / 2*pi
tlmcom	tlm_kfp_prisen	RW	7000_5000h	[11:6]	PRISEN telemetry low-pass filter coefficient index when used as a general-purpose ADC. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_prisen[5:2] kfp_man = 4 + tlm_kfp_prisen[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(kHz) = [kfp / (1 - kfp)] * 125 kHz / 2*pi
tlmcom	tlm_kfp_imon	RW	7000_5000h	[17:12]	IMON telemetry low-pass filter coefficient index when used as a general-purpose ADC. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_imon[5:2] kfp_man = 4 + tlm_kfp_imon[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(kHz) = [kfp / (1 - kfp)] * 125 kHz / 2*pi
tlmcom	tlm_kfp_xaddr	RW	7000_5000h	[23:18]	XADDR1 and XADDR2 telemetry low-pass filter coefficient index when used as a general-purpose ADC. Note that exp. settings greater than 9 are clamped to 9. Set to 63 to bypass filter. kfp_exp = tlm_kfp_xaddr[5:2] kfp_man = 4 + tlm_kfp_xaddr[1:0] kfp = kfp_man * 2^kfp_exp * 2^-13 F3db(kHz) = [kfp / (1 - kfp)] * 125 kHz / 2*pi
tlmcom	tlm_irq_gereric_thr_0	RW	7000_5004h	[15:0]	Telemetry IRQ 0 threshold. Compared against signal selected by tlm_irq_thr_src_sel_0. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V

Peripheral	Field name	Access	Address	Bits	Description
					Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_0	RW	7000_5004h	[20:16]	Telemetry IRQ 0 source select. Selects signal to compare against tlm_irq_gereric_thr_0 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_1	RW	7000_5008h	[15:0]	Telemetry IRQ 1 threshold. Compared against signal selected by tlm_irq_thr_src_sel_1. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0

Peripheral	Field name	Access	Address	Bits	Description
					to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_1	RW	7000_5008h	[20:16]	Telemetry IRQ 1 source select. Selects signal to compare against tlm_irq_gereric_thr_1 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_2	RW	7000_500Ch	[15:0]	Telemetry IRQ 2 threshold. Compared against signal selected by tlm_irq_thr_src_sel_2. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0

Peripheral	Field name	Access	Address	Bits	Description
					to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_2	RW	7000_500Ch	[20:16]	Telemetry IRQ 2 source select. Selects signal to compare against tlm_irq_gereric_thr_2 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to31: Unused
tlmcom	tlm_irq_gereric_thr_3	RW	7000_5010h	[15:0]	Telemetry IRQ 3 threshold. Compared against signal selected by tlm_irq_thr_src_sel_3. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB =

Peripheral	Field name	Access	Address	Bits	Description
					1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_3	RW	7000_5010h	[20:16]	Telemetry IRQ 3 source select. Selects signal to compare against tlm_irq_gereric_thr_3 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_4	RW	7000_5014h	[15:0]	Telemetry IRQ 4 threshold. Compared against signal selected by tlm_irq_thr_src_sel_4. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes

Peripheral	Field name	Access	Address	Bits	Description
					IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_4	RW	7000_5014h	[20:16]	Telemetry IRQ 4 source select. Selects signal to compare against tlm_irq_gereric_thr_4 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_5	RW	7000_5018h	[15:0]	Telemetry IRQ 5 threshold. Compared against signal selected by tlm_irq_thr_src_sel_5. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_5	RW	7000_5018h	[20:16]	Telemetry IRQ 5 source select. Selects signal to compare against tlm_irq_gereric_thr_5 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_6	RW	7000_501Ch	[15:0]	Telemetry IRQ 6 threshold. Compared against signal selected by tlm_irq_thr_src_sel_6. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_6	RW	7000_501Ch	[20:16]	Telemetry IRQ 6 source select. Selects signal to compare against tlm_irq_gereric_thr_6 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_7	RW	7000_5020h	[15:0]	Telemetry IRQ 7 threshold. Compared against signal selected by tlm_irq_thr_src_sel_7. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_7	RW	7000_5020h	[20:16]	Telemetry IRQ 7 source select. Selects signal to compare against tlm_irq_gereric_thr_7 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_8	RW	7000_5024h	[15:0]	Telemetry IRQ 8 threshold. Compared against signal selected by tlm_irq_thr_src_sel_8. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_8	RW	7000_5024h	[20:16]	Telemetry IRQ 8 source select. Selects signal to compare against tlm_irq_gereric_thr_8 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_9	RW	7000_5028h	[15:0]	Telemetry IRQ 9 threshold. Compared against signal selected by tlm_irq_thr_src_sel_9. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_9	RW	7000_5028h	[20:16]	Telemetry IRQ 9 source select. Selects signal to compare against tlm_irq_gereric_thr_9 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_10	RW	7000_502Ch	[15:0]	Telemetry IRQ 10 threshold. Compared against signal selected by tlm_irq_thr_src_sel_10. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_10	RW	7000_502Ch	[20:16]	Telemetry IRQ 10 source select. Selects signal to compare against tlm_irq_gereric_thr_10 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_11	RW	7000_5030h	[15:0]	Telemetry IRQ 11 threshold. Compared against signal selected by tlm_irq_thr_src_sel_11. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_11	RW	7000_5030h	[20:16]	Telemetry IRQ 11 source select. Selects signal to compare against tlm_irq_gereric_thr_11 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_12	RW	7000_5034h	[15:0]	Telemetry IRQ 12 threshold. Compared against signal selected by tlm_irq_thr_src_sel_12. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2:

Peripheral	Field name	Access	Address	Bits	Description
					U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes
tlmcom	tlm_irq_thr_src_sel_12	RW	7000_5034h	[20:16]	Telemetry IRQ 12 source select. Selects signal to compare against tlm_irq_gereric_thr_12 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_13	RW	7000_5038h	[15:0]	Telemetry IRQ 13 threshold. Compared against signal selected by tlm_irq_thr_src_sel_13. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes

Peripheral	Field name	Access	Address	Bits	Description
tlmcom	tlm_irq_thr_src_sel_13	RW	7000_5038h	[20:16]	Telemetry IRQ 13 source select. Selects signal to compare against tlm_irq_gereric_thr_13 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_14	RW	7000_503Ch	[15:0]	Telemetry IRQ 14 threshold. Compared against signal selected by tlm_irq_thr_src_sel_14. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes

Peripheral	Field name	Access	Address	Bits	Description
tlmcom	tlm_irq_thr_src_sel_14	RW	7000_503Ch	[20:16]	Telemetry IRQ 14 source select. Selects signal to compare against tlm_irq_gereric_thr_14 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_irq_gereric_thr_15	RW	7000_5040h	[15:0]	Telemetry IRQ 15 threshold. Compared against signal selected by tlm_irq_thr_src_sel_15. Format based on selected source. I_{OUT}: S9.4, LSB = 62.5 mA, range = -256 to +255.9375 A V_{OUT}: U12.3, LSB = 156.25 µV, range = 0.0 to 5.11984375 V I_{IN}: U6.7, LSB = 7.8125 mA, range = 0.0 to 63.9922 A V_{IN}: U7.4, LSB = 62.5 mV, range = 0.0 to 127.9375 V Duty: U0.16, LSB = 2^-16, range = 0.0 to 0.99998 F_{switch}: U11.0, LSB = 1 kHz, range = 0 to 2047 kHz ATSEN, BTSEN, ITSEN: U10.0, LSB = 1 ADC code, range = 0 to 1023 ADC codes IMON, PRISEN, XADDR1, XADDR2: U10.4, LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes

Peripheral	Field name	Access	Address	Bits	Description
tlmcom	tlm_irq_thr_src_sel_15	RW	7000_5040h	[20:16]	Telemetry IRQ 15 source select. Selects signal to compare against tlm_irq_gereric_thr_15 for IRQ generation. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT} 2: Loop 0 V_{OUT} 3: Loop 1 V_{OUT} 4: Loop 0 I_{IN} 5: Loop 1 I_{IN} 6: Loop 0 V_{IN} 7: Loop 1 V_{IN} 8: Loop 0 duty 9: Loop 1 duty 10: Loop 0 F_{switch} 11: Loop 1 F_{switch} 12: ATSEN 13: BTSEN 14: Internal temp. 15: IMON 16: PRISEN 17: XADDR1 18: XADDR2 19 to 31: Unused
tlmcom	tlm_atsen_lpf	R	7000_5044h	[9:0]	Low-pass filtered ATSEN telemetry output. Note that the format of this register is ADC codes. The ADC code to temperature conversion is implemented in FW. LSB = 1 ADC code, range = 0 to 1023 ADC codes
tlmcom	tlm_atsen_fs	R	7000_5048h	[11:0]	Unfiltered ATSEN telemetry output. Note that the format of this register is ADC codes. The ADC code to temperature conversion is implemented in FW. LSB = 0.25 ADC code, range = 0 to 1023.75 ADC codes
tlmcom	tlm_btsen_lpf	R	7000_504Ch	[9:0]	Low-pass filtered BTSEN telemetry output. Note that the format of this register is ADC codes. The ADC code to temperature conversion is implemented in FW. LSB = 1 ADC code, range = 0 to 1023 ADC codes

Peripheral	Field name	Access	Address	Bits	Description
tlmcom	tlm_btsen_fs	R	7000_5050h	[11:0]	Unfiltered BTSEN telemetry output. Note that the format of this register is ADC codes. The ADC code to temperature conversion is implemented in FW. LSB = 0.25 ADC code, range = 0 to 1023.75 ADC codes
tlmcom	tlm_itsen_lpf	R	7000_5054h	[9:0]	Low-pass filtered ITSEN telemetry output. Note that the format of this register is ADC codes. The ADC code to temperature conversion is implemented in FW. LSB = 1 ADC code, range = 0 to 1023 ADC codes
tlmcom	tlm_itsen_fs	R	7000_5058h	[11:0]	Unfiltered ITSEN telemetry output. Note that the format of this register is ADC codes. The ADC code to temperature conversion is implemented in FW. LSB = 0.25 ADC code, range = 0 to 1023.75 ADC codes
tlmcom	tlm_irq_polarity	RW	7000_505Ch	[15:0]	Telemetry IRQ comparison polarity select where bit [X] corresponds to IRQX. 0: Assert IRQ when signal is greater than IRQ threshold 1: Assert IRQ when signal is less than or equal to IRQ threshold
tlmcom	tlm_irq_en	RW	7000_5060h	[15:0]	Telemetry IRQ enable/clear where bit [X] corresponds to IRQX. 0: Interrupt disabled and cleared 1: Interrupt enabled
tlmcom	tlm_irq_stat	R	7000_5064h	[15:0]	Telemetry IRQ status where bit [X] corresponds to IRQX. 0: IRQ not set 1: IRQ set
tlmcom	tlm_imon_adc_lpf	R	7000_5068h	[13:0]	Low-pass filtered IMON telemetry output when used as a general-purpose ADC. LSB = 0.0625 ADC code, range = 0 to 1023.9375 ADC codes
tlmcom	tlm_prisen_adc_lpf	R	7000_506Ch	[13:0]	Low-pass filtered PRISEN telemetry output when used as a general-purpose ADC. LSB = 0.0625 ADC code, range = 0 to 1023.9375 ADC codes

Peripheral	Field name	Access	Address	Bits	Description
tlmcom	tlm_xaddr1_pinset	R	7000_5070h	[3:0]	XADDR1 pinset measurement decoded value.
tlmcom	tlm_xaddr2_pinset	R	7000_5070h	[3:0]	XADDR2 pinset measurement decoded value.
tlmcom	tlm_xaddr1_adc_lpf	R	7000_5074h	[13:0]	Low-pass filtered XADDR1 telemetry output when used as a general-purpose ADC. LSB = 0.0625 ADC code, range = 0 to 1023.9375 ADC codes
tlmcom	tlm_xaddr2_adc_lpf	R	7000_5078h	[13:0]	Low-pass filtered XADDR2 telemetry output when used as a general-purpose ADC. LSB = 0.0625 ADC code, range = 0 to 1023.9375 ADC codes

8.14 PMBus

The relevant PMBus commands for telemetry are given in Table 62.

Table 62 PWM-related PMBus commands

Command name	Access	Length	Address	Bits	Description
VOUT_SCALE_LOOP	RW	Word	29h	[15:0]	Scales VOUT_COMMAND and other V_{OUT}-related commands for the external resistor divider between V_{OUT} and VSEN. The format is DIRECT, U0.16 and is computed as: VOUT_SCALE_LOOP = INT(2^16 * VSEN / V_{OUT})
READ_VIN	R	Word	88h	[15:0]	Returns the measured input voltage in volts in the LINEAR11 format with exponent defined by FW_CONFIG_TELEMETRY.READ_VIN_EXP.
READ_IIN	R	Word	89h	[15:0]	Returns the measured input current in amps in the LINEAR11 format with exponent defined by FW_CONFIG_TELEMETRY.READ_IIN_EXP.
READ_VOUT	R	Word	8Bh	[15:0]	Returns the measured (not commanded) output voltage in

Command name	Access	Length	Address	Bits	Description
					the same format as set by the VOUT_MODE command.
READ_IOUT	R	Word	8Ch	[15:0]	Returns the measured output current in amps in the LINEAR11 format with exponent defined by FW_CONFIG_TELEMETRY.READ_IOUT_EXP.
READ_TEMPERATURE_1	R	Word	8Dh	[15:0]	Returns the temperature at sensor 1 in degrees Celsius in the LINEAR11 format. The temperature source for sensor 1 is defined by MFR_SELECT_TEMPERATURE_SENSOR.Read_Temperature_1_Read_Temperature_2_source_select
READ_TEMPERATURE_2	R	Word	8Eh	[15:0]	Returns the temperature at sensor 2 in degrees Celsius in the LINEAR11 format. The temperature source for sensor 2 is defined by MFR_SELECT_TEMPERATURE_SENSOR.Read_Temperature_1_Read_Temperature_2_source_select
READ_DUTY_CYCLE	R	Word	94h	[15:0]	Returns the duty cycle in percent in the LINEAR11 format with exponent defined by FW_CONFIG_TELEMETRY.READ_DUTY_EXP
READ_FREQUENCY	R	Word	95h	[15:0]	Returns the switching frequency in kHz in the LINEAR11 format. If the device is configured as a SYNC slave, this command will return the actual switching frequency and not the commanded switching frequency
READ_POUT	R	Word	96h	[15:0]	Returns the FW-computed output power in watts in the LINEAR11 format with exponent defined by FW_CONFIG_TELEMETRY.READ_POWER_EXP
READ_PIN	R	Word	97h	[15:0]	Returns the FW-computed input power in watts in the LINEAR11 format with exponent defined by

Command name	Access	Length	Address	Bits	Description
					FW_CONFIG_TELEMETRY.READ_POWER_EXP

MFR_VRECT_SCALE	RW	Word	CDh	[15:0]	Scales the V_{RECT} input voltage computation for the external resistor divider between V_{RECT} and VRSEN. The format is LINEAR11 with recommended exponents -10, -11, -12
					if (VRSEN / V_{RECT} is greater than 0.5)
					MFR_VRECT_SCALE[15:11] = -10d = 16h
					MFR_VRECT_SCALE[10:0] = INT(2^10 * VRSEN / V_{RECT})
					else if (VRSEN / V_{RECT} is greater than 0.25)
					MFR_VRECT_SCALE[15:11] = -11d = 15h
					MFR_VRECT_SCALE[10:0] = INT(2^11 * VRSEN / V_{RECT})
					else
					MFR_VRECT_SCALE[15:11] = -12d = 14h
					MFR_VRECT_SCALE[10:0] = INT(2^12 * VRSEN / V_{RECT})
					Note: When **tlm1_vin_src_sel** = 2 (Loop 0 V_{OUT}), set PAGE 1 MFR_VRECT_SCALE = PAGE 0 VOUT_SCALE_LOOP converted to LINEAR11 format for correct scaling of V_{IN} telemetry

MFR_TRANSFORMER_SCALE	RW	Word	CEh	[15:0]	Defines the transformer turns ratio, N_{turn_sec} / N_{turn_prim}. The format is LINEAR11 with recommended exponents -10, -11, -12 if (N_{turn_sec} / N_{turn_prim} is greater than 0.5) MFR_ TRANSFORMER _SCALE[15:11] = -10d = 16h MFR_ TRANSFORMER _SCALE[10:0] = INT(2^10 * N_{turn_sec} / N_{turn_prim}) else if (N_{turn_sec} / N_{turn_prim} is greater than 0.25) MFR_ TRANSFORMER _SCALE[15:11] =

Command name	Access	Length	Address	Bits	Description
					-11d = 15h MFR_ TRANSFORMER _SCALE[10:0] = INT(2^11 * N_{turn_sec} / N_{turn_prim}) else MFR_ TRANSFORMER _SCALE[15:11] = -12d = 14h MFR_ TRANSFORMER _SCALE[10:0] = INT(2^12 * N_{turn_sec} / N_{turn_prim}) **Note:** When **tlm1_vin_src_sel** = 2 (Loop 0 V_{OUT}), set PAGE 1 MFR_TRANSFORMER_SCALE = B3FFh (0.999) for correct scaling of V_{IN} telemetry
FW_CONFIG_TELEMETRY	RW	Block 21 bytes	C6h	[103:100]	**READ_DUTY_EXP**: Defines the LINEAR11 exponent for READ_DUTY
				[99:96]	**READ_IIN_EXP**: Defines the LINEAR11 exponent for READ_IIN
				[95:92]	**READ_IOUT_EXP**: Defines the LINEAR11 exponent for READ_IOUT
				[91:88]	**READ_VIN_EXP**: Defines the LINEAR11 exponent for READ_VIN
				[2:0]	**READ_POWER_EXP**: Defines the LINEAR11 exponent for READ_POUT and READ_PIN
MFR_SELECT_TEMPERATURE_SENSOR	RW	Byte	DCh	[2:0]	**Read_Temperature_1_Read_Temperature_2_source_select**: Defines the temp. sense input source selection for READ_TEMPERATURE_1 and READ_TEMPERATURE_2 commands. 0: READ_TEMPERATURE_1 = ATSEN, READ_TEMPERATURE_2 = BTSEN 1: READ_TEMPERATURE_1 = ATSEN, READ_TEMPERATURE_2 = ITSEN 2: READ_TEMPERATURE_1 = BTSEN, READ_TEMPERATURE_2 = ATSEN 3: READ_TEMPERATURE_1 =

Command name	Access	Length	Address	Bits	Description
					BTSEN, READ_TEMPERATURE_2 = ITSEN 4: READ_TEMPERATURE_1 = ITSEN, READ_TEMPERATURE_2 = ATSEN 5: READ_TEMPERATURE_1 = ITSEN, READ_TEMPERATURE_2 = BTSEN 6, 7: READ_TEMPERATURE_1 = ATSEN, READ_TEMPERATURE_2 = BTSEN

9 Fault handler

This chapter discusses the fault handler module and its main functionalities and relevant registers programming. All the PMBus commands mentioned in this section are described in Section 9.8.

The fault module is responsible for detecting faults and reporting them to the FW as well as initiating fault-based shutdown. The fault module block diagram is shown in Figure 76 and it consists of the following submodules:

- Loop fault (1 instance for each loop)
- Common fault
- Fault interrupts
- Fault priority encoding
- Fault shutdown

These submodules are described in detail within the following subsections.

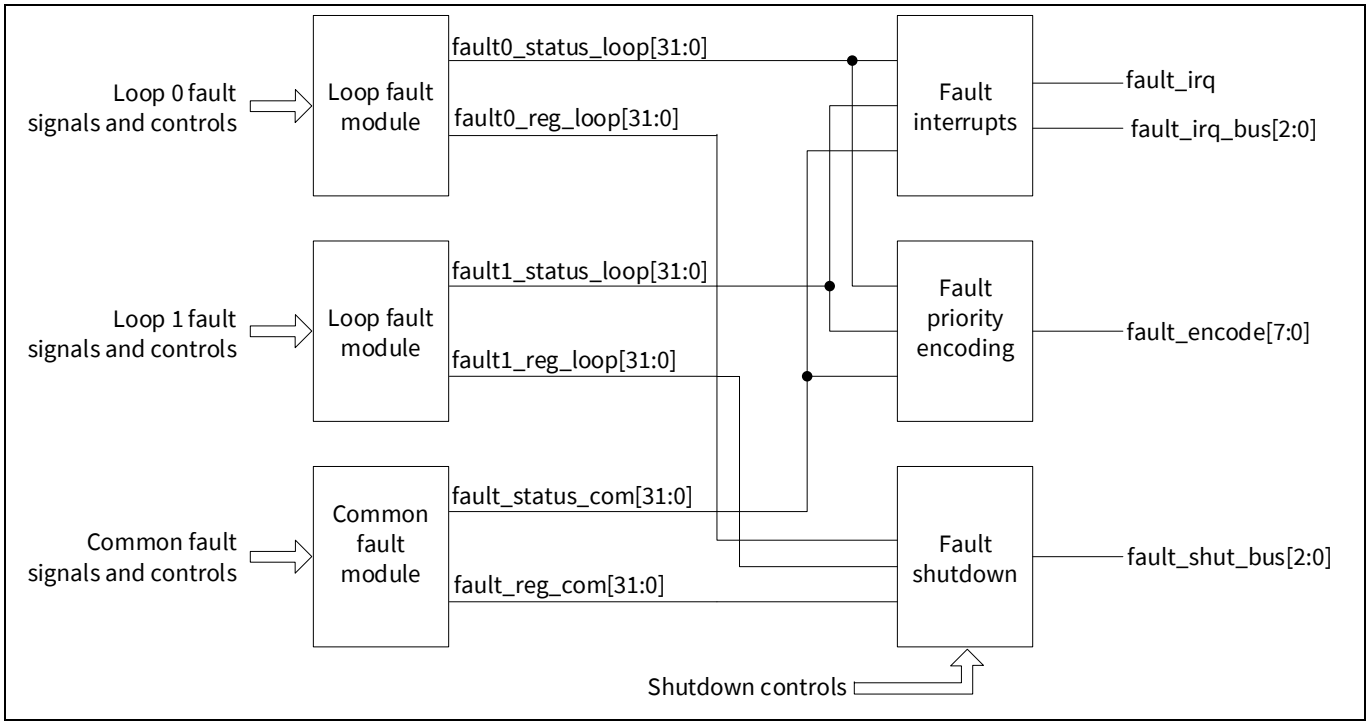

Figure 76 Fault module block diagram

9.1 Loop faults

The loop fault module is responsible for detecting and reporting faults that are specific to a loop, such as:

- Output and input voltage faults and warnings
- Output and input current faults and warnings
- Temperature faults and warnings

However, the loop fault module is not limited to these faults, and the full block diagram is shown in Figure 77. The Section 9.1.1 to 9.1.9 describe the loop fault submodules in detail.

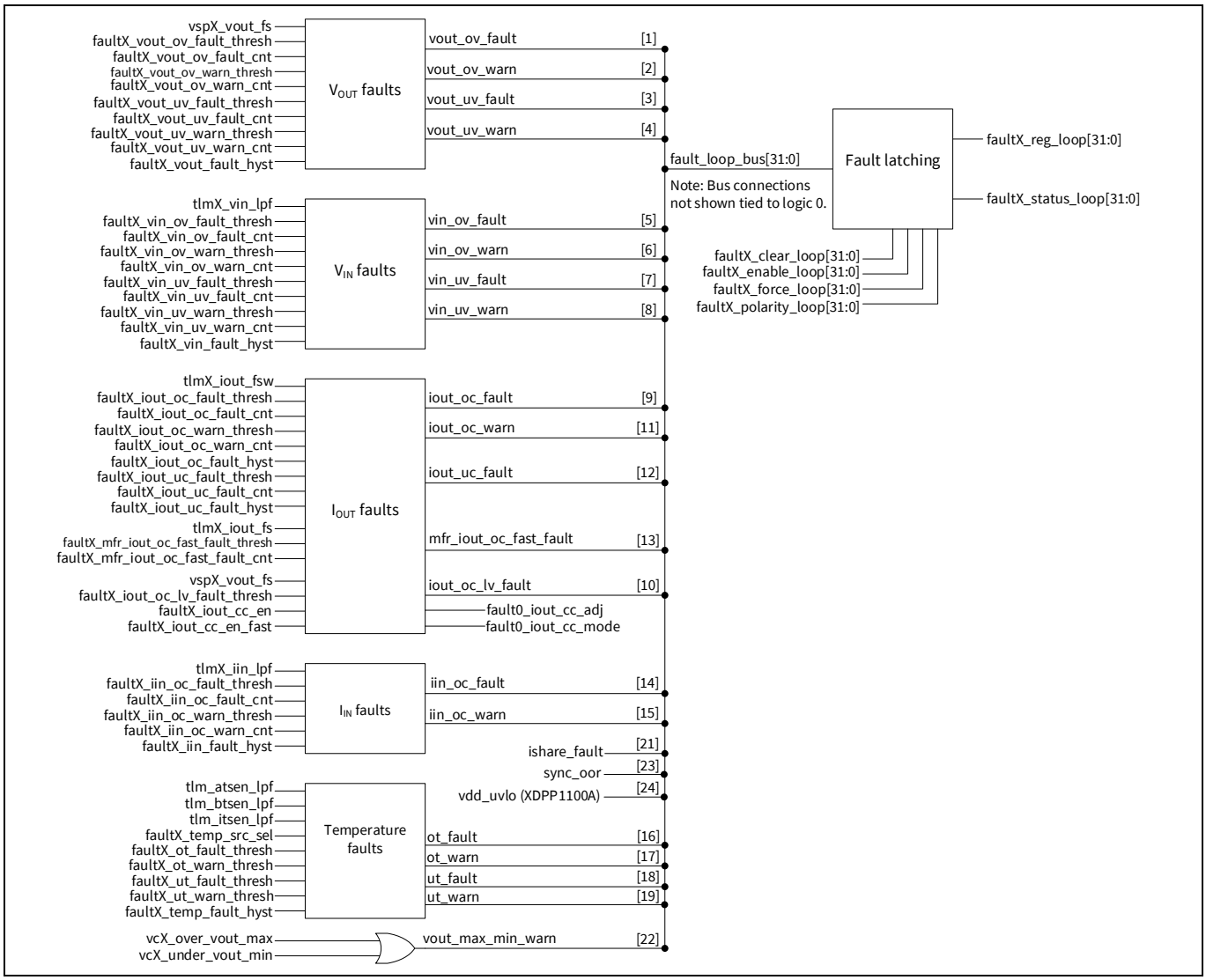

Figure 77 Loop fault submodule block diagram

9.1.1 Output voltage faults

The output voltage fault submodule is responsible for the detection of:

- Output overvoltage fault and warning
- Output undervoltage fault and warning

Output overvoltage fault

The output overvoltage fault is asserted when the VSP output in register **vspX_vout_fs** exceeds the fault threshold (**fault_vout_ov_fault_thresh**) for the number of samples (**fault_vout_ov_fault_cnt**) without falling below the threshold defined in Equation (9.1).

$$threshold = fault_vout_ov_fault_thresh - fault_vout_fault_hyst \quad (9.1)$$

The samples for the overvoltage fault definition do not need to be consecutive, but a single sample below the value in Equation (9.1) will reset the count and deassert the fault. The register **fault_vout_ov_fault_thresh** is referenced to VSEN and is computed by FW based on PMBus commands VOUT_OV_FAULT_LIMIT and VOUT_SCALE_LOOP.

Output overvoltage warning

The output overvoltage warning is asserted when the VSP output in register **vspX_vout_fs** exceeds the warning threshold (**fault_vout_ov_warn_thresh**) for the number of samples (**fault_vout_ov_warn_cnt**) without falling below the threshold defined in Equation (9.2).

$$threshold = fault_vout_ov_warn_thresh - fault_vout_warn_hyst \quad (9.2)$$

The samples for the overvoltage warning definition do not need to be consecutive, but a single sample below the value in Equation (9.2) will reset the count and deassert the warning. The register **fault_vout_ov_warn_thresh** is referenced to VSEN and is computed by FW based on PMBus commands VOUT_OV_WARN_LIMIT and VOUT_SCALE_LOOP.

Output undervoltage fault

The output undervoltage fault is asserted when the VSP output in register **vspX_vout_fs** falls below the fault threshold (**fault_vout_uv_fault_thresh**) for the number of samples (**fault_vout_uv_fault_cnt**) without exceeding the threshold defined in Equation (9.3).

$$threshold = fault_vout_uv_fault_thresh + fault_vout_fault_hyst \quad (9.3)$$

The samples for the undervoltage definition do not need to be consecutive but a single sample above the value in Equation (9.3) will reset the count and deassert the fault. The register **fault_vout_uv_fault_thresh** is referenced to VSEN and is computed by FW based on PMBus commands VOUT_UV_FAULT_LIMIT and VOUT_SCALE_LOOP.

Output undervoltage warning

The output undervoltage warning is asserted when the VSP output in register **vspX_vout_fs** falls below the warning threshold (**fault_vout_uv_warn_thresh**) for the number of samples (**fault_vout_uv_warn_cnt**) without exceeding the threshold defined in Equation (9.4).

$$threshold = fault_vout_uv_warn_thresh + fault_vout_fault_hyst \quad (9.4)$$

The samples for the undervoltage warning do not need to be consecutive but a single sample above the value in Equation (9.4) will reset the count and deassert the warning. The register **fault_vout_uv_warn_thresh** is referenced to VSEN and is computed by FW based on PMBus commands VOUT_UV_WARN_LIMIT and VOUT_SCALE_LOOP.

Output voltage fault-related PMBus commands

The output voltage faults and warnings are reported on the following PMBus commands:

- STATUS_BYTE
- STATUS_WORD
- STATUS_VOUT

It should be noted that VOUT_OV_FAULT and VOUT_UV_FAULT remain asserted in the PMBus STATUS command until VOUT_OV_WARNING and VOUT_UV_WARNING are deasserted. The XDPP1100 response to the output voltage faults is programmed through the PMBus commands:

- VOUT_OV_FAULT_RESPONSE
- VOUT_UV_FAULT_RESPONSE

The time unit for the delay portion of the output voltage RESPONSE commands is defined by PMBus command FW_CONFIG_FAULTS bits [7:6]. All these PMBus commands are described in Section 9.8.

9.1.2 Input voltage faults

The input voltage fault submodule is responsible for the detection of:

- Input overvoltage fault and warning
- Input undervoltage fault and warning

Input overvoltage fault

The input overvoltage fault is asserted when the telemetry output in register **tlmX_vin_lpf** exceeds the fault threshold (**fault_vin_ov_fault_thresh**) for the number of samples (**fault_vin_ov_fault_cnt**) without falling below the threshold defined in Equation (9.5).

$$threshold = fault_vin_ov_fault_thresh - fault_vin_fault_hyst \quad (9.5)$$

The samples for the overvoltage fault definition do not need to be consecutive, but a single sample below the value in Equation (9.5) will reset the count and deassert the fault. The register **fault_vin_ov_fault_thresh** is computed by FW based on PMBus command VIN_OV_FAULT_LIMIT.

Input overvoltage warning

The input overvoltage warning is asserted when the telemetry output in register **tlmX_vin_lpf** exceeds the warning threshold (**fault_vin_ov_warn_thresh**) for the number of samples (**fault_vin_ov_warn_cnt**) without falling below the threshold defined in Equation (9.6) (fault_vin_ov_warn_thresh – fault_vin_warn_hyst).

$$threshold = fault_vin_ov_warn_thresh - fault_vin_warn_hyst \quad (9.6)$$

The samples for the overvoltage warning definition do not need to be consecutive, but a single sample below the value in Equation (9.6) will reset the count and deassert the warning. The register **fault_vin_ov_warn_thresh** is computed by FW based on PMBus command VIN_OV_WARN_LIMIT.

Input undervoltage fault

The input undervoltage fault is asserted when the telemetry output in register **tlmX_vin_lpf** falls below the fault threshold (**fault_vin_uv_fault_thresh**) for the number of samples (**fault_vin_uv_fault_cnt**) without exceeding the threshold defined in Equation (9.7).

$$threshold = fault_vin_uv_fault_thresh + fault_vin_fault_hyst \quad (9.7)$$

The samples for the undervoltage definition do not need to be consecutive, but a single sample above the value in Equation (9.7) will reset the count and deassert the fault. The register **fault_vin_uv_fault_thresh** is computed by FW based on PMBus command VIN_UV_FAULT_LIMIT.

Input undervoltage warning

The input undervoltage warning is asserted when the telemetry output in register **tlmX_vin_lpf** falls below the warning threshold defined (**fault_vin_uv_warn_thresh**) for the number of samples (**fault_vin_uv_warn_cnt**) without exceeding the threshold defined in Equation (9.8).

$$threshold = fault_vin_uv_warn_thresh + fault_vin_fault_hyst \quad (9.8)$$

The samples for the undervoltage warning do not need to be consecutive but a single sample above the value in Equation (9.8) will reset the count and deassert the warning. The register **fault_vin_uv_warn_thresh** is computed by FW based on PMBus command VIN_UV_WARN_LIMIT.

Input voltage fault-related PMBus commands

The input voltage faults and warnings are reported on the following PMBus commands:

- STATUS_BYTE
- STATUS_WORD
- STATUS_INPUT

It should be noted that VIN_OV_FAULT and VIN_UV_FAULT remain asserted in the PMBus STATUS command until VIN_OV_WARNING and VIN_UV_WARNING are deasserted. The XDPP1100 response to the input voltage faults is programmed through the PMBus commands:

- VIN_OV_FAULT_RESPONSE
- VIN_UV_FAULT_RESPONSE

The time unit for the delay portion of the input voltage RESPONSE commands is defined by PMBus command FW_CONFIG_FAULTS bits [5:4]. All these PMBus commands are described in Section 9.8.

9.1.3 Output current faults

The output current fault submodule is responsible for detection of:

- Output overcurrent fault and warning
- Fast output overcurrent fault
- Output undercurrent fault
- Output overcurrent low-voltage fault

Output overcurrent fault

The output overcurrent fault is asserted when the filtered telemetry output in register **tlmX_iout_fsw** exceeds the fault threshold (**fault_iout_oc_fault_thresh**) for the number of samples (**fault_iout_oc_fault_cnt**) without falling below the threshold defined in Equation (9.9).

$$threshold = fault_iout_oc_fault_thresh - fault_iout_oc_fault_hyst \quad (9.9)$$

The samples for the overcurrent fault do not need to be consecutive, but a single sample below the value in Equation (9.9) will reset the count and deassert the warning. The register **fault_iout_oc_fault_thresh** is computed by FW based on PMBus command IOUT_OC_FAULT_LIMIT.

Output overcurrent warning

The output overcurrent warning is asserted when the filtered telemetry output in register **tlmX_iout_fsw** exceeds the warning threshold (**fault_iout_oc_warn_thresh**) for the number of samples (**fault_iout_oc_warn_cnt**) without falling below the threshold defined in Equation (9.10).

$$threshold = fault_iout_oc_warn_thresh - fault_iout_oc_fault_hyst \quad (9.10)$$

The samples for the overcurrent warning do not need to be consecutive, but a single sample below the value in Equation (9.10) will reset the count and deassert the fault. The register **fault_iout_oc_warn_thresh** is computed by FW based on PMBus command IOUT_OC_WARN_LIMIT.

Fast output overcurrent fault

The fast output overcurrent fault is asserted when the unfiltered telemetry output in register **tlmX_iout_fs** exceeds the fault threshold defined (**fault_mfr_iout_oc_fast_thresh**) for the number of samples (**fault_mfr_iout_oc_fast_cnt**) without falling below the threshold defined in Equation (9.11).

$$threshold = fault_mfr_iout_oc_fast_thresh - fault_iout_oc_fault_hyst \quad (9.11)$$

The samples for the fast overcurrent fault do not need to be consecutive, but a single sample below the value in Equation (9.11) will reset the count and deassert the fault. The register **fault_mfr_iout_oc_fast_thresh** is computed by FW based on PMBus command MFR_IOUT_OC_FAST_FAULT_LIMIT.

Output undercurrent fault

The output undercurrent fault is asserted when the filtered telemetry output in register **tlmX_iout_fsw** drops below the fault threshold (**fault_iout_uc_fault_thresh**) for the number of samples (**fault_iout_uc_fault_cnt**) without exceeding the threshold defined in Equation (9.12) (fault_iout_uc_fault_thresh + fault_iout_uc_fault_hyst).

$$threshold = fault_iout_uc_fault_thresh + fault_iout_uc_fault_hyst \quad (9.12)$$

The samples for the undercurrent fault do not need to be consecutive, but a single sample above the value in Equation (9.12) will reset the count and deassert the fault. The register **fault_iout_uc_fault_thresh** is computed by FW based on PMBus command IOUT_UC_FAULT_LIMIT.

Output current fault-related PMBus commands

The output current faults and warnings are reported on the following PMBus commands:

- STATUS_BYTE
- STATUS_WORD
- STATUS_IOUT
- STATUS_MFR_SPECIFIC

It should be noted that the IOUT_OC_FAULT will remain asserted in the PMBus STATUS commands until IOUT_OC_WARNING is deasserted. The XDPP1100 response to output current faults is programmed through the PMBus commands:

- IOUT_OC_FAULT_RESPONSE
- MFR_IOUT_OC_FAST_FAULT_RESPONSE
- IOUT_UC_FAULT_RESPONSE

The time unit for the delay portion of the output current RESPONSE commands is defined by PMBus command FW_CONFIG_FAULTS bits [3:2]. All these PMBus commands and their programming are described in Section 9.8.

Output overcurrent low-voltage fault

The XDPP1100 supports the current-limiting fault response to the following overcurrent faults:

- IOUT_OC_FAULT
- MFR_IOUT_OC_FAST_FAULT

In this mode, the XDPP1100 attempts to continue operation while maintaining the output current at IOUT_OC_FAULT_LIMIT by reducing the output voltage. If the output voltage drops below the threshold defined by register **fault_iout_oc_lv_fault_thresh**, IOUT_OC_LV_FAULT is declared.

9.1.4 Input current faults

The input current fault submodule is responsible for detection of input overcurrent fault and warning.

Input overcurrent fault

The input overcurrent fault is asserted when the filtered telemetry output in register **tlmX_iin_lpf** exceeds the fault threshold (**fault_iin_oc_fault_thresh**) for the number of samples (**fault_iin_oc_fault_cnt**) without falling below the threshold defined in Equation (9.13).

$$threshold = fault_iin_oc_fault_thresh - fault_iin_fault_hyst \quad (9.13)$$

The samples for the input overcurrent fault do not need to be consecutive, but a single sample below the value in Equation (9.13) will reset the count and deassert the fault. The register **fault_iin_oc_fault_thresh** is computed by FW based on PMBus command IIN_OC_FAULT_LIMIT.

Input overcurrent warning

The input overcurrent warning is asserted when the filtered telemetry output in register **tlmX_iin_lpf** exceeds the warning threshold defined (**fault_iin_oc_warn_thresh**) for the number of samples (**fault_iin_oc_warn_cnt**) without falling below the threshold defined in Equation (9.14).

$$threshold = fault_iin_oc_warn_thresh - fault_iin_fault_hyst \quad (9.14)$$

The samples for the input overcurrent fault do not need to be consecutive, but a single sample below the value in Equation (9.14) will reset the count and deassert the fault. The register **fault_iin_oc_warn_thresh** is computed by FW based on PMBus command IIN_OC_WARN_LIMIT.

Input current fault-related PMBus commands

The input current faults and warnings are reported on the following PMBus commands:

- STATUS_BYTE
- STATUS_WORD
- STATUS_IINPUT

It should be noted that the IIN_OC_FAULT will remain asserted in the PMBus STATUS commands until IIN_OC_WARNING is deasserted. The XDPP1100 response to input current faults is programmed through the PMBus commands IIN_OC_FAULT_RESPONSE. All of these PMBus commands and their programming are described in Section 9.8.

9.1.5 Power warnings

The XDPP1100 supports output and input power warnings. Since the power computation itself is implemented in FW, the warning detection is also performed in FW. The following PMBus commands are used for power warnings:

- POUT_OP_WARN_LIMIT defines the threshold for the output power warning
- PIN_OP_WARN_LIMIT defines the threshold for the input power warning

The output and input power warnings are reported on PMBus commands:

- STATUS_BYTE
- STATUS_WORD
- STATUS_IOUT
- STATUS_INPUT

All these PMBus commands and their programming are described in Section 9.8.

9.1.6 Temperature faults

The temperature fault submodule is responsible for the detection of:

- Overtemperature fault and warning
- Undertemperature fault and warning

The temperature source for the temperature faults and warnings is programmable through register **faultX_temp_src_sel**, as shown in Table 63. The temperature faults are based on comparisons in TSADC codes. The sensor TC is required to determine the polarity of the comparison, which can be also programmed through the register **faultX_temp_src_sel**.

Table 63 Temperature fault source select

faultX_temp_src_sel[2:0]	Source	Sensor temp. coefficient
0	TSEN	NTC
1	BTSEN	NTC
2	Internal	PTC
3	TSEN	PTC
4	BTSEN	PTC

The temperature faults and warnings are asserted and deasserted based on the sensor TC:

- PTC OT_FAULT:
  - Asserted when temp. is greater than **fault_ot_fault_thresh**
  - Deasserted when temp. is less than (**fault_ot_fault_thresh** + **fault_temp_fault_hyst**)
- PTC OT_WARN:
  - Asserted when temp. is greater than **fault_ot_warn_thresh**
  - Deasserted when temp. is less than (**fault_ot_warn_thresh** + **fault_temp_fault_hyst**)
- PTC UT_WARN:
  - Asserted when temp. is less than **fault_ut_warn_thresh**
  - Deasserted when temp. is greater than (**fault_ut_warn_thresh** - **fault_temp_fault_hyst**)
- PTC UT_FAULT:
  - Asserted when temp. is less than **fault_ut_fault_thresh**
  - Deasserted when temp. is greater than (**fault_ut_fault_thresh** - **fault_temp_fault_hyst**)
- NTC OT_FAULT:
  - Asserted when temp. is less than **fault_ot_fault_thresh**
  - Deasserted when temp. is greater than (**fault_ot_fault_thresh** + **fault_temp_fault_hyst**)

- NTC OT_WARN:
  - Asserted when temp. is less than **fault_ot_warn_thresh**
  - Deasserted when temp. is greater than (**fault_ot_warn_thresh** + **fault_temp_fault_hyst**)
- NTC UT_WARN:
  - Asserted when temp. is greater than **fault_ut_warn_thresh**
  - Deasserted when temp. is less than (**fault_ut_warn_thresh** - **fault_temp_fault_hyst**)
- NTC UT_FAULT:
  - Asserted when temp. is greater than **fault_ut_fault_thresh**
  - Deasserted when temp. is less than (fault_ut_fault_thresh - fault_temp_fault_hyst)

Register **fault_temp_fault_hyst** should be set to a negative value for PTC sensors and a positive value for NTC sensors. The above-mentioned registers are computed by FW as follows:

- **fault_ot_fault_thresh** based on OT_FAULT_LIMIT and the FW-implemented temperature LUT
- **fault_ot_warn_thresh** based on OT_WARN_LIMIT and the FW-implemented temperature LUT
- **fault_ut_fault_thresh** based on UT_FAULT_LIMIT and the FW-implemented temperature LUT
- **fault_ut_warn_thresh** based on UT_WARN_LIMIT and the FW-implemented temperature LUT.

The temperature faults and warnings are reported on the following PMBus commands:

- STATUS_BYTE
- STATUS_WORD
- STATUS_TEMP

It should be noted that the OT_FAULT and UT_FAULT remain asserted in the PMBus STATUS commands until OT_WARNING and UT_WARNING are deasserted. The XDPP1100 response to temperature faults is programmed through the PMBus commands:

- OT_FAULT_RESPONSE
- UT_FAULT_RESPONSE

The time unit for the delay portion of the input voltage RESPONSE commands is defined by PMBus command FW_CONFIG_FAULTS bits [1:0]. All PMBus commands and their programming are described in Section 9.8.

9.1.7 Current sharing fault

XDPP1100 asserts a current sharing fault under the conditions described in Section 10.1. The current sharing fault is reported on PMBus command STATUS_IOUT bit [3].

9.1.8 V_{OUT_MAX}, V_{OUT_MIN} warning

XDPP1100 supports the VOUT_MAX and VOUT_MIN warnings as described in Section 5.5. The VOUT_MAX_MIN warning is reported on PMBus command STATUS_VOUT bit [3].

9.1.9 Sync fault

XDPP1100 asserts the sync fault when the external SYNC input is out of range as described in Section 7.1.2. The sync fault is reported on PMBus command STATUS_MFR_SPECIFIC bit [7].

9.1.10 V_{DD} undervoltage lock out fault (XDPP1100A only)

XDPP1100A asserts the undervoltage lockout (UVLO) fault when the external 3.3 V supply drops below the VDD UVLO fault threshold. This threshold is fixed and is defined in the XDPP1100A datasheet. This fault requires a FW patch to activate, and it is enabled for shutdown by default with the required patch. The user must set the PMBus command FW_CONFIG_FAULTS field Fault_enable_mask_loop_hw bit [24] to disable (logic 1) if this fault is not required. The VDD UVLO fault is reported on the PMBus command STATUS_MFR_SPECIFIC bit [2].

9.1.11 T_{ON_MAX}, T_{OFF_MAX} faults

The XDPP1100 supports the TON_MAX fault and TOFF_MAX warning through FW. For these, the following PMBus commands are used to set limits:

- TON_MAX_FAULT_LIMIT sets an upper limit on how long the XDPP1100 attempts to power up the output without reaching the output undervoltage limit. If this limit is exceeded the response defined by PMBus command TON_MAX_FAULT_RESPONSE is taken
- TOFF_MAX_WARN_LIMIT sets an upper limit on how long the XDPP1100 attempts to power down the output without reaching 12.5 percent of the programmed output voltage at the time the XDPP1100 is turned off. Since this is only a warning, no response is required

The TON_MAX fault and TOFF_MAX warning are reported on STATUS_VOUT bits [2] and [1] respectively.

9.1.12 Loop fault latching

The loop fault latching submodule is responsible for registering and holding the occurrence of loop faults for use by the shutdown submodule and reporting to the FW. As was shown in Figure 77, the outputs from the fault detection submodules are gathered together to form the fault_loop_bus signal, which is the data input to the loop fault latching submodule. A block diagram of the loop fault latching function is shown in Figure 78.

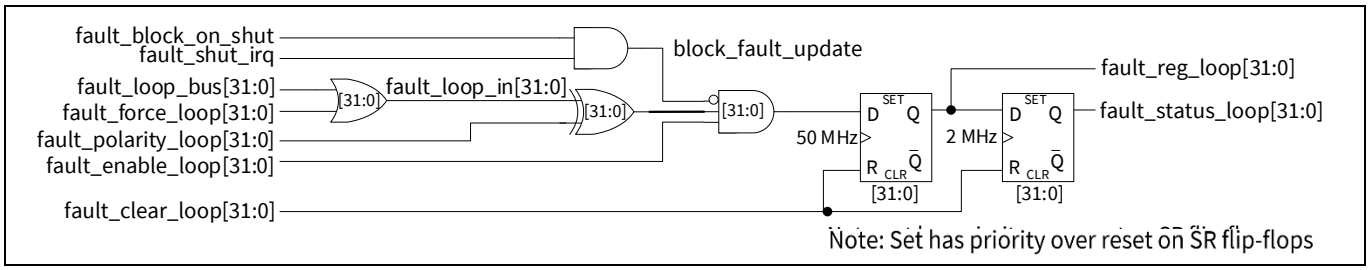

Figure 78 Loop fault latching block diagram

Several registers are provided to allow the FW to control if and how the loop faults are reported on **fault_reg_loop** and **fault_status_loop**. These registers are:

- **fault_enable_loop**, programmed via PMBus command FW_CONFIG_FAULTS bits [103:72], enables individual faults for reporting on **fault_reg_loop** and **fault_status_loop**
- **fault_polarity_loop** controls the polarity, active high or low, of individual faults reported on **fault_reg_loop** and **fault_status_loop**
- **fault_force_loop** is bitwise ORed with fault_loop_bus to provide a method for FW to trigger the HW-based loop faults
- **fault_clear_loop** allows individual faults to be cleared on **fault_reg_loop** and **fault_status_loop**
- **fault_block_on_shut** prevents additional fault reporting after an initial shutdown fault has occurred

Table 64 shows the shared bit-to-fault mapping that applies to registers:

- **fault_enable_loop**
- **fault_polarity_loop**
- **fault_force_loop**
- **fault_clear_loop**
- **fault_reg_loop**
- **fault_status_loop**

Table 64 Bit mapping of loop faults

Bit	Fault	Bit	Fault
[0]	Reserved	[12]	IOUT_UC_FAULT
[1]	VOUT_OV_FAULT	[13]	MFR_IOUT_OC_FAST_FAULT
[2]	VOUT_OV_WARN	[14]	IIN_OC_FAULT
[3]	VOUT_UV_FAULT	[15]	IIN_OC_WARN
[4]	VOUT_UV_WARN	[16]	OT_FAULT
[5]	VIN_OV_FAULT	[17]	OT_WARN
[6]	VIN_OV_WARN	[18]	UT_FAULT
[7]	VIN_UV_FAULT	[19]	UT_WARN
[8]	VIN_UV_WARN	[20]	POWER_LIMIT_MODE
[9]	IOUT_OC_FAULT	[21]	ISHARE_FAULT
[10]	IOUT_OC_LV_FAULT	[22]	VOUT_MAX_MIN_WARN
[11]	IOUT_OC_WARN	[23]	SYNC_FAULT
[24]	VDD_UVLO (XDPP1100A)	–	–

9.2 Common faults

The common fault submodule is responsible for reporting faults from functions that are not specific to one loop. This includes but is not limited to faults such as:

- Current sense tracking fault
- PCL fault
- Short-circuit protection fault
- Flux balance fault
- Open VS protection faults

The common fault submodule reports the faults as shown in Figure 79. However, it does not handle the detection of common faults. This is generally handled within the module related to the fault (e.g., the flux balance fault is detected within the flux balance module). The following Section 9.2.1 through 9.2.6 describe the details of the common faults shown in the block diagram.

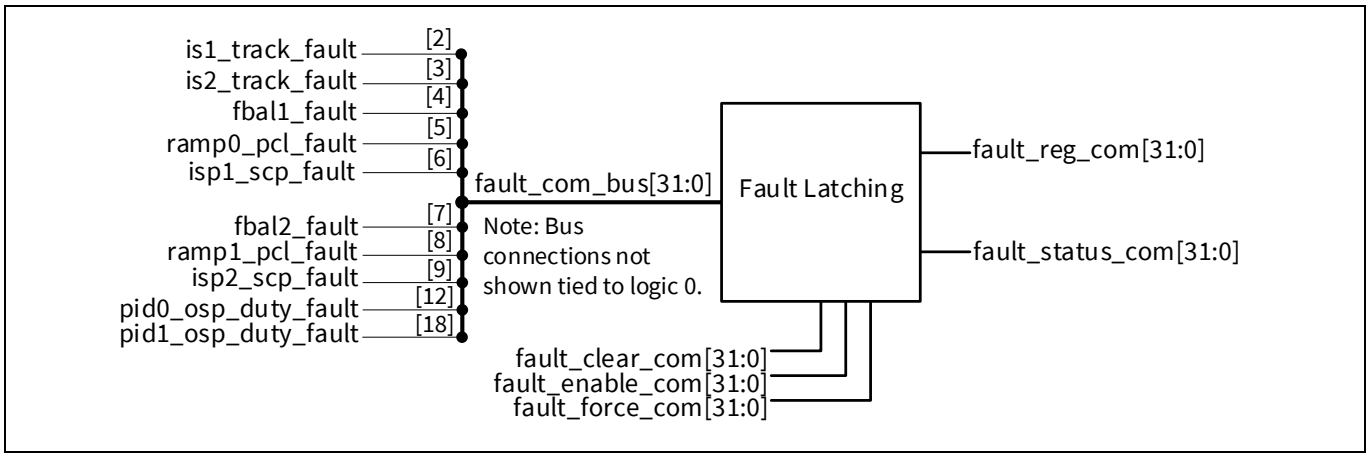

Figure 79 Common fault submodule block diagram

9.2.1 Current sense tracking fault

XDPP1100 supports a current sense error tracking fault as described in Section 3.2.2.3. The error tracking fault is bitwise ORed with all common faults and reported on STATUS_MFR_SPECIFIC bit [4]. The individual fault can be determined by reading the register **fault_status_com**.

9.2.2 Peak current limit fault

XDPP1100 supports a PCL fault as described in Section 3.3.2. The PCL fault is bitwise ORed with all common faults and reported on STATUS_MFR_SPECIFIC bit [4]. The individual fault can be determined by reading the register **fault_status_com**.

9.2.3 SCP fault

XDPP1100 supports a short-circuit protection fault as described in Section 3.3.4. The short-circuit protection fault is bitwise ORed with all common faults and reported on STATUS_MFR_SPECIFIC bit [4]. The individual fault can be determined by reading register **fault_status_com**.

9.2.4 Flux balance fault

XDPP1100 supports a flux balance fault as described in Section 12.5. The flux balance fault is bitwise ORed with all common faults and reported on STATUS_MFR_SPECIFIC bit [4]. The individual fault can be determined by reading register **fault_status_com**.

9.2.5 Open VS fault protection

XDPP1100 is capable of detecting open or missing sense resistors on the VSEN/VREF, VRSEN/VRREF and BVSEN_BVRSEN/BVREF_BVRREF input pairs. The sense configuration consists of an upper and lower sense resistor pair, dividing the sensed voltage into a range suitable for the XDPP1100 inputs, as was shown in Figure 2. In a typical sense configuration, the:

- Upper sense resistor, R_H, is connected between V_{OUT} or V_{RECT} and the sense input
- Lower sense resistor, R_L, is connected between the sense input and the reference input
- Reference input is connected to the V_{OUT} or V_{RECT} return (GND)

Detection of a missing R_H sense resistor is handled indirectly by different methods depending on the voltage being sensed:

- When V_{OUT} is sensed on VSEN or BVSEN, the missing R_H is detected through the PID compensators' open sense fault detection as described in Section 6.4
- When V_{RECT} or V_{IN} is sensed on VRSEN or BVRSEN, the missing R_H leads to an input undervoltage fault or input voltage insufficient for operation (i.e., V_{IN} is less than VIN_ON)

The detection of a missing R_L sense resistor can lead to the following faults:

- Open sense fault
- Open reference sense fault

The open sense fault is asserted if the voltage difference between:

- FW measured pre-bias voltage on the sense input during TON_DELAY period (prior to TON_RISE) and
- Remeasured pre-bias voltage subsequent to the enabling of 50 µA current source on the VSEN/VRSEN/BVSEN input

exceeds the programmable threshold value in register **vsp_osp_thresh**. In case the open sense fault is not asserted, the FW enables a 50 µA current source on the VREF/VRREF/BVREF input and measures the voltage on the sense input. If the voltage rises above the sense input, it leads to a 0 reading on the ADC output. This refers to the fact that the reference input is open, and the open reference sense fault is asserted. If neither fault is asserted, the XDPP1100 proceeds to TON_RISE.

9.2.6 Common fault latching

The common fault latching submodule is responsible for registering and holding the occurrence of common faults for use by the shutdown submodule and reporting to the FW. As was shown in Figure 79, the fault inputs to the common fault submodule are gathered together to form the fault_com_bus signal, which is the data input to the common fault latching submodule.

A block diagram of the common fault latching function is illustrated in Figure 80.

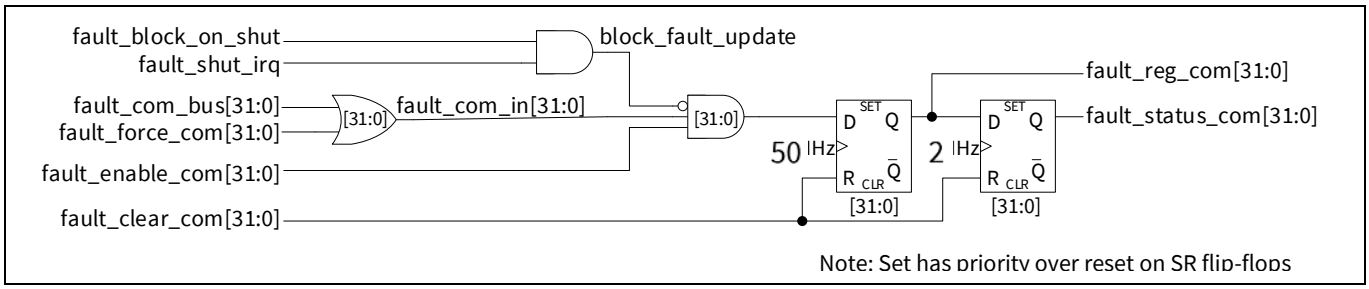

Figure 80 Common fault latching block diagram

Several registers are provided to allow the FW to control if common faults are reported on **fault_reg_com** and **fault_status_com** and how. These are:

- **fault_enable_com**, programmed through the PMBus command FW_CONFIG_FAULTS bits [167:136], enables individual faults for reporting on **fault_reg_com** and **fault_status_com**
- **fault_force_com** is bitwise ORed with fault_com_bus to provide a method for FW to trigger the HW-based common faults
- **fault_clear_com** allows individual faults to be cleared on **fault_reg_com** and **fault_status_com**

- **fault_block_on_shut** prevents additional fault reporting after an initial shutdown fault has occurred

Table 65 shows the shared bit-to-fault mapping that applies to registers:

- **fault_enable_com**
- **fault_force_com**
- **fault_clear_com**
- **fault_reg_com**
- **fault_status_com**

Table 65 Bit mapping of common faults

Bit	Fault	Bit	Fault
[0]	Reserved	[10]	Reserved
[1]	Reserved	[11]	Reserved
[2]	ISEN tracking fault	[12]	PID0 OSP duty fault
[3]	BISEN tracking fault	[13]	Reserved
[4]	Flux balance 1 fault	[14]	Reserved
[5]	ISEN PCL fault	[15]	Reserved
[6]	ISEN SCP fault	[16]	Reserved
[7]	Flux balance 2 fault	[17]	Reserved
[8]	BISEN PCL fault	[18]	PID1 OSP duty fault
[9]	BISEN SCP fault	–	–

9.3 Fault interrupts (IRQ)

The fault interrupt generation is shown in Figure 81. From the figure it can be observed that three fault status registers:

- **fault0_status_loop**
- **fault1_status_loop**
- **fault_status_com**

are individually bitwise ORed to produce three interrupt signals:

- fault0_irq
- fault1_irq
- faultcm_irq

These three IRQ signals are ORed together to form the interrupt going to the CPU. They are also gathered together to form the register **fault_irq_bus**, allowing the FW to determine in which status register to look for the fault.

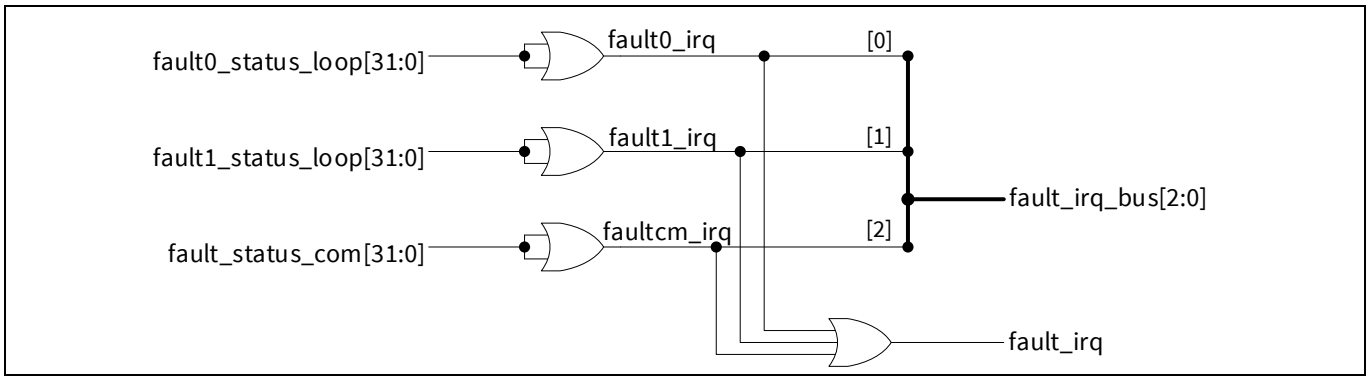

Figure 81 Fault interrupt block diagram

9.4 Faults priority encoding

A second and more practical method for determining the fault that has occurred is via the fault priority encoding module. A portion of the priority encoder LUT is shown in Figure 82. The LUT input is formed from a 96-bit wide bus formed by concatenation of the following status registers:

- **fault_status_com[31:0**]
- **fault1_status_loop[31:0**]
- **fault0_status_loop[31:0]**

As shown in the figure, the highest priority is given to the LSB of the input bus (i.e., **fault0_status_loop[0]**) and lowest priority is given to the MSB of the input bus (i.e., **fault_status_com[31]**). The output is provided on read-only register **fault_encode**. Note that **fault_encode** returns 0d for both the no-fault case and the **fault0_status_loop[0]** = 1 case. To distinguish between the two cases the FW can read the value of **fault0_status_loop** or **fault_irq_bus**.

{ fault_status_com[31:0], fault1_status_loop[31:0], fault0_status_loop[31:0] }	fault_encode[7:0]
00000000000000000000000000000000_00000000000000000000000000000000_00000000000000000000000000000000	0d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx1	0d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx10	1d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxx100	2d
...	
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xx100000000000000000000000000000	29d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_x100000000000000000000000000000	30d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_1000000000000000000000000000000	31d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx1_0000000000000000000000000000000	32d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx10_0000000000000000000000000000000	33d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxx100_0000000000000000000000000000000	34d
...	
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_xx100000000000000000000000000000_0000000000000000000000000000000	61d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_x1000000000000000000000000000000_0000000000000000000000000000000	62d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx_10000000000000000000000000000000_0000000000000000000000000000000	63d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx1_0000000000000000000000000000000_0000000000000000000000000000000	224d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx10_0000000000000000000000000000000_0000000000000000000000000000000	225d
xxxxxxxxxxxxxxxxxxxxxxxxxxxxx100_0000000000000000000000000000000_0000000000000000000000000000000	226d
...	
xx100000000000000000000000000000_0000000000000000000000000000000_0000000000000000000000000000000	253d
x1000000000000000000000000000000_0000000000000000000000000000000_0000000000000000000000000000000	254d
10000000000000000000000000000000_0000000000000000000000000000000_0000000000000000000000000000000	255d

Figure 82 Fault priority encoding

9.5 Fault shutdown

The fault shutdown submodule is responsible for:

- Detecting the occurrence of a shutdown-enabled fault
- Signaling to the PWM generator that the PWM outputs should be disabled
- Signaling to the FW that a fault-based shutdown has occurred

A block diagram of the fault shutdown submodule is shown in Figure 83.

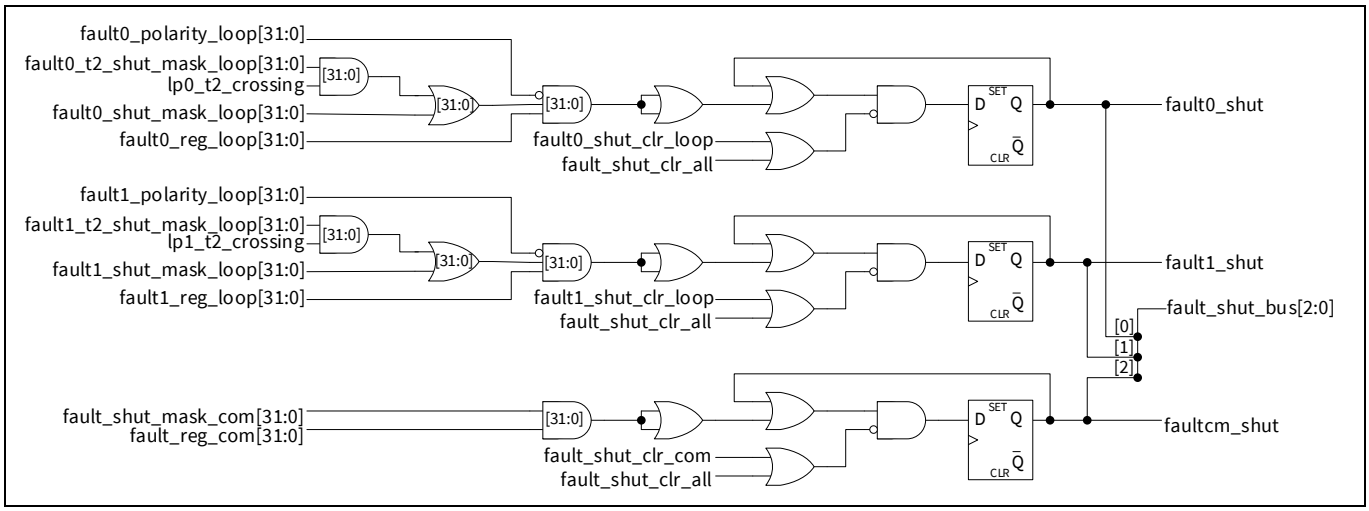

Figure 83 Fault shutdown block diagram

XDPP1100 supports two different fault-based shutdown timings:

- Aligned to the PWM ramp t2 time
- Immediate (non-t2 aligned)

Figure 84 shows example timing for these two cases.

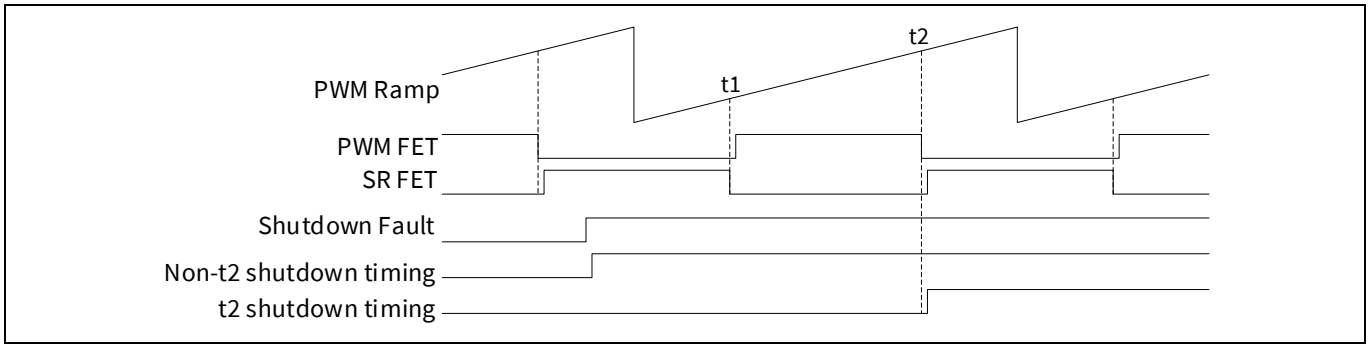

Figure 84 Immediate (non-t2) vs. t2 aligned shutdown timing

The immediate shutdown is enabled through the following registers:

- **fault0_shut_mask_loop** for the Loop 0 faults
- **fault1_shut_mask_loop** for the Loop 1 faults
- **fault_shut_mask_com** for the common faults

Registers **fault0_shut_mask_loop** and **fault1_shut_mask_loop** are programmed by FW based on the PMBus fault RESPONSE commands.

The t2-aligned shutdown is enabled through the following registers:

- **fault0_t2_shut_mask_loop** for the Loop 0 faults
- **fault1_t2_shut_mask_loop** for the Loop 1 faults

Register **faultX_t2_shut_mask_loop** is programmed through the PMBus command FW_CONFIG_FAULTS bits [199:168]. The t2-aligned shutdown is not available for common fault shutdown.

When a shutdown-enabled fault occurs, the event is registered as fault0_shut, fault1_shut or faultcm_shut based on the fault type:

- fault0_shut and faultcm_shut turn off the Loop 0 PWM outputs
- fault1_shut and faultcm_shut turn off the Loop 1 PWM outputs

The status of the shutdown signals is available on the read-only register **fault_shut_bus**.

Fault events remain registered until they are cleared by FW, typically after the FW has completed its shutdown tasks (e.g., disable unnecessary modules, reset target voltage, etc.). The following registers are used for clearing the faults:

- **fault0_shut_clr_loop** clears the Loop 0 shutdown event
- **fault1_shut_clr_loop** clears the Loop 1 shutdown event
- **fault_shut_clr_com** clears the common shutdown event
- **fault_shut_clr_all** clears all shutdown events

9.6 Fault pin mapping

The 40-pin version of the XDPP1100 provides two GPIO-based fault pins:

- FAULT1
- FAULT2

By default GPIO0[2] is assigned to FAULT1 and it outputs Loop 0 faults as selected through PMBus page 0 FW_CONFIG_FAULTS[71:8]. Correspondingly, the GPIO1[2] is assigned to FAULT2 and it outputs Loop 1 faults as selected through PMBus Page 1 FW_CONFIG_FAULTS[71:8].

9.7 Fault registers

The relevant fault registers and their descriptions are provided in Table 66.

Table 66 Fault register description

Peripheral	Field name	Access	Address	Bits	Description
vsen	vsp_osp_thresh	RW	7000_080Ch (VSEN) 7000_0C0Ch (VRSEN) 7000_100Ch	[11:0]	Open sense protection fault threshold. LSB = 1.25 mV, range = 0.0 to 5.11875 V

Peripheral	Field name	Access	Address	Bits	Description
			(BV(R)SEN)		
fault	fault_vout_ov_fault_cnt	RW	7000_3C00h (Loop 0) 7000_4000h (Loop 1)	[1:0]	Output overvoltage fault count. Defines the number of 50 MHz samples the output voltage must exceed the fault threshold by in order to assert a fault. 0: 1 sample 1: 2 samples 2: 4 samples 3: 8 samples
fault	fault_vout_ov_warn_cnt	RW	7000_3C00h (Loop 0) 7000_4000h (Loop 1)	[3:2]	Output overvoltage warning count. Defines the number of 50 MHz samples the output voltage must exceed the warn threshold by in order to assert a warning. 0: 1 sample 1: 2 samples 2: 4 samples 3: 8 samples
fault	fault_vout_uv_fault_cnt	RW	7000_3C00h (Loop 0) 7000_4000h (Loop 1)	[5:4]	Output undervoltage fault count. Defines the number of 50 MHz samples the output voltage must exceed the fault threshold by in order to assert a fault. 0: 1 sample 1: 2 samples 2: 4 samples 3: 8 samples
fault	fault_vout_uv_warn_cnt	RW	7000_3C00h (Loop 0) 7000_4000h (Loop 1)	[7:6]	Output undervoltage warning count. Defines the number of 50 MHz samples the output voltage must exceed the warn threshold by in order to assert a warning. 0: 1 sample 1: 2 samples 2: 4 samples 3: 8 samples
fault	fault_iout_oc_fault_cnt	RW	7000_3C00h (Loop 0) 7000_400	[12:8]	Output overcurrent fault count. Defines the number of consecutive switching cycles (T_{switch}) the low-pass filtered current must exceed the fault

Peripheral	Field name	Access	Address	Bits	Description
			0h (Loop 1)		threshold by in order to assert a fault. Count = fault_iout_oc_fault_cnt + 1
fault	fault_iout_oc_warn_cnt	RW	7000_3C00h (Loop 0) 7000_4000h (Loop 1)	[17:13]	Output overcurrent warning count. Defines the number of consecutive switching cycles (T_{switch}) the low-pass filtered current must exceed the warn threshold by in order to assert a warning. Count = fault_iout_oc_warn_cnt + 1
fault	fault_mfr_iout_oc_fast_cnt	RW	7000_3C00h (Loop 0) 7000_4000h (Loop 1)	[22:18]	MFR_IOUT_OC_FAST overcurrent fault count. Defines the number of consecutive switching cycles (T_{switch}) the cycle-averaged current must exceed the fault threshold by in order to assert a fault. Count = fault_mfr_iout_oc_fast_cnt + 1
fault	fault_iout_uc_fault_cnt	RW	7000_3C00h (Loop 0) 7000_4000h (Loop 1)	[27:23]	Output undercurrent fault count. Defines the number of consecutive switching cycles (T_{switch}) the cycle-averaged current must exceed the fault threshold by in order to assert a fault. Count = fault_iout_uc_fault_cnt + 1
fault	fault_vin_ov_fault_cnt	RW	7000_3C04h (Loop 0) 7000_4004h (Loop 1)	[1:0]	Input overvoltage fault count. Defines the number of consecutive switching cycles (T_{switch}) the input voltage must exceed the fault threshold by in order to assert a fault. 0: 1 T_{switch} 1: 2 T_{switch} 2: 4 T_{switch} 3: 8 T_{switch}
fault	fault_vin_ov_warn_cnt	RW	7000_3C04h (Loop 0) 7000_400	[3:2]	Input overvoltage warning count. Defines the number of consecutive switching cycles (T_{switch}) the input voltage must exceed the warn threshold by

Peripheral	Field name	Access	Address	Bits	Description
			4h (Loop 1)		in order to assert a warning. 0: 1 T_{switch} 1: 2 T_{switch} 2: 4 T_{switch} 3: 8 T_{switch}
fault	fault_vin_uv_fault_cnt	RW	7000_3C04h (Loop 0) 7000_4004h (Loop 1)	[5:4]	Input undervoltage fault count. Defines the number of consecutive switching cycles (T_{switch}) the input voltage must exceed the fault threshold by in order to assert a fault. 0: 1 T_{switch} 1: 2 T_{switch} 2: 4 T_{switch} 3: 8 T_{switch}
fault	fault_vin_uv_warn_cnt	RW	7000_3C04h (Loop 0) 7000_4004h (Loop 1)	[7:6]	Input undervoltage warning count. Defines the number of consecutive switching cycles (T_{switch}) the input voltage must exceed the warn threshold by in order to assert a warning. 0: 1 T_{switch} 1: 2 T_{switch} 2: 4 T_{switch} 3: 8 T_{switch}
fault	fault_iin_oc_fault_cnt	RW	7000_3C04h (Loop 0) 7000_4004h (Loop 1)	[12:8]	Input overcurrent fault count. Defines the number of consecutive switching cycles (T_{switch}) the low-pass filtered input current must exceed the fault threshold by in order to assert a fault. Count = fault_iin_oc_fault_cnt + 1
fault	fault_iin_oc_warn_cnt	RW	7000_3C04h (Loop 0) 7000_4004h (Loop 1)	[17:13]	Input overcurrent warning count. Defines the number of consecutive switching cycles (T_{switch}) the low-pass filtered input current must exceed the warn threshold by in order to assert a warning. Count = fault_iin_oc_warn_cnt + 1
fault	fault_vin_fault_hyst	RW	7000_3C04h (Loop 0) 7000_400	[23:18]	Input voltage (V_{IN}) fault (warning) deassertion hysteresis. Internally, the V_{IN} OV fault is asserted when V_{IN}

Peripheral	Field name	Access	Address	Bits	Description
			4h (Loop 1)		exceeds VIN_OV_FAULT_LIMIT for fault_vin_ov_fault_cnt samples without dropping below (VIN_OV_FAULT_LIMIT-fault_vin_fault_hyst). The samples above VIN_OV_FAULT_LIMIT need not be consecutive but a single sample below (VIN_OV_FAULT_LIMIT-fault_vin_fault_hyst) will reset the count. For the typical case this parameter should be set to a positive voltage. This hysteresis parameter applies to all V_{IN} faults and warnings as shown below. VIN_OV_FAULT: Assert when V_{IN} is greater than VIN_OV_FAULT_LIMIT Deassert when V_{IN} is less than or equal to (VIN_OV_FAULT_LIMIT-fault_vin_fault_hyst) VIN_OV_WARN: Assert when V_{IN} is greater than VIN_OV_WARN_LIMIT Deassert when V_{IN} is less than or equal to (VIN_OV_WARN_LIMIT-fault_vin_fault_hyst) VIN_UV_WARN: Assert when V_{IN} is less than VIN_OV_WARN_LIMIT Deassert when V_{IN} is greater than or equal to (VIN_UV_WARN_LIMIT+fault_vin_fault_hyst) VIN_UV_FAULT: Assert when V_{IN} is less than VIN_OV_FAULT_LIMIT Deassert when V_{IN} is greater than or equal to (VIN_UV_FAULT_LIMIT+fault_vin_fault_hyst) LSB = 125 mV, range = -4.0 to +3.875 V

Peripheral	Field name	Access	Address	Bits	Description
fault	fault_vout_fault_hyst	RW	7000_3C08h (Loop 0) 7000_4008h (Loop 1)	[4:0]	Output voltage (V_{OUT}) fault (warning) deassertion hysteresis referenced to the VSEN input (i.e., after scaling by VOUT_SCALE_LOOP). Internally, the V_{OUT} OV fault is asserted when V_{OUT} exceeds VOUT_OV_FAULT_LIMIT for fault_vout_ov_fault_cnt samples without dropping below (VOUT_OV_FAULT_LIMIT-fault_vout_fault_hyst). The samples above VOUT_OV_FAULT_LIMIT need not be consecutive, but a single sample below (VOUT_OV_FAULT_LIMIT-fault_vout_fault_hyst) will reset the count. For the typical case this parameter should be set to a positive voltage. This hysteresis parameter applies to all V_{OUT} faults and warnings as shown below. VOUT_OV_FAULT: Assert when V_{OUT} is greater than VOUT_OV_FAULT_LIMIT Deassert when V_{OUT} is less than or equal to (VOUT_OV_FAULT_LIMIT-fault_vout_fault_hyst) VOUT_OV_WARN: Assert when V_{OUT} is greater than VOUT_OV_WARN_LIMIT Deassert when V_{OUT} is less than or equal to (VOUT_OV_WARN_LIMIT-fault_vout_fault_hyst) VOUT_UV_WARN: Assert when V_{OUT} is less than VOUT_OV_WARN_LIMIT Deassert when V_{OUT} is greater than or equal to (VOUT_UV_WARN_LIMIT+fault_vout_fault_hyst) VOUT_UV_FAULT:

Peripheral	Field name	Access	Address	Bits	Description
					Assert when V_{OUT} is less than VOUT_OV_FAULT_LIMIT Deassert when V_{OUT} is greater than or equal to (VOUT_UV_FAULT_LIMIT+fault_vout_fault_hyst) LSB = 5 mV, range = -80 to +75 mV
fault	fault_iin_fault_hyst	RW	7000_3C08h (Loop 0) 7000_4008h (Loop 1)	[9:5]	Input current (I_{IN}) fault (warning) deassertion hysteresis. Internally, the I_{IN} OV fault is asserted when I_{IN} exceeds IIN_OC_FAULT_LIMIT for fault_iin_oc_fault_cnt samples without dropping below (IIN_OC_FAULT_LIMIT-fault_iin_fault_hyst). The samples above IIN_OC_FAULT_LIMIT need not be consecutive, but a single sample below (IIN_OC_FAULT_LIMIT-fault_iin_fault_hyst) will reset the count. For the typical case this parameter should be set to a positive current. This hysteresis parameter applies to all I_{IN} faults and warnings as shown below. IIN_OC_FAULT: Asserted when I_{IN} is greater than IIN_OC_FAULT_LIMIT Deasserted when I_{IN} is less than or equal to (IIN_OC_FAULT_LIMIT-fault_iin_fault_hyst) IIN_OC_WARN: Asserted when I_{IN} is greater than IIN_OC_WARN_LIMIT Deasserted when I_{IN} is less than or equal to (IIN_OC_WARN_LIMIT-fault_iin_fault_hyst) LSB = 0.25 A, range = -4 to 3.75 A
fault	fault_iout_oc_fault_hyst	RW	7000_3C08h (Loop	[15:10]	Output overcurrent (IOUT_OC) fault (warning) deassertion

Peripheral	Field name	Access	Address	Bits	Description
			0) 7000_4008h (Loop 1)		hysteresis. Internally, the I_{OUT} OC fault is asserted when I_{OUT} exceeds IOUT_OC_FAULT_LIMIT for fault_iout_oc_fault_cnt samples without dropping below (IOUT_OC_FAULT_LIMIT-fault_iout_oc_fault_hyst). The samples above IOUT_OC_FAULT_LIMIT need not be consecutive, but a single sample below (IOUT_OC_FAULT_LIMIT-fault_iout_oc_fault_hyst) will reset the count. For the typical case this parameter should be set to a positive current. This hysteresis parameter applies to all I_{OUT} OC faults and warnings as shown below. IOUT_OC_FAULT: Asserted when I_{OUT} is greater than IOUT_OC_FAULT_LIMIT Deasserted when I_{OUT} is less than or equal to (IOUT_OC_FAULT_LIMIT-fault_iout_oc_fault_hyst) IOUT_OC_WARN: Asserted when I_{OUT} is greater than IOUT_OC_WARN_LIMIT Deasserted when I_{OUT} is less than or equal to (IOUT_OC_WARN_LIMIT-fault_iout_oc_fault_hyst) IOUT_OC_FAST_FAULT: Asserted when I_{OUT} is greater than MFR_IOUT_OC_FAST_FAULT_LIMIT Deasserted when I_{OUT} is less than or equal to (MFR_IOUT_OC_FAST_FAULT_LIMIT-fault_iout_oc_fault_hyst) LSB = 0.25 A, range = -8.0 to +7.75 A

Peripheral	Field name	Access	Address	Bits	Description
fault	fault_iout_uc_fault_hyst	RW	7000_3C08h (Loop 0) 7000_4008h (Loop 1)	[21:16]	Output undercurrent (IOUT_UC) fault deassertion hysteresis. Internally, the I_{OUT} UC fault is asserted when I_{OUT} exceeds IOUT_UC_FAULT_LIMIT for fault_iout_uc_fault_cnt samples without rising above (IOUT_UC_FAULT_LIMIT+fault_iout_uc_fault_hyst). The samples below IOUT_UC_FAULT_LIMIT need not be consecutive, but a single sample above (IOUT_UC_FAULT_LIMIT+fault_iout_uc_fault_hyst) will reset the count. For the typical case this parameter should be set to a positive current. This hysteresis parameter applies to only the I_{OUT} UC fault, as shown below. IOUT_UC_FAULT: Asserted when I_{OUT} is less than IOUT_UC_FAULT_LIMIT Deasserted when I_{OUT} is greater than or equal to (IOUT_UC_FAULT_LIMIT+fault_iout_uc_fault_hyst) LSB = 0.25 A, range = -8.0 to +7.75 A
fault	fault_temp_fault_hyst	RW	7000_3C08h (Loop 0) 7000_4008h (Loop 1)	[26:22]	Temperature fault (warning) deassertion hysteresis. Internally, the OT fault is asserted when the temperature exceeds the ADC code corresponding to OT_FAULT_LIMIT. The direction of the comparison depends on whether a negative temperature coefficient (NTC) or positive temperature coefficient (PTC) sensor is used. This means the sign of this parameter will also depend on the sensor chosen. Typically one should select a negative value for PTC and a

Peripheral	Field name	Access	Address	Bits	Description
					positive value for NTC. This hysteresis parameter applies to all temperature faults and warnings, as shown below. PTC OT_FAULT: Asserted when temp. is greater than fault_ot_fault_thresh Deasserted when temp. is less than (fault_ot_fault_thresh + fault_temp_fault_hyst) PTC OT_WARN: Asserted when temp. is greater than fault_ot_warn_thresh Deasserted when temp. is less than (fault_ot_warn_thresh + fault_temp_fault_hyst) PTC UT_WARN Asserted when temp. is less than fault_ut_warn_thresh Deasserted when temp. is greater than (fault_ut_warn_thresh - fault_temp_fault_hyst) PTC UT_FAULT: Asserted when temp. is less than fault_ut_fault_thresh Deasserted when temp. is greater than (fault_ut_fault_thresh - fault_temp_fault_hyst) NTC OT_FAULT: Asserted when temp. is less than fault_ot_fault_thresh Deasserted when temp. is greater than (fault_ot_fault_thresh + fault_temp_fault_hyst) NTC OT_WARN: Asserted when temp. is less than fault_ot_warn_thresh Deasserted when temp. is greater than (fault_ot_warn_thresh + fault_temp_fault_hyst) NTC UT_WARN: Asserted when temp. is greater than fault_ut_warn_thresh Deasserted when temp. is less

Peripheral	Field name	Access	Address	Bits	Description
					than (fault_ut_warn_thresh - fault_temp_fault_hyst) NTC UT_FAULT: Asserted when temp. is greater than fault_ut_fault_thresh Deasserted when temp. is less than (fault_ut_fault_thresh - fault_temp_fault_hyst) LSB = 1 TSADC code, range = -16 to +15 TSADC codes
fault	fault_shut_mask_loop	RW	7000_3C0Ch (Loop 0) 7000_400Ch (Loop 1)	[31:0]	Shutdown mask for Loop faults. Individual faults are enabled for shutdown when their corresponding bit is high. Enabled faults disable the loop output and assert the shutdown interrupt immediately upon fault assertion. The register is controlled by FW based on settings of the various PMBus fault responses. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A

Peripheral	Field name	Access	Address	Bits	Description
					only) 25 to 31: Unused
fault	fault_t2_shut_mask_loop	RW	7000_3C10h (Loop 0) 7000_4010h (Loop 1)	[31:0]	T2 shutdown mask for loop faults. Individual faults are enabled for T2 shutdown when their corresponding bit is high. Enabled faults disable the loop output and assert the shutdown interrupt aligned with the T2 ramp time, which typically aligns with the falling edge of the primary-side PWM pulse. Note that when corresponding bits in both fault_shut_mask_loop and fault_t2_shut_mask_loop are set to 1, the immediate shutdown behavior of fault_shut_mask_loop is implemented. This register should not be written directly but rather through PMBus command FW_CONFIG_FAULTS bits [199:168], which have a 1 to 1 mapping with bits [31:0] of this register. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE

Peripheral	Field name	Access	Address	Bits	Description
					21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
fault	fault_enable_loop	RW	7000_3C14h (Loop 0) 7000_4014h (Loop 1)	[31:0]	Fault enable for loop faults. Individual faults are enabled for fault interrupt generation when their corresponding bit is high. This register is controlled by FW based on the PMBus command FW_CONFIG_FAULTS bits [103:72], which have a 1 to 1 mapping with bits [31:0] of this register. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
fault	fault_vout_ov_fault_thresh	RW	7000_3C18h (Loop 0) 7000_401	[9:0]	Output overvoltage fault threshold. Note that this threshold is defined with respect to the voltage at the

Peripheral	Field name	Access	Address	Bits	Description
			8h (Loop 1)		VSEN pin, after the V_{OUT} sense resistor divider. **Computed by FW from PMBus command as follows:** fault_vout_ovp_fault_thresh = Int(VOUT_OV_FAULT_LIMIT(U16-X.X) * VOUT_SCALE_LOOP(U0.16) * 200 / (2^X * 2^16)) where X = negative of VOUT_MODE exponent LSB = 5 mV, range = 0.0 to 5.115 V
fault	fault_vout_ov_warn_thresh	RW	7000_3C1Ch (Loop 0) 7000_401Ch (Loop 1)	[9:0]	Output overvoltage warning threshold. Note that this threshold is defined with respect to the voltage at the VSEN pin, after the V_{OUT} sense resistor divider. **Computed by FW from PMBus command as follows:** fault_vout_ovp_warn_thresh = Int(VOUT_OV_WARN_LIMIT(U16-X.X) * VOUT_SCALE_LOOP(U0.16) * 200 / (2^X * 2^16)) where X = negative of VOUT_MODE exponent LSB = 5 mV, range = 0.0 to 5.115 V
fault	fault_vout_uv_fault_thresh	RW	7000_3C20h (Loop 0) 7000_4020h (Loop 1)	[9:0]	Output undervoltage fault threshold. Note that this threshold is defined with respect to the voltage at the VSEN pin, after the V_{OUT} sense resistor divider. **Computed by FW from PMBus command as follows:** fault_vout_uvp_fault_thresh = Int(VOUT_UV_FAULT_LIMIT(U16-X.X) * VOUT_SCALE_LOOP(U0.16) * 200 / (2^X * 2^16)) where X = negative of VOUT_MODE exponent LSB = 5 mV, range = 0.0 to 5.115 V

Peripheral	Field name	Access	Address	Bits	Description
fault	fault_vout_uv_warn_thresh	RW	7000_3C24h (Loop 0) 7000_4024h (Loop 1)	[9:0]	Output undervoltage warning threshold. Note that this threshold is defined with respect to the voltage at the VSEN pin, after the V_{OUT} sense resistor divider. **Computed by FW from PMBus command as follows:** fault_vout_uvp_warn_thresh = Int(VOUT_UV_WARN_LIMIT(U16-X.X) * VOUT_SCALE_LOOP(U0.16) * 200 / (2^X * 2^16)) where X = negative of VOUT_MODE exponent LSB = 5 mV, range = 0.0 to 5.115 V
fault	fault_vin_ov_fault_thresh	RW	7000_3C28h (Loop 0) 7000_4028h (Loop 1)	[9:0]	Input overvoltage fault threshold. **Computed by FW from PMBus command as follows:** fault_vin_ovp_fault_thresh = Int(VIN_OV_FAULT_LIMIT[10:0] * 2^(VIN_OV_FAULT_LIMIT[15:11] + 3)) LSB = 125 mV, range = 0.0 to 127.875 V
fault	fault_vin_ov_warn_thresh	RW	7000_3C2Ch (Loop 0) 7000_402Ch (Loop 1)	[9:0]	Input overvoltage warning threshold. **Computed by FW from PMBus command as follows:** fault_vin_ovp_warn_thresh = Int(VIN_OV_WARN_LIMIT[10:0] * 2^(VIN_OV_WARN_LIMIT[15:11] + 3)) LSB = 125 mV, range = 0.0 to 127.875 V
fault	fault_vin_uv_fault_thresh	RW	7000_3C30h (Loop 0) 7000_4030h (Loop 1)	[9:0]	Input undervoltage fault threshold. **Computed by FW from PMBus command as follows:** fault_vin_uvp_fault_thresh = Int(VIN_UV_FAULT_LIMIT[10:0] * 2^(VIN_UV_FAULT_LIMIT[15:11] + 3))

Peripheral	Field name	Access	Address	Bits	Description
					LSB = 125 mV, range = 0.0 to 127.875 V
fault	fault_vin_uv_warn_thresh	RW	7000_3C34h (Loop 0) 7000_4034h (Loop 1)	[9:0]	Input undervoltage warning threshold. **Computed by FW from PMBus command as follows:** fault_vin_uvp_warn_thresh = Int(VIN_UV_WARN_LIMIT[10:0] * 2^(VIN_UV_WARN_LIMIT[15:11] + 3)) LSB = 125 mV, range = 0.0 to 127.875 V
fault	fault_iout_oc_fault_thresh	RW	7000_3C38h (Loop 0) 7000_4038h (Loop 1)	[9:0]	Output overcurrent fault threshold **Computed by FW from PMBus command as follows:** fault_iout_oc_fault_thresh = Int(IOUT_OC_FAULT_LIMIT[10:0] * 2^(IOUT_OC_FAULT_LIMIT[15:11])) LSB = 1 A, range = 0 to 255 A
fault	fault_iout_oc_lv_fault_thresh	RW	7000_3C3Ch (Loop 0) 7000_403Ch (Loop 1)	[9:0]	Output overcurrent LV fault threshold. Note that this threshold is defined with respect to the voltage at the VSEN pin, after the V_{OUT} sense resistor divider. **Computed by FW from PMBus command as follows:** fault_iout_oc_lv_fault_thresh = Int(IOUT_OC_LV_FAULT_LIMIT(U16-X.X) * VOUT_SCALE_LOOP(U0.16) * 200 / (2^X * 2^16)) where X = negative of VOUT_MODE exponent LSB = 5 mV, range = 0.0 to 5.115 V
fault	fault_iout_oc_warn_thresh	RW	7000_3C40h (Loop 0) 7000_4040h (Loop 1)	[7:0]	Output overcurrent warn threshold. **Computed by FW from PMBus command as follows:** fault_iout_oc_warn_thresh = Int(IOUT_OC_WARN_LIMIT[10:

Peripheral	Field name	Access	Address	Bits	Description
					0] * 2^IOUT_OC_WARN_LIMIT[15:11]) LSB = 1 A, range = 0 to 255 A
fault	fault_iout_uc_fault_thresh	RW	7000_3C44h (Loop 0) 7000_4044h (Loop 1)	[7:0]	Output undercurrent fault threshold. Fault asserted when the cycle-averaged output current is below this threshold. **Computed by FW from PMBus command as follows:** fault_iout_uc_fault_thresh = Int(IOUT_UC_FAULT_LIMIT[10:0] * 2^(IOUT_UC_FAULT_LIMIT[15:11])) LSB = 1 A, range = -128 to +127 A
fault	fault_mfr_iout_oc_fast_thresh	RW	7000_3C48h (Loop 0) 7000_4048h (Loop 1)	[7:0]	Fast output overcurrent fault threshold. **Computed by FW from PMBus command as follows:** fault_mfr_iout_oc_fast_thresh = Int(MFR_IOUT_OC_FAST_FAULT_LIMIT[10:0] * 2^MFR_IOUT_OC_FAST_FAULT_LIMIT[15:11]) LSB = 1 A, range = 0 to 255 A
fault	fault_iin_oc_fault_thresh	RW	7000_3C4Ch (Loop 0) 7000_404Ch (Loop 1)	[7:0]	Input overcurrent fault threshold. **Computed by FW from PMBus command as follows:** fault_iin_ocp_fault_thresh = Int(IIN_OC_FAULT_LIMIT[10:0] * 2^(IIN_OC_FAULT_LIMIT[15:11] + 2)) LSB = 0.25 A, range = 0.0 to 63.75 A
fault	fault_iin_oc_warn_thresh	RW	7000_3C50h (Loop 0) 7000_4050h (Loop 1)	[7:0]	Input overcurrent warning threshold. **Computed by FW from PMBus command as follows:** fault_iin_ocp_warn_thresh = Int(IIN_OC_WARN_LIMIT[10:0] * 2^(IIN_OC_WARN_LIMIT[15:11]

Peripheral	Field name	Access	Address	Bits	Description
					+ 2)) LSB = 0.25 A, range = 0.0 to 63.75 A
fault	fault_ot_fault_thresh	RW	7000_3C54h (Loop 0) 7000_4054h (Loop 1)	[9:0]	Overtemperature fault threshold in TSADC codes. This threshold is computed by FW based on the OT_FAULT_LIMIT PMBus command and the FW-implemented ADC code to temperature LUT. LSB = 1 TSADC code, range = 0.0 to 1023 AD codes
fault	fault_ot_warn_thresh	RW	7000_3C58h (Loop 0) 7000_4058h (Loop 1)	[9:0]	Overtemperature warn threshold in TSADC codes. This threshold is computed by FW based on the OT_WARN_LIMIT PMBus command and the FW-implemented ADC code to temperature LUT. LSB = 1 TSADC code, range = 0.0 to 1023 AD codes
fault	fault_ut_fault_thresh	RW	7000_3C5Ch (Loop 0) 7000_405Ch (Loop 1)	[9:0]	Undertemperature fault threshold in TSADC codes. This threshold is computed by FW based on the UT_FAULT_LIMIT PMBus command and the FW-implemented ADC code to temperature LUT. LSB = 1 TSADC code, range = 0.0 to 1023 AD codes
fault	fault_ut_warn_thresh	RW	7000_3C60h (Loop 0) 7000_4060h (Loop 1)	[9:0]	Undertemperature warn threshold in TSADC codes. This threshold is computed by FW based on the UT_WARN_LIMIT PMBus command and the FW-implemented ADC code to temperature LUT. LSB = 1 TSADC code, range = 0.0 to 1023 AD codes
fault	fault_temp_src_sel	RW	7000_3C64h (Loop 0) 7000_4064h (Loop 1)	[2:0]	Fault temperature source and sensor TC select. 0: TSEN, NTC 1: BTSEN, NTC 2: XDPP1100 internal temperature

Peripheral	Field name	Access	Address	Bits	Description
					3: TSEN, PTC 4: BTSEN, PTC
fault	fault_shut_clr_loop	W	7000_3C68h (Loop 0) 7000_4068h (Loop 1)	[0]	Clears shutdown faults in fault_reg_loop when set high. This field should not be written until after FW has completed shutdown-related cleanup (e.g., $V_{control}$ target returned to 0, appropriate blocks disabled).
fault	fault_polarity_loop	RW	7000_3C6Ch (Loop 0) 7000_406Ch (Loop 1)	[31:0]	Defines the polarity of the fault with respect to the threshold. When set to 0, the fault/warning is asserted using its default polarity (e.g., over-type fault asserted when above the threshold). When set to 1, the fault/warning is asserted with the inverse to its default polarity (e.g., over-type fault asserted when below the threshold). The inverse polarity may be used to generate an interrupt when a parameter no longer exceeds its fault/warning threshold. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN

Peripheral	Field name	Access	Address	Bits	Description
					20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
fault	fault_force_loop	RW	7000_3C70h (Loop 0) 7000_4070h (Loop 1)	[31:0]	Fault force set register. When the bit corresponding to a fault/warning is set to 1, that fault/warning is asserted as if it had exceeded its threshold. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
fault	fault_clear_loop	W	7000_3C74h (Loop 0) 7000_4074h (Loop 1)	[31:0]	Fault force clear register. When the bit corresponding to a fault/warning is set to 1, that fault/warning is cleared in the fault_status_loop and fault_ref_loop registers. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN

Peripheral	Field name	Access	Address	Bits	Description
					3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
fault	fault_status_loop	R	7000_3C78h (Loop 0) 7000_4078h (Loop 1)	[31:0]	Fault status register generated by sub-sampling fault_reg_loop at 2 MHz. Fault interrupts are generated from this register. Once a fault bit is set it is latched in this register and can only be cleared via fault_clear_loop. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN

Peripheral	Field name	Access	Address	Bits	Description
					16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
fault	fault_reg_loop	R	7000_3C7Ch (Loop 0) 7000_407Ch (Loop 1)	[31:0]	Fault status register updated at 50 MHz. Once a fault bit is set it is latched in this register and can only be cleared via fault_clear_loop. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
fault	fault_iout_cc_mode	R	7000_3C80h (Loop 0) 7000_408	[0]	Status flag indicating I_{OUT} constant current mode operation. 0: Normal operation 1: Constant current mode

Peripheral	Field name	Access	Address	Bits	Description
			0h (Loop 1)		
fault	fault_iout_cc_adj	R	7000_3C80h (Loop 0) 7000_4080h (Loop 1)	[14:1]	Status register indicating magnitude of constant current mode output voltage adjustment referenced to the VSEN pin (i.e., after the V_{OUT} sense resistor divider). LSB = 156.25 µV, range = 0.0 to 2.55984375 V
fault	fault_iout_cc_en	RW	7000_3C84h (Loop 0) 7000_4084h (Loop 1)	[0]	Constant current mode enable for IOUT_OC_FAULT. Set by FW based on IOUT_OC_FAULT_RESPONSE. 0: Disabled 1: Enabled
fault	fault_iout_cc_en_fast	RW	7000_3C84h (Loop 0) 7000_4084h (Loop 1)	[1]	Constant current mode enable for MFR_IOUT_OC_FAST_FAULT. Set by FW based on MFR_IOUT_OC_FAST_FAULT_RESPONSE. 0: Disabled 1: Enabled
faultcom	fault_shut_mask_com	RW	7000_5400h	[31:0]	Shutdown mask for “common” faults. Individual faults are enabled for shutdown when their corresponding bit is high. Enabled faults disable both loop outputs and assert the shutdown interrupt. 0: Unused 1: Unused 2: IS1 (ISEN) tracking fault 3: IS2 (BISEN) tracking fault 4: Fbal1 fault 5: IS1 (ISEN) PCL fault 6: IS1 (ISEN) SCP fault 7: Fbal2 fault 8: IS2 (BISEN) PCL fault 9: IS2 (BISEN) SCP fault 10: Unused 11: VREF open fault 12: VSEN open fault 13: Unused

Peripheral	Field name	Access	Address	Bits	Description
					14: VRREF open fault 15: VRSEN open fault 16: Unused 17: BVREF_BVRREF open fault 18: BVSEN_BVRSEN open fault 19 to 31: Unused
faultcom	fault_block_on_shut	RW	7000_5404h	[0]	Defines whether faults continued to be reported after an initial shutdown fault had occurred and triggered shutdown. 0: Fault reporting continues after shutdown fault. 1: Fault reporting blocked on new faults after initial shutdown fault occurs. Fault will continue to be blocked until the shutdown fault is cleared via fault_shut_clr_loop, fault_shut_clr_com or fault_shut_clr_all.
faultcom	fault_enable_com	RW	7000_5408h	[31:0]	Fault enable for common faults. Individual faults are enabled for fault interrupt generation when their corresponding bit is high. This register is controlled by FW based on the PMBus command FW_CONFIG_FAULTS bits [167:136] which have a 1 to 1 mapping with bits [31:0] of this register. 0: Unused 1: Unused 2: IS1 (ISEN) tracking fault 3: IS2 (BISEN) tracking fault 4: Fbal1 fault 5: IS1 (ISEN) PCL fault 6: IS1 (ISEN) SCP fault 7: Fbal2 fault 8: IS2 (BISEN) PCL fault 9: IS2 (BISEN) SCP fault 10: Unused 11: VREF open fault 12: VSEN open fault 13: Unused 14: VRREF open fault

Peripheral	Field name	Access	Address	Bits	Description
					15: VRSEN open fault 16: Unused 17: BVREF_BVRREF open fault 18: BVSEN_BVRSEN open fault 19 to 31: Unused
faultcom	fault_shut_clr_all	W	7000_540Ch	[0]	Clears shutdown faults in fault_reg_com and both fault_reg_loop registers when high. This field should not be written until after FW has completed shutdown-related cleanup (e.g., $V_{control}$ target returned to 0, appropriate blocks disabled).
faultcom	fault_shut_clr_com	W	7000_5410h	[0]	Clears shutdown faults in fault_reg_com when set high. This field should not be written until after FW has completed shutdown-related cleanup (e.g., $V_{control}$ target returned to 0, appropriate blocks disabled).
faultcom	fault_shut_bus	R	7000_5414h	[2:0]	Shutdown IRQ status. [0]: Loop 0 fault [1]: Loop 1 fault [2]: Common fault 0: IRQ asserted, 1: IRQ deasserted
faultcom	fault_irq_bus	R	7000_5414h	[5:3]	Fault IRQ status. [0]: Loop 0 fault [1]: Loop 1 fault [2]: Common fault 0: IRQ asserted, 1: IRQ deasserted
faultcom	fault_force_com	RW	7000_5418h	[31:0]	Fault force set register. When the bit corresponding to a fault or warning is set to 1, that fault or warning is asserted as if it had exceeded its threshold. 0: Unused 1: Unused 2: IS1 (ISEN) tracking fault 3: IS2 (BISEN) tracking fault 4: Fbal1 fault 5: IS1 (ISEN) PCL fault 6: IS1 (ISEN) SCP fault

Peripheral	Field name	Access	Address	Bits	Description
					7: Fbal2 fault 8: IS2 (BISEN) PCL fault 9: IS2 (BISEN) SCP fault 10: Unused 11: VREF open fault 12: VSEN open fault 13: Unused 14: VRREF open fault 15: VRSEN open fault 16: Unused 17: BVREF_BVRREF open fault 18: BVSEN_BVRSEN open fault 19 to 31: Unused
faultcom	fault_clear_com	W	7000_541Ch	[31:0]	Fault force clear register. When the bit corresponding to a fault or warning is set to 1, that fault or warning is cleared in the fault_status_loop and fault_ref_loop registers. 0: Unused 1: Unused 2: IS1 (ISEN) tracking fault 3: IS2 (BISEN) tracking fault 4: Fbal1 fault 5: IS1 (ISEN) PCL fault 6: IS1 (ISEN) SCP fault 7: Fbal2 fault 8: IS2 (BISEN) PCL fault 9: IS2 (BISEN) SCP fault 10: Unused 11: VREF open fault 12: VSEN open fault 13: Unused 14: VRREF open fault 15: VRSEN open fault 16: Unused 17: BVREF_BVRREF open fault 18: BVSEN_BVRSEN open fault 19 to 31: Unused
faultcom	fault_status_com	R	7000_5420h	[31:0]	Fault status register generated by sub-sampling fault_reg_com at 2 MHz. Fault interrupts are generated from this register. Once a fault bit is set, it is latched in this register and can only be cleared via fault_clear_com. 0: Unused

Peripheral	Field name	Access	Address	Bits	Description
					1: Unused 2: IS1 (ISEN) tracking fault 3: IS2 (BISEN) tracking fault 4: Fbal1 fault 5: IS1 (ISEN) PCL fault 6: IS1 (ISEN) SCP fault 7: Fbal2 fault 8: IS2 (BISEN) PCL fault 9: IS2 (BISEN) SCP fault 10: Unused 11: VREF open fault 12: VSEN open fault 13: Unused 14: VRREF open fault 15: VRSEN open fault 16: Unused 17: BVREF_BVRREF open fault 18: BVSEN_BVRSEN open fault 19 to 31: Unused
faultcom	fault_encode	R	7000_5424h	[7:0]	Priority encoding of the word {fault_status_com[31:0],fault1_status_loop[31:0],fault0_status_loop[31:0]} with fault_encode indicating the position of the first “1” searching from the LSB. Fault0_status_loop[31:0] reports in the range 0 to 31, fault1_status_loop[31:0] reports in the range 32 to 63 and fault_status_com[31:0] reports in the range 224 to 255. **Note:** Both the no-fault case and fault0_status_loop[0]=1 will return fault_encode=0. fault_irq_bus can be used along with fault_encode to distinguish between the two cases
faultcom	fault_reg_com	R	7000_5428h	[31:0]	Fault status register updated at 50 MHz. Once a fault bit is set it is latched in this register and can only be cleared via fault_clear_com. 0: Unused 1: Unused 2: IS1 (ISEN) tracking fault 3: IS2 (BISEN) tracking fault

Peripheral	Field name	Access	Address	Bits	Description
					4: Fbal1 fault 5: IS1 (ISEN) PCL fault 6: IS1 (ISEN) SCP fault 7: Fbal2 fault 8: IS2 (BISEN) PCL fault 9: IS2 (BISEN) SCP fault 10: Unused 11: VREF open fault 12: VSEN open fault 13: Unused 14: VRREF open fault 15: VRSEN open fault 16: Unused 17: BVREF_BVRREF open fault 18: BVSEN_BVRSEN open fault 19 to 31: Unused

9.8 Fault PMBus commands

Table 67 Fault-related PMBus command descriptions

Command name	Access	Length	Address	Bits	Description
VOUT_SCALE_LOOP	RW	Word	29h	[15:0]	Scales VOUT_COMMAND and other V_{OUT}-related commands for the external resistor divider between V_{OUT} and VSEN. The format is DIRECT, U0.16 and is computed as: VOUT_SCALE_LOOP = INT(2^16 * VSEN / V_{OUT})
VOUT_OV_FAULT_LIMIT	RW	Word	40h	[15:0]	Sets the value of the output voltage that causes an output overvoltage fault. The data bytes are formatted according to the setting of the VOUT_MODE command.
VOUT_OV_FAULT_RESPONSE	RW	Byte	41h	[7:0]	Instructs the device on what action to take in response to an output overvoltage fault. **[7:6]: Response** 0: Continue operation without interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down and respond

Command name	Access	Length	Address	Bits	Description
					according to retry bits [5:3] 3: Shut down until fault no longer present **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
VOUT_OV_WARN_LIMIT	RW	Word	42h	[15:0]	Sets the value of the output voltage that causes an output overvoltage warning. The data bytes are formatted according to the setting of the VOUT_MODE command.
VOUT_UV_WARN_LIMIT	RW	Word	43h	[15:0]	Sets the value of the output voltage that causes an output undervoltage warning. The data bytes are formatted according to the setting of the VOUT_MODE command.
VOUT_UV_FAULT_LIMIT	RW	Word	44h	[15:0]	Sets the value of the output voltage that causes an output undervoltage fault. The data bytes are formatted according to the setting of the VOUT_MODE command.
VOUT_UV_FAULT_RESPONSE	RW	Byte	45h	[7:0]	Instructs the device on what action to take in response to an output undervoltage fault. **[7:6]: Response** 0: Continue operation without interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down and respond

Command name	Access	Length	Address	Bits	Description
					according to retry bits [5:3] 3: Shut down until fault no longer present **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
IOUT_OC_FAULT_LIMIT	RW	Word	46h	[15:0]	Sets the value of the output current, in amps, that causes an overcurrent fault condition. The format is LINEAR11. The hardware LSB is 1 A, suggesting an exponent of 0.
IOUT_OC_FAULT_RESPONSE	RW	Byte	47h	[7:0]	Instructs the device on what action to take in response to an output overcurrent fault. **[7:6]: Response** 0: Continue operation maintaining I_{OUT} at IOUT_OC_FAULT_LIMIT without regard to output voltage 1: Continue operation maintaining I_{OUT} at IOUT_OC_FAULT_LIMIT as long as the voltage remains above IOUT_OC_LV_FAULT_LIMIT 2: Continue operation maintaining I_{OUT} at IOUT_OC_FAULT_LIMIT as long as the voltage remains above IOUT_OC_LV_FAULT_LIMIT for the time specified by delay bits [2:0] then if still operating in current limiting, shut down and respond according to retry bits [5:3] 3: Shut down and respond according to retry bits [5:3]

Command name	Access	Length	Address	Bits	Description
					[5:3]: Retry setting 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
IOUT_OC_LV_FAULT_LIMIT	RW	Word	48h	[15:0]	In the case where the response to an overcurrent condition is to operate in a constant current mode unless the output voltage is pulled below the specified value, the IOUT_OC_LV_FAULT_LIMIT specifies that voltage threshold. The data bytes are formatted according to the setting of the VOUT_MODE command. Note that the IOUT_OC_LV_FAULT_RESPONSE command is not supported. The only supported response is shutdown.
IOUT_OC_WARN_LIMIT	RW	Word	4Ah	[15:0]	Sets the value of the output current, in amps, that causes an overcurrent warning condition. The format is LINEAR11. The hardware LSB is 1 A, suggesting an exponent of 0.
IOUT_UC_FAULT_LIMIT	RW	Word	4Bh	[15:0]	Sets the value of the output current, in amps, that causes an undercurrent fault condition. The format is LINEAR11. The hardware LSB is 1 A, suggesting an exponent of 0.
IOUT_UC_FAULT_RESPONSE	RW	Byte	4Ch	[7:0]	Instructs the device on what action to take in response to an output undercurrent fault. **[7:6]: Response** 0: Continue operation without

Command name	Access	Length	Address	Bits	Description
					interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down SR FETs and respond according to retry bits [5:3] 3: Shut down SR until fault condition removed **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
OT_FAULT_RESPONSE	RW	Byte	50h	[7:0]	Instructs the device on what action to take in response to an overtemperature fault. **[7:6]: Response** 0: Continue operation without interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down and respond according to retry bits [5:3] 3: Shut down until fault no longer present **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded

Command name	Access	Length	Address	Bits	Description
					OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
OT_WARN_LIMIT	RW	Word	51h	[15:0]	Sets the temperature of the unit, in degrees Celsius, at which it should indicate an overtemperature warning. The format is LINEAR11.
UT_WARN_LIMIT	RW	Word	52h	[15:0]	Sets the temperature of the unit, in degrees Celsius, at which it should indicate an undertemperature warning. The format is LINEAR11.
UT_FAULT_LIMIT	RW	Word	53h	[15:0]	Sets the temperature of the unit, in degrees Celsius, at which it should indicate an undertemperature fault. The format is LINEAR11.
UT_FAULT_RESPONSE	RW	Byte	54h	[7:0]	Instructs the device on what action to take in response to an overtemperature fault. **[7:6]: Response** 0: Continue operation without interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down and respond according to retry bits [5:3] 3: Shut down until fault no longer present **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time

Command name	Access	Length	Address	Bits	Description
					units between restart attempts, unit defined by FW_CONFIG_FAULTS command
VIN_OV_FAULT_LIMIT	RW	Word	55h	[15:0]	Sets the value of the input voltage that causes an input overvoltage fault. The format is LINEAR11. The hardware LSB is 0.125 V, suggesting an exponent of -3.
VIN_OV_FAULT_RESPONSE	RW	Byte	56h	[7:0]	Instructs the device on what action to take in response to an input overvoltage fault. **[7:6]: Response** 0: Continue operation without interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down and respond according to retry bits [5:3] 3: Shut down until fault no longer present **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
VIN_OV_WARN_LIMIT	RW	Word	57h	[15:0]	Sets the value of the input voltage that causes an input overvoltage warning. The format is LINEAR11. The hardware LSB is 0.125 V, suggesting an exponent of -3.
VIN_UV_WARN_LIMIT	RW	Word	58h	[15:0]	Sets the value of the input voltage that causes an input undervoltage warning. The

Command name	Access	Length	Address	Bits	Description
					format is LINEAR11. The hardware LSB is 0.125 V, suggesting an exponent of -3.
VIN_UV_FAULT_LIMIT	RW	Word	59h	[15:0]	Sets the value of the input voltage that causes an input undervoltage fault. The format is LINEAR11. The hardware LSB is 0.125 V, suggesting an exponent of -3.
VIN_UV_FAULT_RESPONSE	RW	Byte	5Ah	[7:0]	Instructs the device on what action to take in response to an input undervoltage fault. **[7:6]: Response** 0: Continue operation without interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down and respond according to retry bits [5:3] 3: Shut down until fault no longer present **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
IIN_OC_FAULT_LIMIT	RW	Word	5Bh	[15:0]	Sets the value of the input current, in amps, that causes an input overcurrent fault. The format is LINEAR11. The hardware is 0.25 A, suggesting an exponent of -2
IIN_OC_FAULT_RESPONSE	RW	Byte	5Ch	[7:0]	Instructs the device on what action to take in response to an input overcurrent fault.

Command name	Access	Length	Address	Bits	Description
					[7:6]: Response 0: Continue operation without interruption 1: Continue operation for time specified by delay bits [2:0] then if fault condition still present, shut down and respond according to retry bits [5:3] 2: Shut down and respond according to retry bits [5:3] 3: Shut down until fault no longer present **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
IIN_OC_WARN_LIMIT	RW	Word	5Dh	[15:0]	Sets the value of the input current, in amps, that causes an input overcurrent warning. The format is LINEAR11. The hardware is 0.25 A, suggesting an exponent of -2
TON_MAX_FAULT_LIMIT	RW	Word	62h	[15:0]	Sets an upper limit, in ms, on how long the unit can attempt to power up the output without reaching the output undervoltage fault limit. A value of 0 ms means that there is no limit and that the unit can attempt to bring up the output voltage indefinitely. The format is LINEAR11
TON_MAX_FAULT_RESPONSE	RW	Byte	63h	[7:0]	Instructs the device on what action to take in response to a TON_MAX fault. **[7:6]: Response** 0: Continue operation without interruption

Command name	Access	Length	Address	Bits	Description
					1: Not supported 2: Shut down and respond according to retry bits [5:3] 3: Not supported **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
TOFF_MAX_WARN_LIMIT	RW	Word	66h	[15:0]	Sets an upper limit, in ms, on how long the unit can attempt to power down the output without reaching 12.5 percent of the output voltage programmed at the time the unit is turned off. A value of 0 ms means that there is no limit and that the unit waits indefinitely for the output voltage to decay
POUT_OP_WARN_LIMIT	RW	Word	6Ah	[15:0]	Sets the value of the output power, in watts, that causes an output overpower warning. The format is LINEAR11
PIN_OP_WARN_LIMIT	RW	Word	6Bh	[15:0]	Sets the value of the output power, in watts, that causes an input overpower warning. The format is LINEAR11
STATUS_BYTE	RW	Byte	78h	[7:0]	Returns one byte of information with a summary of the most critical faults. [7] BUSY: A fault was declared because the device was busy and unable to respond [6] OFF: Asserted if the unit is not providing power to the output, regardless of the reason, including simply not being enabled

Command name	Access	Length	Address	Bits	Description
					[5] VOUT_OV_FAULT: An output overvoltage fault has occurred [4] IOUT_OC_FAULT: An output overcurrent fault has occurred [3] VIN_UV_FAULT: An input undervoltage fault has occurred [2] TEMPERATURE: A temperature fault or warning has occurred [1] CML: A communications, memory or logic fault has occurred [0] NONE_OF_THE_ABOVE: A fault or warning not listed in bits [7:1] has occurred
STATUS_WORD	RW	Word	79h	[15:0]	The STATUS_WORD command returns two bytes of information with a summary of the unit's fault condition. Based on the information in these bytes, the host can get more information by reading the appropriate status registers. The low byte of the STATUS_WORD is the same register as the STATUS_BYTE command. [15] V_{OUT}: An output voltage fault or warning has occurred. [14] I_{OUT}/P_{OUT}: An output current or output power fault or warning has occurred [13] INPUT: An input voltage, input current or input power fault of warning has occurred [12] MFRSPECIFIC: A manufacturer-specific fault or warning has occurred [11] PG_STATUS: The POWER_GOOD signal, if present, is negated [10] FANS: A fan fault or warning has occurred [9] OTHER: A bit in STATUS_OTHER is set [8] UNKNOWN: A fault or warning not listed in bits [15:9] of STATUS_WORD has occurred [7:0]: See STATUS_BYTE

Command name	Access	Length	Address	Bits	Description
STATUS_VOUT	RW	Byte	7Ah	[7:0]	Returns one data byte with contents as follows: [7]: VOUT_OV_FAULT [6]: VOUT_OV_WARNING [5]: VOUT_UV_WARNING [4]: VOUT_UV_FAULT [3]: VOUT_MAX_MIN warning [2]: TON_MAX_FAULT [1]: TOFF_MAX_WARNING [0]: Reserved
STATUS_IOUT	RW	Byte	7Bh	[7:0]	Returns one data byte with contents as follows: [7]: IOUT_OC_FAULT [6]: IOUT_OC_LV_FAULT [5]: IOUT_OC_WARNING [4]: IOUT_UC_FAULT [3]: Current sharing fault [2]: In power limiting mode [1]: Reserved [0]: POUT_OP_WARNING
STATUS_INPUT	RW	Byte	7Ch	[7:0]	Returns one data byte with contents as follows: [7]: VIN_OV_FAULT [6]: VIN_OV_WARNING [5]: VIN_UV_WARNING [4]: VIN_UV_FAULT [3]: Unit off for insufficient voltage [2]: IIN_OC_FAULT [1]: IIN_OC_WARNING [0]: PIN_OP_WARNING
STATUS_TEMPERATURE	RW	Byte	7Dh	[7:0]	Returns one data byte with contents as follows: [7]: OT_FAULT [6]: OT_WARNING [5]: UT_WARNING [4]: UT_FAULT [3]: Reserved [2]: Reserved [1]: Reserved [0]: Reserved
STATUS_CML	RW	Byte	7Eh	[7:0]	Returns one data byte with contents as follows: [7]: Invalid or unsupported command received [6]: Invalid or unsupported data received

Command name	Access	Length	Address	Bits	Description
					[5]: Packet error check (PEC) failed [4]: Memory fault detected [3]: Processor fault detected [2]: Reserved [1]: A communication fault not listed has occurred [0]: Other memory or logic fault has occurred
STATUS_OTHER	RW	Byte	7Fh	[7:0]	Returns one data byte with contents as follows: [7]: Reserved [6]: Reserved [5]: Reserved [4]: Reserved [3]: Reserved [2]: Reserved [1]: Reserved [0]: First to assert SMBALERT#
STATUS_MFR_SPECIFIC	RW	Byte	80h	[7:0]	Returns one data byte with contents as follows: [7]: Sync fault [6]: Reserved [5]: IOUT_OC_FAST_FAULT [4]: Common fault [3]: External fault [2]: VDD_UVLO fault (XDPP1100A only) [1]: Reserved [0]: Reserved
FW_CONFIG_FAULTS	RW	Block 25 Bytes	C8h	[1:0]	**Temperature_Delay_Unit**: Time unit for retry responses. 0: 1 ms 1: 4 ms 2: 16 ms 3: 256 ms
				[3:2]	**Iout_Delay_Unit**: Time unit for retry responses. 0: 1 ms 1: 4 ms 2: 16 ms 3: 256 ms
				[5:4]	**Vin_Delay_Unit**: Time unit for retry responses. 0: 1 ms 1: 4 ms

Command name	Access	Length	Address	Bits	Description
					2: 16 ms 3: 256 ms
				[7:6]	**Vout_Delay_Unit**: Time unit for retry responses. 0: 1 ms 1: 4 ms 2: 16 ms 3: 256 ms
				[39:8]	**Fault_pin_mask_hw[31:0]**: FAULT pin masking for hardware faults. Set bit to 1 to disable fault reporting on FAULT pin. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
				[71:40]	**Fault_pin_mask_fw[31:0]**: FAULT pin masking for FW faults. Set bit to 1 to disable fault reporting on FAULT pin. 0: Reserved 1: Common fault 2: TON_MAX_FAULT 3: TOFF_MAX_WARN 4: PIN_OP_WARN 5: POUT_OP_WARN

Command name	Access	Length	Address	Bits	Description
					6: VIN_INSUFFICIENT 7 to 31: Unused
				[103:72]	**Fault_enable_mask_loop_hw[31:0]**: Masking for loop hardware faults. Set a bit to 1 to disable PMBus reporting on the corresponding fault. 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT 13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
				[135:104]	**Fault_enable_mask_loop_fw[31:0]**: Masking for loop FW faults. Set a bit to 1 to disable PMBus reporting on the corresponding fault. 0: Reserved 1: Common fault 2: TON_MAX_FAULT 3: TOFF_MAX_WARN 4: PIN_OP_WARN 5: POUT_OP_WARN 6: VIN_INSUFFICIENT 7 to 31: Unused
				[167:136]	**Fault_enable_mask_loop_common[31:0]**: Masking for common faults. Set a bit to 1 to disable

Command name	Access	Length	Address	Bits	Description
					PMBus reporting on the corresponding fault. 0: Unused 1: Unused 2: IS1 (ISEN) tracking fault 3: IS2 (BISEN) tracking fault 4: Fbal1 fault 5: IS1 (ISEN) PCL fault 6: IS1 (ISEN) SCP fault 7: Fbal2 fault 8: IS2 (BISEN) PCL fault 9: IS2 (BISEN) SCP fault 10: unused 11: VREF open fault 12: VSEN open fault 13: Unused 14: VRREF open fault 15: VRSEN open fault 16: Unused 17: BVREF_BVRREF open fault 18: BVSEN_BVRSEN open fault 19 to 31: Unused
				[199:168]	**Fault_t2_shut_mask_loop_hw[31:0]**: Masking for loop HW fault shutdown on pwm ramp t2 time (usually falling edge of PWM pulse). Bits [199:168] correspond to bits [31:0] of HW parameter fault_t2_shut_mask_loop. Set corresponding bit high to enable fault shutdown on t2. Note that the fault must still be enabled for shutdown via the corresponding PMBus “RESPONSE” command 0: Reserved 1: VOUT_OV_FAULT 2: VOUT_OV_WARN 3: VOUT_UV_FAULT 4: VOUT_UV_WARN 5: VIN_OV_FAULT 6: VIN_OV_WARN 7: VIN_UV_FAULT 8: VIN_UV_WARN 9: IOUT_OC_FAULT 10: IOUT_OC_LV_FAULT 11: IOUT_OC_WARN 12: IOUT_UC_FAULT

Command name	Access	Length	Address	Bits	Description
					13: MFR_IOUT_OC_FAST 14: IIN_OC_FAULT 15: IIN_OC_WARN 16: OT_FAULT 17: OT_WARN 18: UT_FAULT 19: UT_WARN 20: POWER_LIMIT_MODE 21: ISHARE_FAULT 22: VOUT_MAX_MIN_WARN 23: SYNC_FAULT 24: VDD_UVLO (XDPP1100A only) 25 to 31: Unused
MFR_IOUT_OC_FAST_FAULT_RESPONSE	RW	Byte	CAh	[7:0]	Instructs the device on what action to take in response to a fast output overcurrent fault **[7:6]: Response** 0: Continue operation maintaining I_{OUT} at IOUT_OC_FAULT_LIMIT without regard to output voltage 1: Continue operation maintaining I_{OUT} at IOUT_OC_FAULT_LIMIT as long as the voltage remains above IOUT_OC_LV_FAULT_LIMIT 2: Not supported 3: Shut down and respond according to retry bits [5:3] **[5:3]: Retry setting** 0: Remain disabled until fault cleared 1 to 6: Attempt to restart [5:3] times with delay [2:0] between restarts 7: Attempt to restart continuously until commanded OFF **[2:0]: Delay time** 0 to 7: Number of delay time units between restart attempts, unit defined by FW_CONFIG_FAULTS command
MFR_IOUT_OC_FAST_FAULT_LIMIT	RW	Word	D1h	[15:0]	Sets the value of the output current, in amps, that causes a fast overcurrent fault condition. The “fast” overcurrent uses an unfiltered version of the output

Command name	Access	Length	Address	Bits	Description
					current while the standard overcurrent fault uses a filtered version of the output current. The format is LINEAR11. The hardware LSB is 1 A, suggesting an exponent of 0

10 Current sharing (I_{SHARE})

This chapter discusses the current sharing feature in more detail as well as the relevant register settings related to current sensing.

To increase power delivery to the load, multiple power supplies may be connected in parallel to drive a common output voltage. However, equal current sharing between the supplies is not automatically guaranteed. For example, if one supply has a slightly higher V_{OUT} it tends to deliver more power to the load than a lower-voltage supply. In extreme cases, the higher-voltage supply may even source current into the lower-voltage supply leading to reduced system efficiency. Other detrimental effects of unequal current sharing include reduced total power delivery and early triggering of overcurrent or overtemperature protection.

Current sharing can be performed either actively or passively, and the XDPP1100 supports both methods to balance currents between supplies. Passive current sharing uses the droop resistance load-line as described in Section 5.4. No additional external components or connections are required, and each supply is programmed with a non-zero droop resistance. A supply with a higher no-load output voltage tends to drive a higher portion of the output current, leading to a higher V_{OUT} droop than a lower output voltage supply. The current in the parallel units reaches a balance when the output voltage in each unit droops to the same voltage.

Active current sharing requires a connection between the supplies to share average current implementation. XDPP1100 implementation also requires one external resistor per supply to compute the average system current. The remaining subsections in this chapter will discuss the XDPP1100 active current sharing implementation.

10.1 Current sharing circuit

For active current sharing, the XDPP1100 IMON pin is used to share the current information between power supplies. An example system with two XDPP1100-based power supplies connected in parallel is shown in Figure 85.

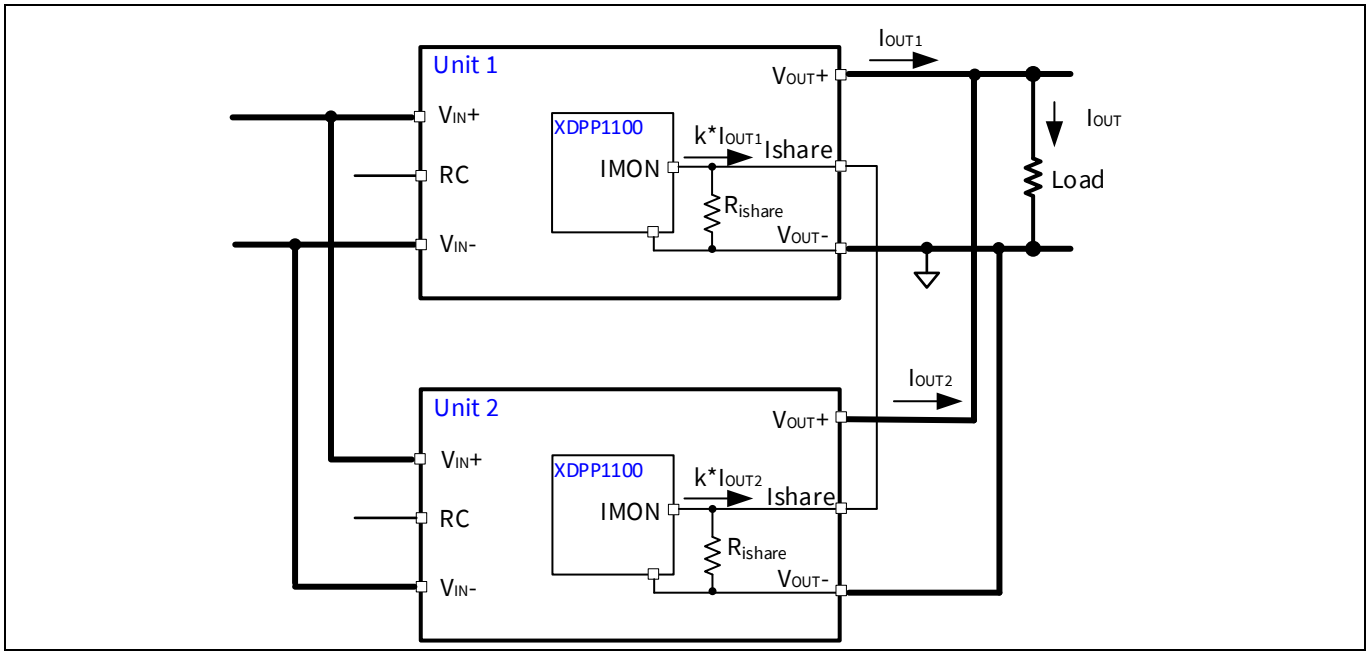

Figure 85 Active current sharing example with two units in parallel with simultaneous module start-up

The IMON pin is connected to:

- Current DAC output; in the current sharing mode it outputs a current proportional to the load current share of the individual supply
- TSADC input; in the current sharing mode it allows measurement of the voltage across the resistor

Each supply also requires a resistor, R_{ishare} = 1875 Ω, connected between IMON and ground. With N supplies in parallel, the voltage drop across the resistors can be computed according to Equation (10.1).

$$V_{IMON} = \sum_{i=1}^{N}\left(A + B \cdot I_{OUT_i}\right) \cdot \frac{R_{ishare}}{N} = \frac{N \cdot A + B \cdot \left(I_{OUT_1} + \cdots + I_{OUT_N}\right)}{N} \cdot R_{ishare} = \left(A + B \cdot \frac{I_{OUT}}{N}\right) \cdot R_{ishare} \quad (10.1)$$

where:

- I_{OUTi} is the output current of supply i
- I_{OUT} is the total output current (sum of all I_{OUTi})
- (I_{OUT}/N) is the target output current per supply
- A is the IMON DAC output current at I_{OUTi} = 0 A
- B is a scaling factor converting supply output current to IMON DAC output current as given in Equation (10.2):

$$B = 10\ \mu A \cdot \frac{ishr_scale}{16} \quad (10.2)$$

A block diagram of the current sharing function including the connections to the current DAC and TSADC is shown in Figure 86.

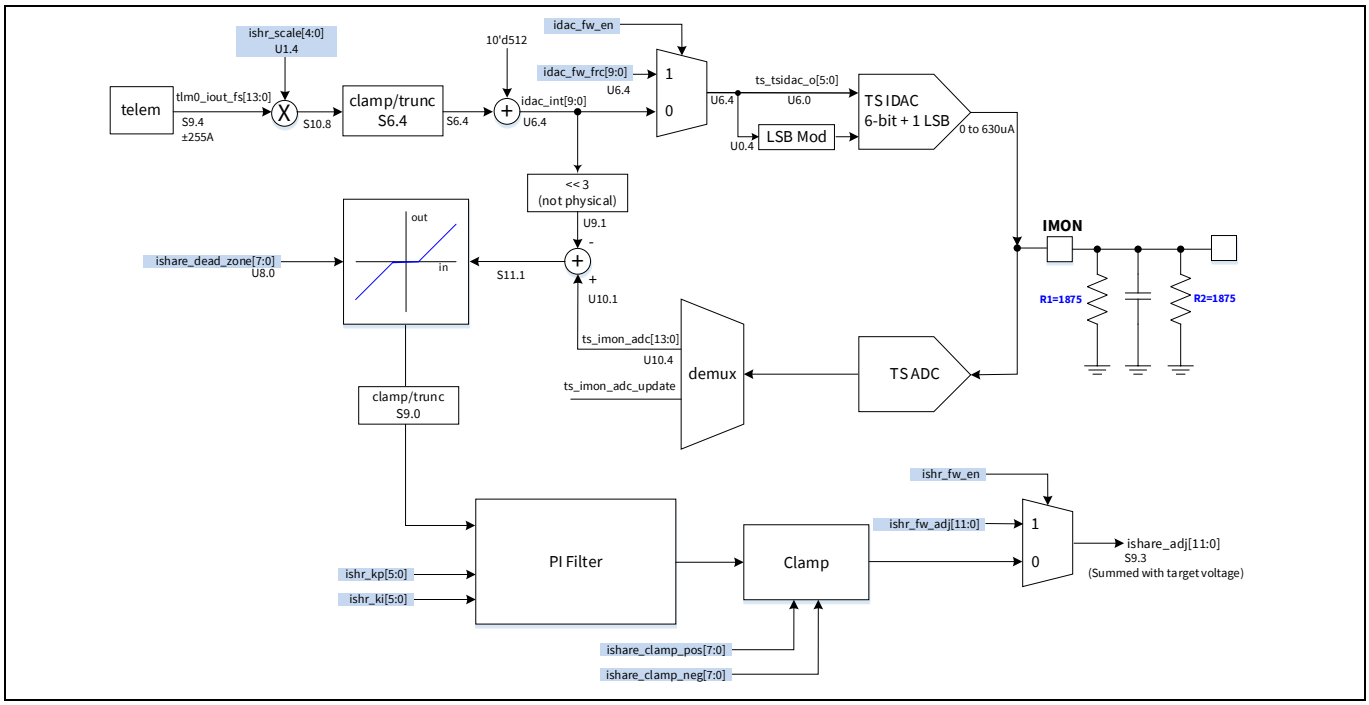

Figure 86 Simplified current sharing block diagram

The measured total current is received from the telemetry block and scaled by register **ishr_scale** to fit the range of the current DAC. The value for the **ishr_scale** is computed according to Equation (10.3), where the supply current is given in amps. As an example, if an individual supply can handle 50 A, then the **ishr_scale** value is as given in Equation (10.4).

$$ishr_scale = integer\left(16 * \frac{32}{I_supply_max}\right) \quad (10.3)$$

$$integer\left(16 * \frac{32}{50}\right) = 10 \quad (10.4)$$

The scaled current is then offset to allow the DAC to also represent negative currents using the bottom half of its range. The DAC is a 6-bit current DAC with four additional LSBs used for LSB modulation. Thus, it is possible to achieve approximately 10 bits of voltage resolution across the R_{ishare} resistor when a filter capacitor is connected between the IMON pin and ground.

The TSADC measures the voltage at the IMON pin and converts it to a digital representation. An error current is generated from the difference between:

- The measured IMON voltage (representing the target current I_{OUT}/N)
- The scaled current from the telemetry block

The scaled current from the telemetry block is further scaled by a factor of 8 (the left shift by 3 shown in Figure 86) to align the LSBs of the TS IDAC and the TS ADC.

$$\frac{TS\ IDAC\ LSB * R1}{TS\ ADC\ LSB} = \frac{10uA * 1875\Omega}{2.34375mV} = \frac{18.75mV}{2.34375mV} = 8 \quad (10.5)$$

A programmable deadzone is provided through register **ishare_dead_zone**, allowing the current error to be zeroed out within a zone of current around 0 A. This parameter is applied in a +/- sense. That is, the scaled and offset output current less the TS ADC result must be greater than +**ishare_dead_zone** or less than -**ishare_dead_zone** to adjust the output voltage. Subsequent to the dead zone, the error current proceeds to a PI compensation filter (described in Section 10). The output of this PI filter is in the format of an adjustment term to the target voltage. Both positive and negative voltage adjustments are allowed using the following registers:

- **ishare_clamp_pos**
- **ishare_clamp_neg**

The registers can be used to limit the magnitude of the target voltage adjustment or to limit the adjustment to a single direction only. To prevent adjustment in a given direction, the associated clamp parameter needs to be set to 0.

A current sharing fault is declared under the following conditions:

- If the PI filter output exceeds the register value **ishare_clamp_pos** and the output of the deadzone function is greater than 0
- If the PI filter output is below the register value **ishare_clamp_neg** and the output of the deadzone function is less than 0

10.2 Current sharing PI filter

The error current is processed by the PI filter, which consists of:

- A proportional term operating on the instantaneous current error
- An integral term operating on the accumulated current error

The integral term sets the low-frequency gain and the proportional term sets the high-frequency gain of the filter. The magnitude response of the filter is defined by Equations (10.6) to (10.9).

$$|H_{PI}(f)| = \sqrt{{K_P}^2 + \left(\frac{K_I}{2\pi \cdot T_{SAMPLE} \cdot f}\right)^2} \quad (10.6)$$

$$K_P = (8 + ishr_kp[2:0]) \cdot 2^{(ishr_kp[5:3]-10)} \quad (10.7)$$

$$K_I = (8 + ishr_ki[2:0]) \cdot 2^{(ishr_ki[5:3]-12)} \quad (10.8)$$

$$T_{SAMPLE} = \frac{1}{F_{SWITCH}} \quad (10.9)$$

Register values **ishr_kp** and **ishr_ki** larger than 55 are clamped to 55. Figure 87 shows an example magnitude response with the following values:

- **ishr_kp** = 24
- **ishr_ki** = 16
- F_{switch} = 250 kHz

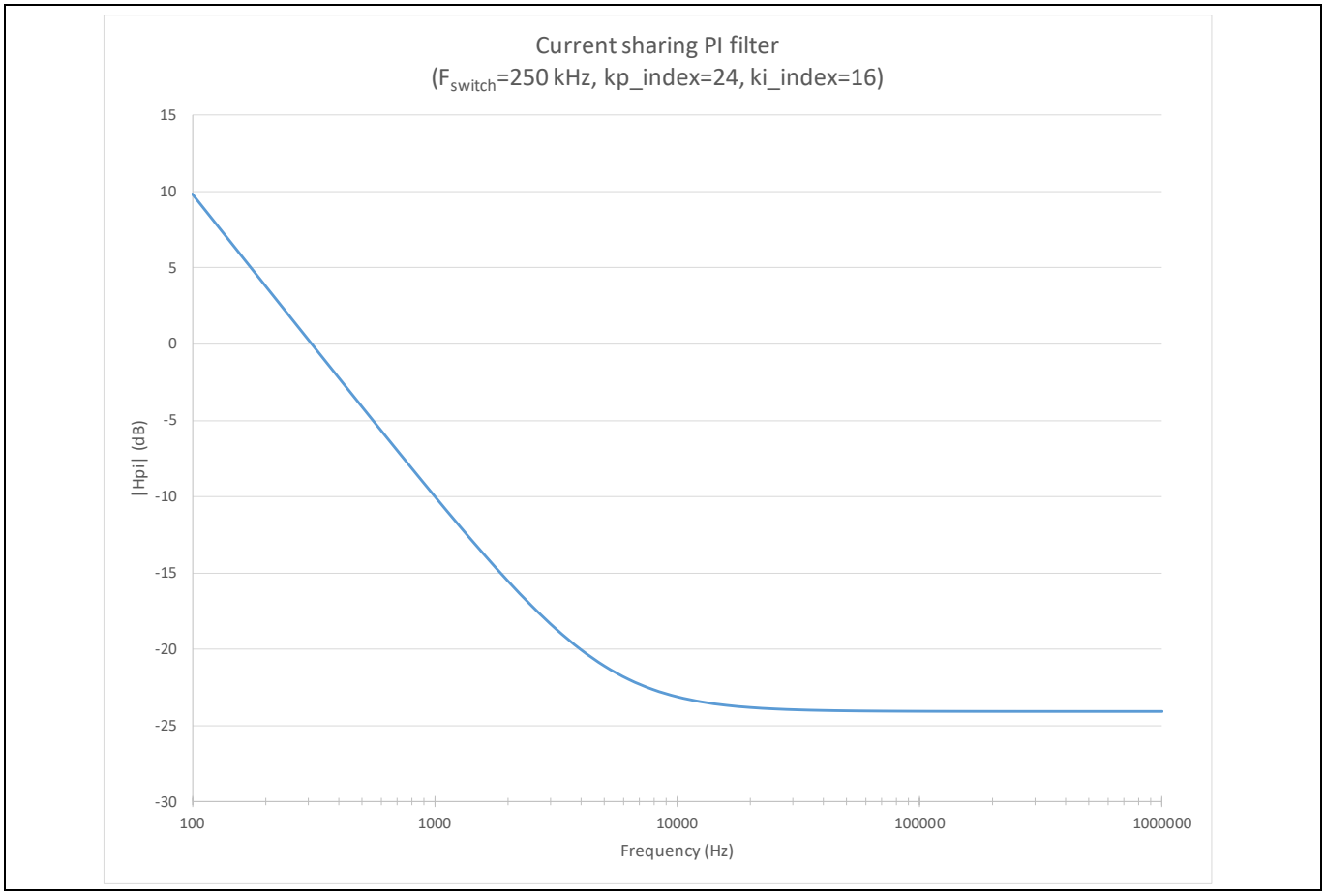

Figure 87 Example PI magnitude plot for ishr_kp = 24, ishr_ki = 16, F_{switch} = 250 kHz

10.3 Current sharing FW override

The current share PI filter output, ishare_adj, may be overridden through the following registers:

- **ishr_fw_en**
- **ishr_fw_adj**

Similarly, the IMON current DAC inputs may be overridden through the registers:

- **idac_fw_en**
- **idac_fw_frc**

These overrides could be used as part of a user-written FW patch to implement a different current sharing scheme.

10.4 Current sharing pin, DAC and ADC configuration

Some additional parameters are required to be set for the proper active current sharing operation. Table 68 shows relevant register programming in order to configure the IMON pin for current sharing. Correspondingly, Table 69 shows the register settings in order to configure the DAC and TSADC for current sharing.

Table 68 IMON pin current sharing parameter configuration

Register	Current sharing setting	Notes
imon_func	0	Connects IMON pin to the DAC and TSADC
imon_pd	0	Disables weak pull-down resistor on IMON pin
imon_pu_n	1	Disables weak pull-up resistor on IMON pin
imon_ppen	0	Sets digital output drive to open drain

Table 69 Current DAC and TSADC current sharing parameter configuration

Register	Current sharing setting	Notes
imon_meas_en	1	Allows IMON input to be measured by TSADC
ts_tsidac_imon_sel	1	Connects current DAC output to IMON pin
ts_muxmode	4, 6 or 7	These settings include measurement of the IMON input in the TSADC input mux auto sequencing

10.5 R_{ishare} FW patch

The average current value at the IMON pin depends on the configuration and sequencing of the parallel connected power supply modules. Therefore, the R_{ishare} resistor needs to be connected differently depending on the module configuration. The following resistor connections are to be applied:

- If parallel modules are all enabled simultaneously from start-up, R_{ishare} is connected between IMON pin and ground (Figure 85 and Figure 86)
- If parallel modules start in a sequence or are disabled for power reduction, R_{ishare} is connected between:
  - IMON pin and PWM6 for XDPP1100-Q024 (Figure 88)
  - IMON pin and PWM6 or PWM11 for XDPP1100-Q040 (Figure 88)

For the second R_{ishare} connection with module sequencing, a FW patch is available. It drives the R_{ishare} negative connection to ground through a PWM output allowing the negative connection to float when the supply is not enabled. The PWM pin selection is hard-coded in the FW patch.

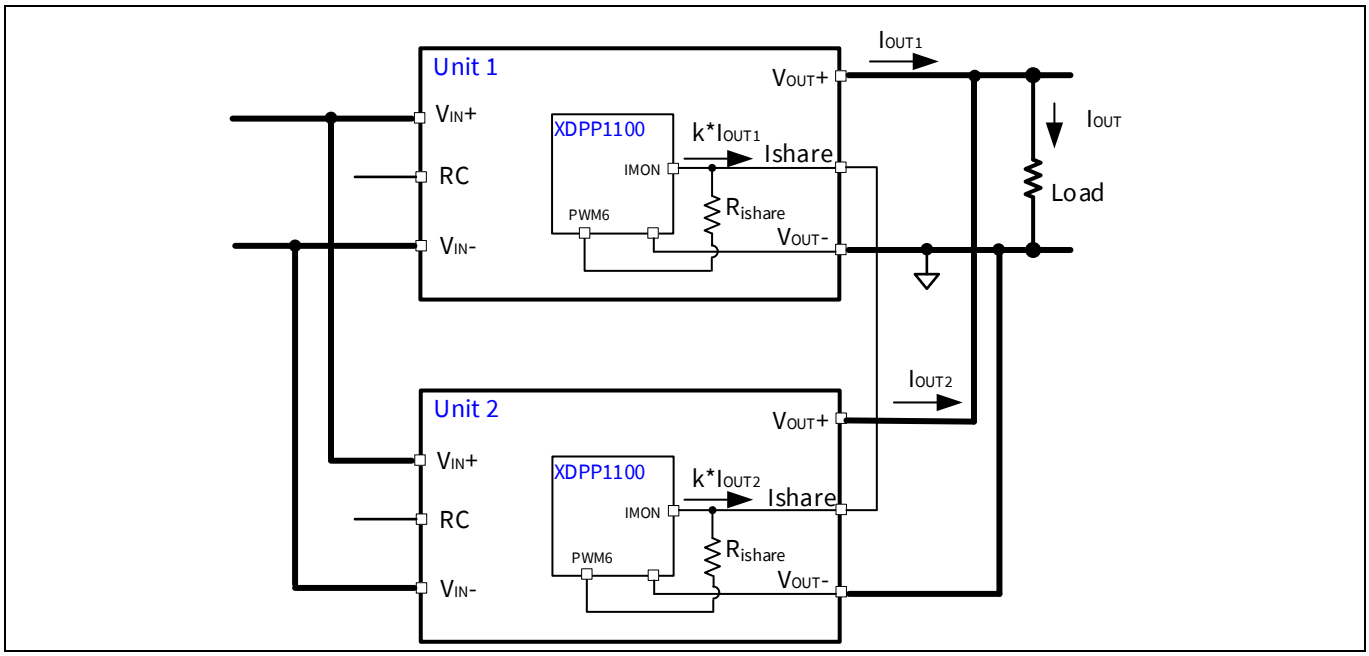

Figure 88 Active current sharing example with two units in parallel while start-up sequencing is applied

10.6 Current sharing registers

The relevant current sharing registers and their descriptions are provided in Table 70.

Table 70 Current sharing register descriptions

Peripheral	Field name	Access	Address	Bits	Description
common	imon_func	RW	7000_3008h	[2:0]	Pin IMON function definition. 0: IMON analog IO (select for current sharing) 1: GPIO0[3] digital IO 2: GPIO1[3] digital IO 3: SYNC digital IO 4: FAN1_TACH digital input 5 to 7: Not used
common	imon_pd	RW	7000_3008h	[3]	Pin IMON weak pull-down enable. 0: Pull-down disabled (select for current sharing) 1: Pull-down enabled
common	imon_pu_n	RW	7000_3008h	[4]	Pin IMON weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled (select for current sharing)
common	imon_ppen	RW	7000_3008h	[5]	Pin IMON output buffer CMOS/open drain select 0: Open drain output (select for current sharing) 1: CMOS output

Peripheral	Field name	Access	Address	Bits	Description
common	ishr_scale	RW	7000_3020h	[4:0]	Used for current sharing, this register defines a pre-scale gain applied to the internal current telemetry before sending to the current output DAC on the IMON pin. Its settings should be computed as follows: ishr_scale = integer(16 * (32 / max. current in amps))
common	ishr_kp	RW	7000_3020h	[10:5]	Current sharing PI filter proportional coefficient index. Note that index settings greater than 55 are clamped to 55. kp_exp = ishr_kp[5:3] kp_man = 8 + ishr_kp[2:0] kp = kp_man * 2^kp_exp * 2^-10
common	ishr_ki	RW	7000_3020h	[16:11]	Current sharing PI filter integral coefficient index. Note that index settings greater than 55 are clamped to 55. ki_exp = ishr_ki[5:3] ki_man = 8 + ishr_ki[2:0] ki = kp_man * 2^ki_exp * 2^-12
common	ishare_clamp_neg	RW	7000_3024h	[7:0]	Negative clamp applied to active current sharing voltage adjustment. This value reflects the voltage at the VSEN input. To convert to V_{OUT} divide by VOUT_SCALE_LOOP. LSB = -1.25 mV, range = 0.0 to -318.75 mV
common	ishare_clamp_pos	RW	7000_3024h	[15:8]	Positive clamp applied to active current sharing voltage adjustment. This value reflects the voltage at the VSEN input. To convert to V_{OUT} divide by VOUT_SCALE_LOOP LSB = 1.25 mV, range = 0.0 to 318.75 mV
common	ishr_fw_adj	RW	7000_3084h	[11:0]	When ishr_fw_en is high, this register overrides the HW current share voltage adjust output with a FW-controlled setting. LSB = 156.25 µV, range = -320 to +319.84375 mV
common	ishr_fw_en	RW	7000_3084h	[12]	Enables FW-controlled current share loop via ishr_fw_adj. 0: Use HW-computed current share adjust 1: Use ishr_fw_adj
common	idac_fw_frc	RW	7000_3088h	[9:0]	When idac_fw_en is high, this register overrides the HW current DAC output with a FW-controlled setting. Note that more than four LSBs of resolution are achieved

Peripheral	Field name	Access	Address	Bits	Description
					through LSB modulation, which requires a filter cap attached to the IMON pin. LSB = 0.625 µA, range = 0.0 to 639.375 µA
common	idac_fw_en	RW	7000_3088h	[10]	Enables FW-controlled current DAC output via idac_fw_frc. 0: Use HW-determined current DAC output 1: Use idac_fw_frc
common	ishare_dead_zone	RW	7000_30A0h	[7:0]	I_{share} deadzone below which current sharing is not attempted. This parameter is applied in a +/- sense, that is, the quantity (Iout * ishr_scale - TS ADC result) must be greater than +ishare_dead_zone or less than -ishare_dead_zone to move the target output voltage. To convert to output current: Dead Zone (Amps) = 2 * ishare_dead_zone / ishr_scale This register is computed by FW based on PMBus command MFR_ISHARE_THRESHOLD. LSB = 1 code, range = 0 to 255 codes
tsen	imon_meas_en	RW	7000_4C00h	[1]	TSADC IMON input measurement enable. When enabled, the TSADC will measure the IMON input when selected by ts_muxmode and tx_muxctrl2. When disabled, no IMON measurement will occur, even if selected by ts_muxmode and ts_muxctrl2. 0: Disabled 1: Enabled
tsen	ts_tsidac_imon_sel	RW	7000_4C00h	[8]	IMON output current DAC enable. This current DAC should be enabled when using the IMON pin for current sharing. The current DAC should be disabled otherwise. 0: Disabled 1: Enabled
tsen	ts_muxctrl2	RW	7000_4C00h	[19:17]	TSADC MUX2 input source select. The output of MUX2 is connected to the TSADC input. Generally, this mux is auto-sequenced by the TSADC (see ts_muxmode below) but may be overridden by setting parameter ts_tsmuxmode=0.

Peripheral	Field name	Access	Address	Bits	Description
					0: 0.6 V reference (test only) 1: PRISEN 2: IMON 3: ATSEN 4: BTSEN 5: XADDR1 unfiltered 6: XADDR2 unfiltered 7: MUX1 (see ts_tsmuxctrl1)
tsen	ts_muxmode	RW	7000_4C00h	[22:20]	TSADC input sequence control. When bit [2] is 0, the TSADC input is entirely determined by the settings of ts_muxctrl1 and ts_muxctrl2. When bit [2] is 1, MUX2 auto-sequences its input using the pattern in the table below. If the sequence includes MUX2 input 7 (MUX1), the source in this time slot is determined by the setting of ts_muxctrl1. 0 to 3: Defined by ts_muxctrl1,2 4: Auto-sequence: 1, 2, 1, 3, 1, 2, 1, 4 5: Auto-sequence: 1, 5, 1, 3, 1, 7, 1, 4 6: Auto-sequence: 1, 5, 2, 3, 1, 7, 2, 4 7: Auto-sequence: 1, 5, 2, 3, 1, 7, 6, 4 Settings 4, 6 and 7 are suitable for IMON-based active current sharing
tsen	ts_imon_adc	R	7000_4C18h	[13:0]	Gain and offset corrected IMON TSADC output. LSB = 0.0625 ADC code, range = 0.0 to 1023.9375 ADC codes

10.7 Current sharing PMBus commands

The relevant current sharing-related PMBus commands and their descriptions are provided in Table 71.

Table 71 Current sharing-related PMBus commands

Command name	Access	Length	Address	Bits	Description
MFR_ISHARE_THRESHOLD	RW	Word	C6h	[15:0]	Defines the current sharing deadzone in amps in LINEAR11 format. This is a patched MFR PMBus command and is only valid if the function is defined in the patch

11 Current balance (I_{BAL})

This chapter describes the current balance functionality and the relevant registers and their programming.

Within interleaved topologies, different inductor currents may become imbalanced. This could be due to:

- Component mismatches
- Driver pulse width absorption
- Transient timing with respect to the phase timing

XDPP1100 supports interleaved topologies of a single loop with up to two phases driving a common output voltage. Figure 89 shows an example of an interleaved HBCT topology, where the inductor currents in L_{OUT1} and L_{OUT2} may become imbalanced.

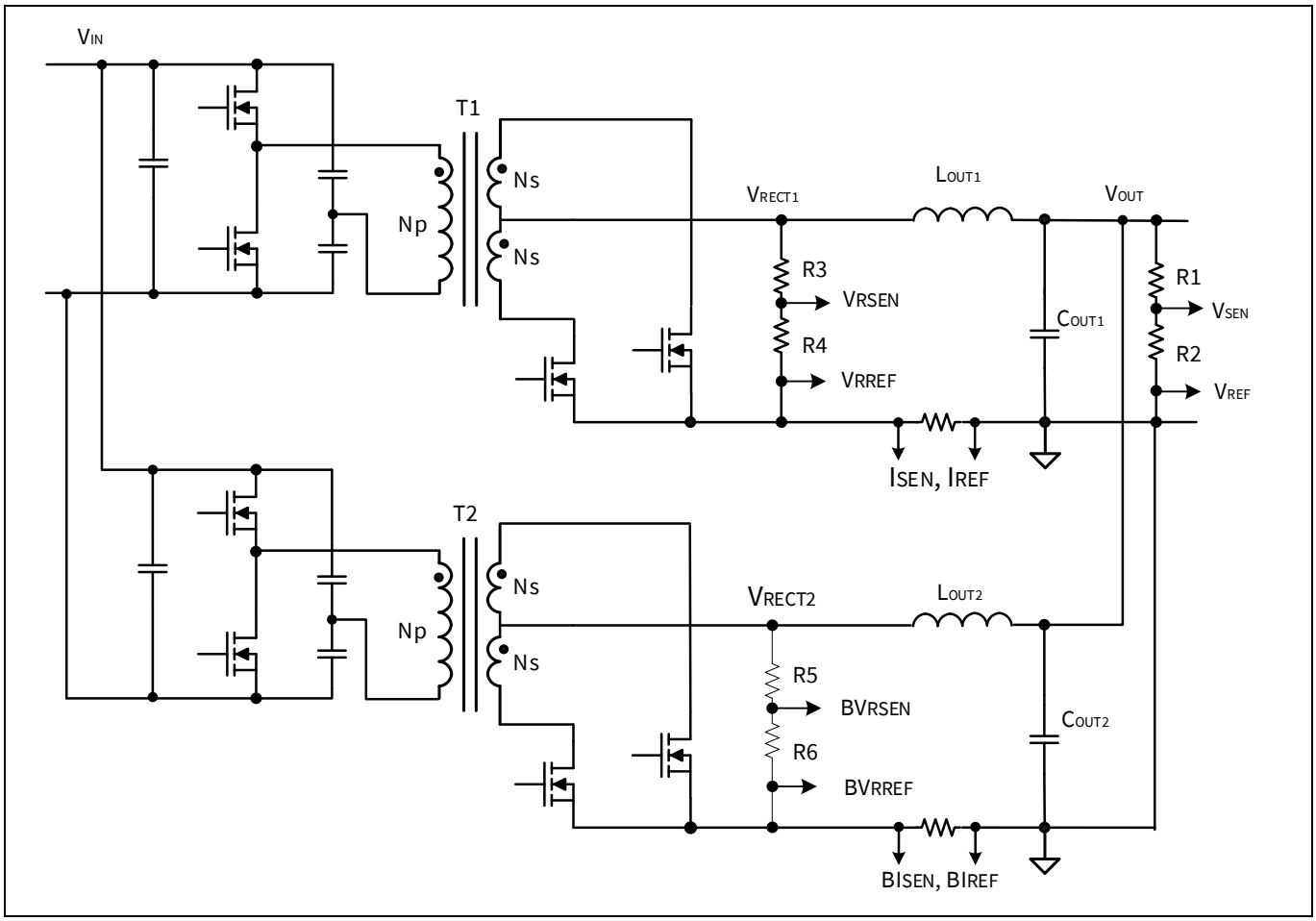

Figure 89 Interleaved HBCT system diagram

11.1 Current balance circuit

The XDPP1100 provides a HW-based current balance circuit. It aims to maintain equal currents on average in the two phases of the interleaved converter by adjusting the duty cycle in one half of interleave. A simplified block diagram of the current balance circuit is shown in Figure 90.

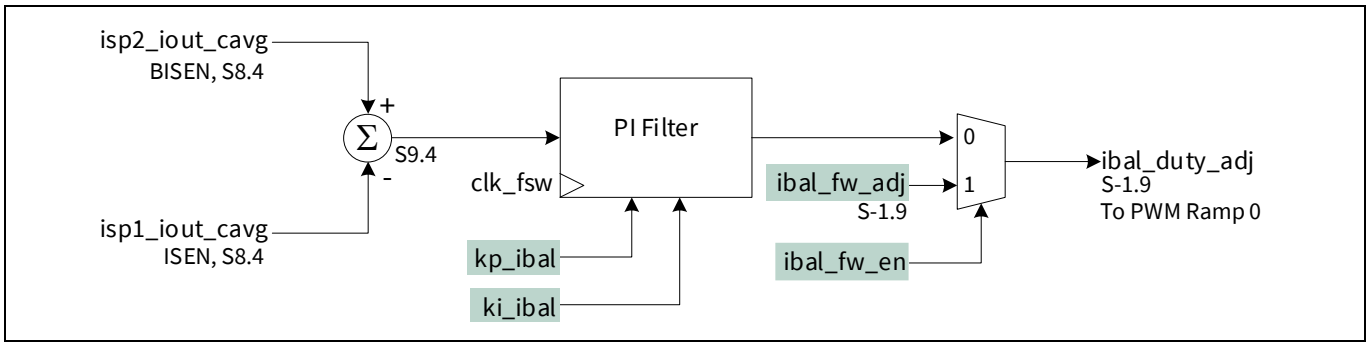

Figure 90 Simplified current balance block diagram

The circuit receives as its input the cycle-averaged current outputs from ISP1 and ISP2, which correspond to the ISEN and BISEN sensed currents. The difference between these current inputs is computed and used as the input to a PI compensation filter. The filter output, ibal_duty_adj, is capable of providing up to ±25 percent duty-cycle adjustment on the PWM ramp0. Since the difference is computed as (BISEN – ISEN) and the adjustment is provided on ramp0 only, it is required that:

- ramp0 is used to generate the PWM outputs associated with the phase whose current is sensed by ISEN
- ramp1 is used to generate the PWM outputs associated with the phase whose current is sensed by BISEN

In case of concern about current sense accuracy at low currents, register **ibal_en_thresh** makes it possible to dynamically disable the current balance at low currents. The programming of this register is shown in Table 72. A setting to always enable the current balance is also provided.

Table 72 Current balance current enable threshold programming

ibal_en_thresh	Current balance enable threshold
0, 1	Always enabled
2	3 A
3	5 A

11.2 Current balance PI filter

The difference between ISEN and BISEN sensed currents is processed by the PI filter. It consists of:

- a proportional term, which operates on the instantaneous current error
- an integral term, which operates on the accumulated current error

The integral term sets the low-frequency gain, and the proportional term sets the high-frequency gain of the filter. The magnitude response of the filter is defined by Equations (11.1) to (11.4).

$$|H_{PI}(f)| = \sqrt{{K_P}^2 + \left(\frac{K_I}{2\pi \cdot T_{SAMPLE} \cdot f}\right)^2} \quad (11.1)$$

$$K_P = (8 + kp_ibal[2:0]) \cdot 2^{(K_P_ibal[5:3]-14)} \quad (11.2)$$

$$K_I = (8 + ki_ibal[2:0]) \cdot 2^{(K_I_ibal[5:3]-20)} \quad (11.3)$$

$$T_{SAMPLE} = \frac{1}{F_{SWITCH}} \quad (11.4)$$

Register values **kp_ibal** and **ki_ibal** larger than 55 are clamped to 55. Setting either of these registers to zero individually disables the proportional or integral component of the filter. It should be noted that when the integral component of the filter is disabled (either through **ki_ibal** = 0 or the current dropping below the value set in register **ibal_en_thresh**) the accumulated current error is held at its last value rather than being reset to 0.

Figure 91 shows an example magnitude response with the following values:

- **kp_ibal** = 48
- **ki_ibal** = 32
- F_{switch} = 250 kHz

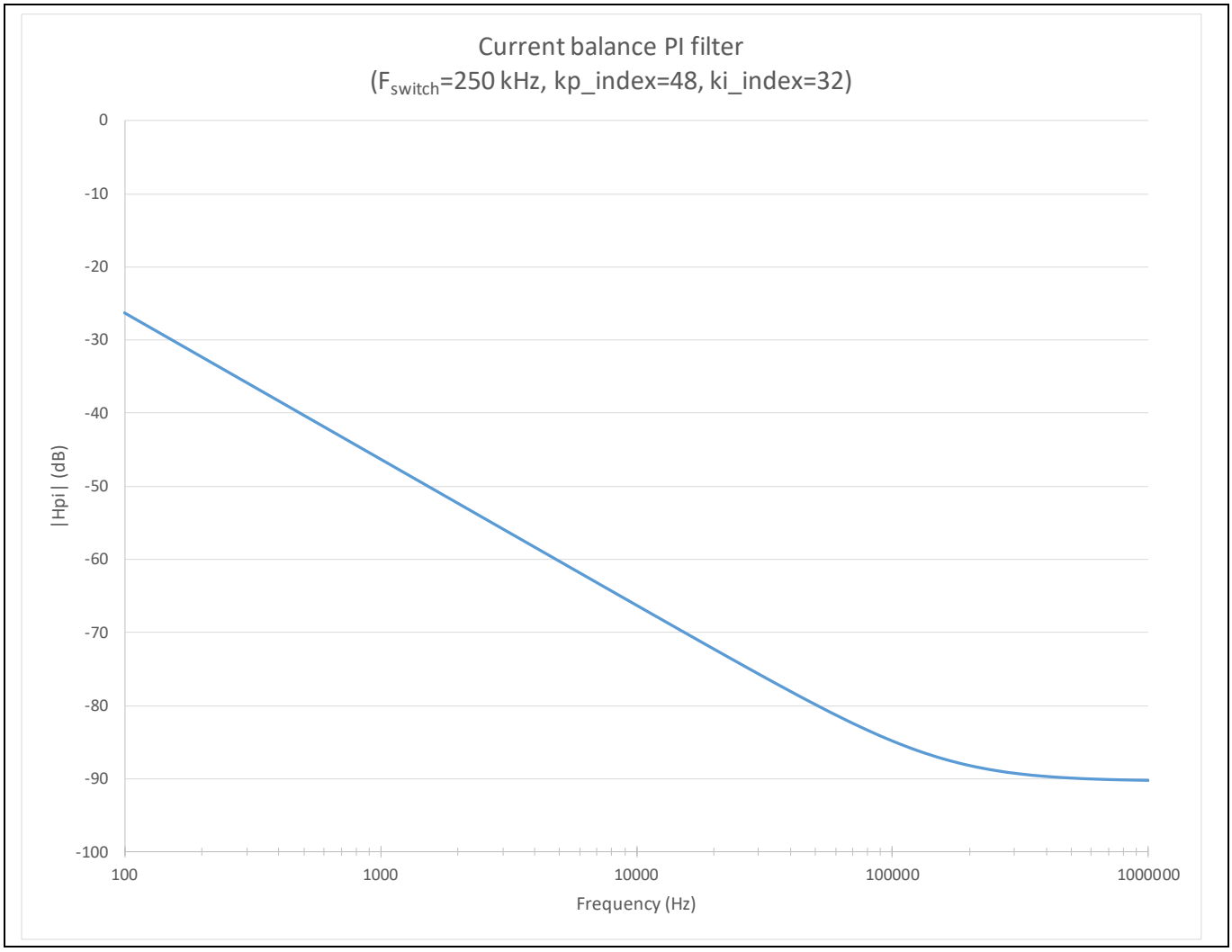

Figure 91 Example PI magnitude plot for kp_ibal = 48, ki_ibal = 32, F_{switch} = 250 kHz

11.3 Current balance FW override

The current balance output, ibal_duty_adj, may be overridden through the following registers:

- **ibal_fw_en**
- **ibal_fw_adj**

These overrides could be used as part of a user-written FW patch to implement a different current balance scheme.

11.4 Current balance registers

The relevant current balance registers and their descriptions are provided in Table 73.

Table 73 Current balance register descriptions

Peripheral	Field name	Access	Address	Bits	Description
common	kp_ibal	RW	7000_3000h	[5:0]	Current balance PI filter proportional coefficient index. Set to 0 to disable the proportional component of the filter. Note that index settings greater than 55 are clamped to 55. kp_exp = kp_ibal[5:3] kp_man = 8 + kp_ibal[2:0] kp = kp_man * 2^kp_exp * 2^-14
common	ki_ibal	RW	7000_3000h	[11:6]	Current balance PI filter integral coefficient index. Set to 0 to disable the integral component of the filter. Note that index settings greater than 55 are clamped to 55. ki_exp = ki_ibal[5:3] ki_man = 8 + ki_ibal[2:0] ki = kp_man * 2^ki_exp * 2^-20
common	ibal_en_thresh	RW	7000_3000h	[27:26]	Total current level above which current balancing is enabled in interleaved (multiphase) topologies. Set to a higher value if concerned about low current accuracy. 0, 1: Always enabled 2: 3 A 3: 5 A
common	ibal_fw_adj	RW	7000_3078h	[7:0]	When ibal_fw_en is high, this register overrides the HW current balance duty-cycle adjust output with a FW-controlled setting. LSB = 0.195 percent, range = 0.0 to 24.805 percent
common	ibal_fw_en	RW	7000_3078h	[8]	Enables FW-controlled current balance loop via ibal_fw_adj. 0: Use HW-computed current balance adjust 1: Use ibal_fw_adj

12 Flux balance (F_{BAL})

This chapter discusses the flux balance feature as well as the relevant registers and their programming.

Flux imbalance is a known problem within isolated FB converters (Figure 92). In these topologies timing mismatch can cause the applied volt-seconds across the transformer during one half-cycle to be greater than the volt-seconds during the opposite half-cycle. This might lead to transformer saturation due to flux “walkaway”.

Typical solutions to avoid flux imbalance are:

- Primary-side PCMC (requires current transformer)
- VMC with capacitor in series with primary winding
- VMC with oversized transformer or gapped transformer

Each method has drawbacks, adding either BOM cost or board area, or reducing efficiency. To overcome these drawbacks while obtaining flux balance, the XDPP1100 contains two flux balance circuits to support interleaved designs. It is capable of maintaining volt-second balance between half-cycles by using measured even/odd half-cycle rectification voltages and pulse widths from the VS function as described in Section 2.

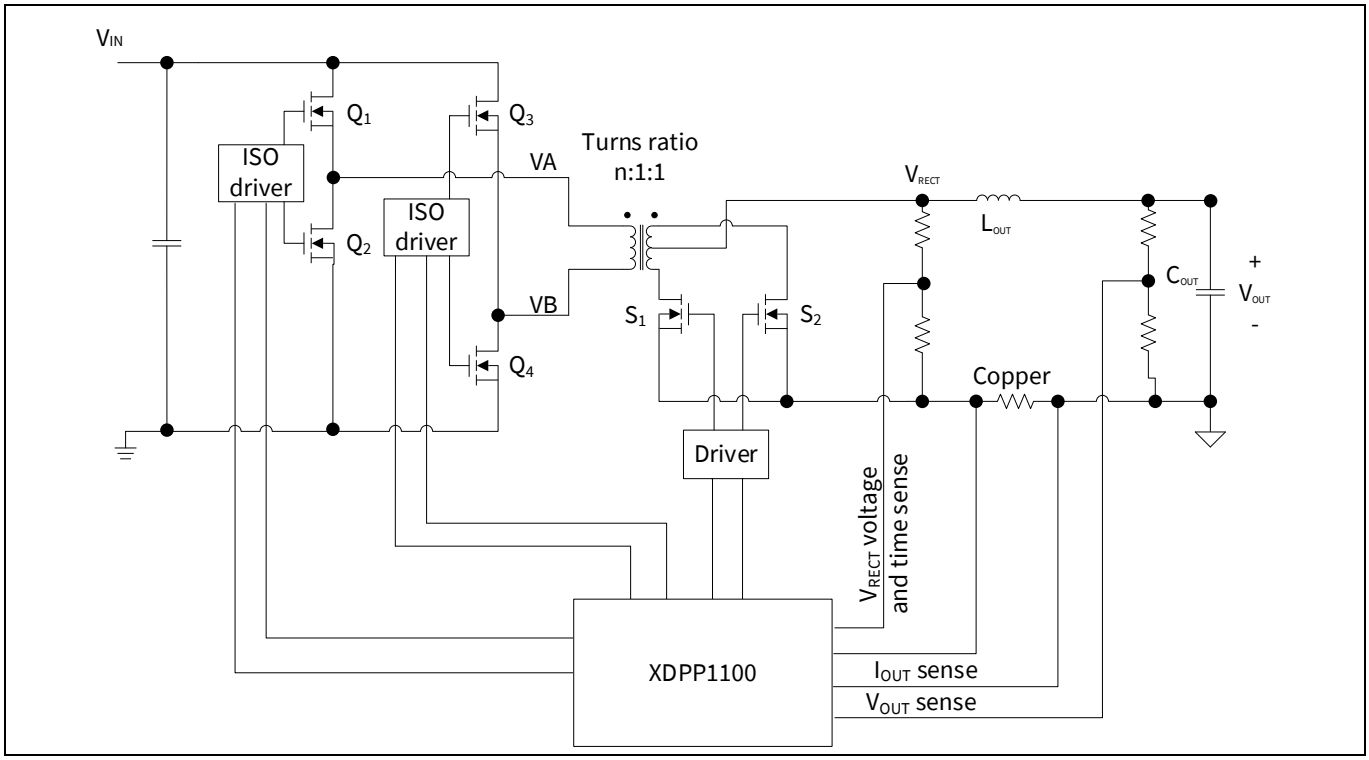

Figure 92 FBCT system diagram

12.1 Flux balance circuit

A simplified block diagram of the flux balance circuit is shown in Figure 93. The following signals are the inputs to the flux balance circuit:

- vrs_vrect_even and vrs_vrect_odd, which are the measured even and odd half-cycle rectification voltages from the VS function
- cnt_vrscomp_even and cnt_vrscomp_odd, which are the measured even and odd half-cycle rectification voltage pulse widths from the VS function

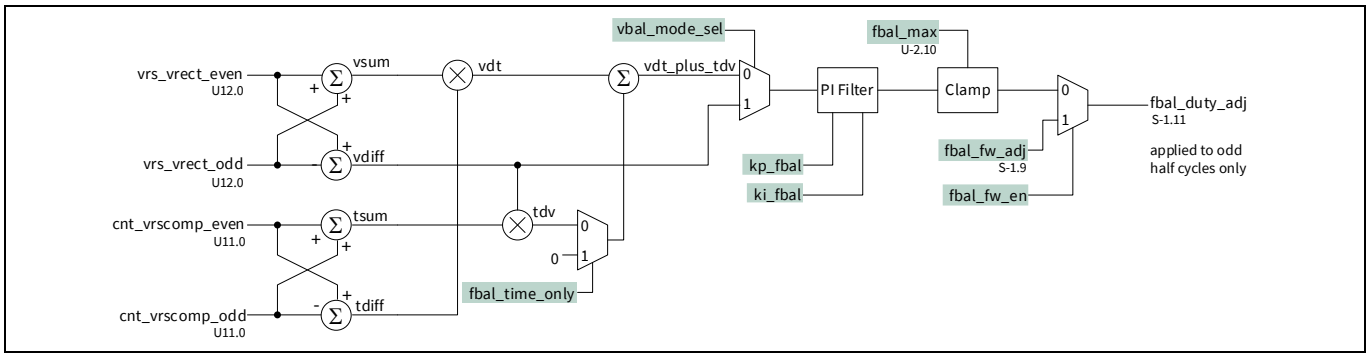

Figure 93 Simplified flux balance block diagram

The flux balance circuit supports three balancing modes:

- Volt-second balance
- Time-only balance
- Voltage-only balance

Volt-second and time-only modes are suitable for FB topologies. It should be noted that time-only mode is sufficient if timing mismatch is the main concern. The voltage-only mode is not suitable for FB topologies. However, it could be used for HB topologies, but due to the lack of significant improvement in the system performance this is not recommended.

The flux balance mode programming is performed through registers **vbal_mode_sel** and **fbal_time_only**, as summarized in Table 74.

Table 74 Flux balance mode programming

vbal_mode_sel	fbal_time_only	Balance mode	PI filter error input
0	0	Volt-seconds	(Vrect_even + Vrect_odd) * (Teven - Todd) + (Vrect_even - Vrect_odd) * (Teven + Todd) = 2 * (Vrect_even*Teven - Vrect_odd*Todd)
0	1	Time	(Vrect_even + Vrect_odd) * (Teven - Todd) = Vrect_avg * (Teven - Todd)
1	x	Voltage	Vrect_odd - Vrect_even

Subsequent to the balance mode selection, the computed error voltage is filtered by a PI compensation filter. The filter output allows for maximum duty-cycle correction of ± 25 percent on the odd half-cycles only. Further limitation on the maximum duty-cycle correction is done via register **fbal_max**. Setting **fbal_max** = 0 blocks the flux balance circuit from adjusting the PWM duty cycle.

The flux balance circuits are enabled automatically by ROM-based FW, under the following conditions:

- FBAL1 associated with Loop 0 phase 1, VRSEN and PWM ramp0
  - topology = FB or HB
  - control mode = VMC
  - Loop 0 is operating (start-up or regulation)
- FBAL2 associated with Loop 0 phase 2, BVRSEN and PWM ramp1
  - topology = FB or HB
  - control mode = VMC

- Loop 0 is operating (start-up or regulation)
- interleave operation enabled via PMBus command PAGE0.FW_CONFIG_REGULATION.INTERLEAVE_ENABLE=1

If flux balance is not desired under the above conditions, the adjustment can be disabled by one of the following two methods:

- Setting **fbal_max** = 0 blocks the flux balance circuit from adjusting the PWM duty cycle (recommended method)
- Setting both PI filter coefficients to 0 in the OTP stored configuration sets the filter gain to 0, effectively disabling the flux balance adjustment

In addition to the above conditions, registers **rampX_dutyc_lock** must be set to 1 to enable even/odd duty-cycle locking in the PWM. Note that the flux balance duty-cycle adjustment is applied after the duty-cycle lock in the PWM function.

12.2 Flux balance PI filter

The voltage error is processed by the PI filter, which consists of:

- A proportional term operating on the instantaneous volt-second or time error
- An integral term operating on the accumulated volt-second or time error

The integral term sets the low-frequency gain and the proportional term sets the high-frequency gain of the filter. The magnitude response of the filter is defined by Equations (12.1) to (12.4).

$$|H_{PI}(f)| = \sqrt{kp^2 + \left(\frac{ki}{2\pi \cdot Ts \cdot f}\right)^2} \tag{12.1}$$

$$kp = (8 + kp_fbal[2:0]) \cdot 2^{(kp_fbal[5:3]-18)} \tag{12.2}$$

$$ki = (8 + ki_fbal[2:0]) \cdot 2^{(ki_fbal[5:3]-22)} \tag{12.3}$$

$$Ts = \frac{1}{F_{switch}} \tag{12.4}$$

Register values **kp_fbal** and **ki_fbal** larger than 55 are clamped to 55. Setting **kp_fbal** or **ki_fbal** to 0 individually disables the proportional or integral filter component. Note that when the integral component is disabled through setting **ki_fbal** = 0, the accumulated current error is reset to 0. Figure 94 shows an example magnitude response with the following values:

- **kp_fbal** = 48
- **ki_fbal** = 32
- F_{switch} = 250 kHz

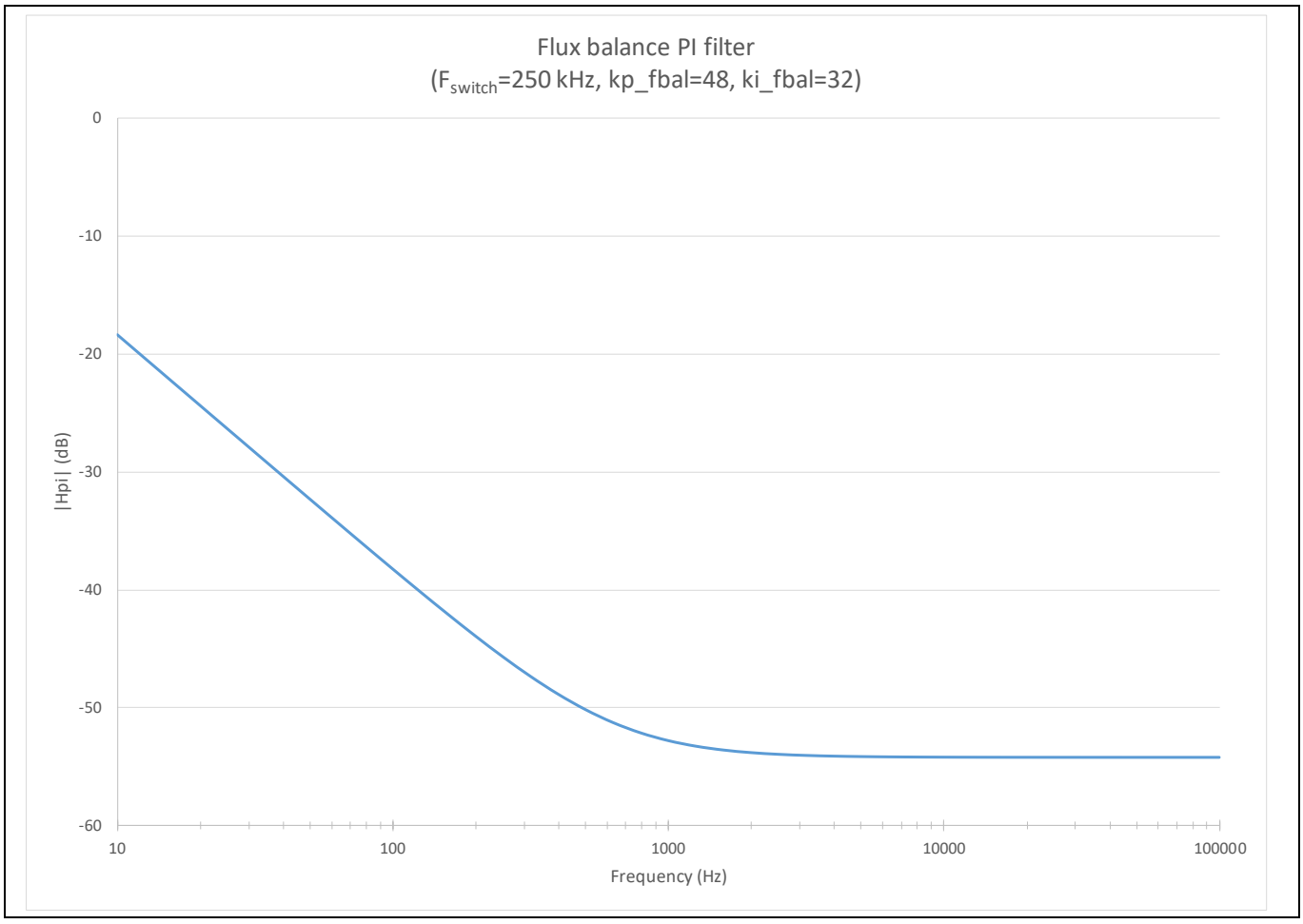

Figure 94 Example PI magnitude plot for kp_fbal = 48, ki_fbal = 32, F_{switch} = 250 kHz

12.3 Flux balance FW override

The flux balance output, fbal_duty_adj, may be overridden through the following registers (where X = 1, 2):

- **fbalX_fw_en**
- **fbalX_fw_adj**

These overrides could be used as part of a user-written FW patch to implement a different current balance scheme.

12.4 Flux balance DCM operation

At light current load, the output inductor current flows in a negative direction if FETs are employed as the secondary rectifier (SR) devices. If the SR FETs are disabled or diodes are used in their place, the inductor current is discontinuous. In this discontinuous conduction mode (DCM), the V_{RECT} rising edge can occur prior to the primary PWM rising edge due to the negative inductor current driving V_{RECT} high. This means the V_{RECT} pulse width is no longer a good indication of the primary-side PWM pulse width mismatch between the half-cycles. Therefore, due to the inability to accurately measure the primary-side pulse width in DCM operation it is recommended to disable flux balance in this mode.

XDPP1100 provides several registers to control the flux balance behavior in and around DCM operation. These registers are:

- **fbal_dcm_thresh**, which sets the current threshold at which the flux balance function is disabled due to the DCM operation; a setting of 63 disables this feature (not recommended)
- **fbal_dcm_dis_cnt**, which defines the number of consecutive current samples (at F_{switch} rate) below **fbal_dcm_thresh** required to disable the flux balance function; a higher setting helps to reduce jitter near the DCM boundary
- **fbal_dcm_ena_cnt** defines the number of consecutive current samples (at F_{switch} rate) above **fbal_dcm_thresh** required to enable the flux balance function; a higher setting will help reduce jitter near the DCM boundary
- **fbal_dcm_0out_duty_adj** defines what action is taken with the **fbal_duty_adj** output on DCM disable, either freeze or reset to 0

12.5 Flux balance fault detection

If the flux balance circuit is unable to achieve balance between half-cycles, it can be configured to declare a fault. Flux balance fault detection diagram is shown in Figure 95.

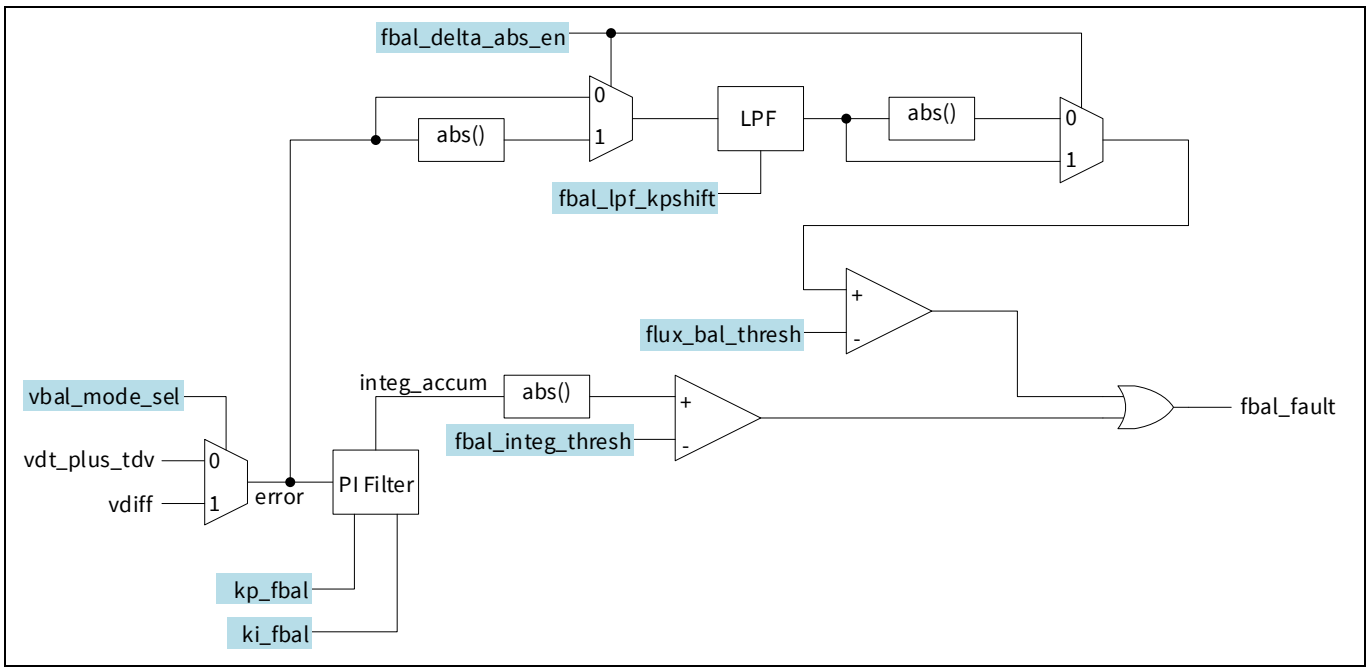

Figure 95 Flux balance fault detection block diagram

The following methods are provided to detect a flux balance fault:

- Absolute volt-second error
- Accumulated integrator error

To declare a flux balance fault, these error methods compare:

- For absolute error, a low-pass filtered version of the PI filter input to a specified register value in **flux_bal_thresh**
- For accumulated error, a PI filter integral term accumulated error to a specified register value in **fbal_integ_thresh**

A setting of 0 on **flux_bal_thresh** disables the absolute error fault method and, correspondingly, a setting of 0 on **fbal_integ_thresh** disables the accumulated error method.

In the case of absolute error method, register **fbal_delta_abs_en** selects where the absolute value of the error is applied, either before or after the LPF. The LPF BW programming is performed via register **fbal_lpf_kpshift** as shown in Table 75. For accumulated error method, register value **fbal_integ_thresh** selects the threshold in terms of the integrator full range as shown in Table 76.

Table 75 Flux balance fault detection filter BW programming

fbal_lpf_kpshift	LPF BW/F_{switch}	LPF BW (kHz) at F_{switch} = 250 kHz
0	Bypass LPF	Bypass LPF
1	0.159	39.79
2	0.053	13.26
3	0.023	5.68
4	0.011	2.65
5	0.005	1.28
6	0.003	0.63
7	0.001	0.31

Table 76 Flux balance integral accumulated error fault threshold

fbal_integ_thresh	Accumulated error threshold
0	Disabled
1	25 percent
2	12.50 percent
3	6.25 percent

As was shown in Figure 95, the results of the two error method comparisons are ORed together. Thereafter they are sent to the fault block for further processing, including being used to shut down the output.

12.6 Flux balance registers

The relevant volt-second balance registers are provided in Table 77.

Table 77 Volt-second balance-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
common	kp_fbal	RW	7000_3000h	[17:12]	Flux/Voltage balance PI filter proportional coefficient index. Set to 0 to disable the proportional component of the filter. Note that index settings greater than 55 are clamped to 55. Note also that flux balancing requires that duty-cycle locking is enabled by rampX_dutyc_lock. kp_exp = kp_fbal[5:3]

Peripheral	Field name	Access	Address	Bits	Description
					kp_man = 8 + kp_fbal[2:0] kp = kp_man * 2^kp_exp * 2^-18
common	ki_fbal	RW	7000_3000h	[23:18]	Flux/Voltage balance PI filter integral coefficient index. Set to 0 to disable the integral component of the filter. Note that index settings greater than 55 are clamped to 55. Note also that flux balancing requires that duty-cycle locking is enabled by rampX_dutyc_lock. ki_exp = ki_fbal[5:3] ki_man = 8 + ki_fbal[2:0] ki = ki_man * 2^ki_exp * 2^-22
common	vbal_mode_sel	RW	7000_3000h	[24]	Flux/Voltage balance mode select. Flux balance mode balances based on the volt-second product unless fbal_time_only is set to 1, in which case balancing is based only on the V_{RECT} pulse width. Voltage balance mode balances based on the V_{RECT} voltage. 0: Flux balance mode (select for FB primary) 1: Voltage balance mode (select for HB primary)
common	fbal_time_only	RW	7000_3000h	[28]	In the flux balance mode of the flux/voltage balance PI filter, select between volt-second or time-only balancing. 0: Volt-second balance mode 1: Time-only balance mode
common	fbal_max	RW	7000_3014h	[30:23]	Flux/Voltage balance max. correction. This register limits the maximum duty-cycle correction applied by the flux balance filter. LSB = 0.09766 percent, range = 0.0 to 24.90234 percent
common	flux_bal_thresh	RW	7000_3030h	[7:0]	Flux balance absolute error fault threshold. A setting of 0 disables this fault. LSB = 2 V-µs, range = 0 to 510 V-µs
common	fbal_integ_thresh	RW	7000_3030h	[9:8]	Flux balance integrator accumulated error fault threshold defined with respect to max. integrator range. 0: Disable 1: 25 percent or greater (railed)

Peripheral	Field name	Access	Address	Bits	Description
					2: 12.5 percent or greater 3: 6.25 percent or greater
common	fbal_lpf_kpshift	RW	7000_3030h	[12:10]	Flux balance error filter coefficient used in flux balance fault computation. Set to 0 to bypass filter. Kp = 2^-Kpshift F3db = [kp/(1-kp)] * F_{switch} / 2pi
common	fbal_delta_abs_en	RW	7000_3030h	[13]	In flux balance computation, determines whether absolute value applied to the error before or after the LPF. 0: Convert to ABS value after LPF 1: Convert to ABS value before LPF
common	fbal_dcm_thresh	RW	7000_3030h	[19:14]	When the inductor current goes negative the flux balance V_{RECT} pulse width measurement may become inaccurate. It is recommended to disable flux balancing at low current levels to prevent incorrect flux balance correction from occurring. This parameter defines the output current level below which the flux balance correction is disabled. A setting of 63 disables this feature. LSB = 0.5 A, range = 0.0 to 31.5 A
common	fbal_dcm_dis_cnt	RW	7000_3030h	[21:20]	Defines the number of consecutive current samples below fbal_dcm_thresh required to disable flux balancing. LSB = 1 T_{switch}, range = 1 to 4 T_{switch}
common	fbal_dcm_ena_cnt	RW	7000_3030h	[23:22]	Defines the number of consecutive current samples above fbal_dcm_thresh required to re-enable flux balancing. LSB = 1 T_{switch}, range = 1 to 4 T_{switch}
common	fbal_dcm_0out_duty_adj	RW	7000_3030h	[24]	When flux balancing is disabled due to low current, this parameter determines whether to zero out the flux balance duty-cycle correction or to freeze the duty-cycle correction at its current value. 0: Freeze the flux balance correction 1: Zero out the flux balance correction (recommended)

Peripheral	Field name	Access	Address	Bits	Description
common	fbal1_fw_adj	RW	7000_307Ch	[7:0]	When fbal1_fw_en is high, this register overrides the Loop 0, Phase 1 HW flux/voltage balance duty-cycle adjust output with a FW-controlled setting. LSB = 0.195 percent, range = 0.0 to 24.805 percent
common	fbal1_fw_en	RW	7000_307Ch	[8]	Enables FW-controlled flux/voltage balance loop via fbal1_fw_adj. 0: Use HW-computed flux/voltage balance adjust 1: Use fbal1_fw_adj
common	fbal2_fw_adj	RW	7000_3080h	[7:0]	When fbal2_fw_en is high, this register overrides the Loop 0, Phase 2 or Loop 1 HW flux/voltage balance duty-cycle adjust output with a FW-controlled setting. LSB = 0.195 percent, range = 0.0 to 24.805 percent
common	fbal2_fw_en	RW	7000_3080h	[8]	Enables FW-controlled flux/voltage balance loop via fbal2_fw_adj. 0: Use HW-computed flux/voltage balance adjust 1: Use fbal2_fw_adj
pwm	ramp0_dutyc_lock	RW	7000_2C00h	[16]	ramp0 duty-cycle lock enable. When enabled, the odd half-cycle duty cycle is locked to the even half-cycle duty cycle prior to applying any flux balance correction. Duty-cycle lock is required when using flux balancing but may also be used without flux balance. 0: Duty lock disabled 1: Duty lock enabled
pwm	ramp1_dutyc_lock	RW	7000_2C00h	[17]	ramp1 duty-cycle lock enable. When enabled, the odd half-cycle duty cycle is locked to the even half-cycle duty cycle prior to applying any flux balance correction. Duty-cycle lock is required when using flux balancing but may also be used without flux balance. 0: Duty lock disabled 1: Duty lock enabled

13 Fan support

This chapter describes the fan support functionality and the relevant register settings.

Fan support is provided through a combination of HW blocks and a FW patch. The HW blocks drive the fan PWM output and sense the fan speed tachometer input. These blocks are described in Sections 13.1 through 13.3. FW patch is required to enable the fan-related PMBus commands described in Section 13.4. The required commands are provided as part of the patch_user_app patch file.

The fan HW blocks are enabled through the PMBus command FAN_CONFIG_1_2 as follows:

- FAN_CONFIG_1_2[7] enables/disables FAN1
- FAN_CONFIG_1_2[3] enables/disables FAN2

13.1 Fan PWM output

The XDPP1100 provides HW support for up to two fan PWM outputs, identified as:

- FAN1_PWM
- FAN2_PWM

The pin programming options for these fan PWM outputs are shown in Table 78. The PWM output switching frequency can be programmed to one of four common fan switching frequency settings through register **fan_freq**, as shown in Table 79.

Table 78 FAN1_PWM, FAN2_PWM pin programming options

FAN1_PWM pin	FAN2_PWM pin	Register programming
PWM11	–	common.pwm11_func = 4
SYNC	–	common.sync_func = 4
–	FAULT1	common.fault1_func = 4
–	PWM7	common.pwm7_func = 4

Table 79 Fan PWM switching frequency programming

fan_freq	Fan F_{switch} (kHz)
0	25
1	50
2	100
3	200

XDPP1100 HW provides two methods for controlling the PWM output duty cycle:

- Duty-cycle mode
- Current mode

Register **fan_mode** selects between these two operational modes, as shown in Table 80. It should be noted that additional control modes such as temperature-based duty-cycle control and RPM control are possible through a user-written FW patch.

Table 80 Fan PWM operating mode selection

fan_mode	Fan operating mode
0	Duty-cycle mode
1	Current mode

13.1.1 Duty-cycle mode

In duty-cycle mode, the user directly programs the PWM duty cycle through one of the following methods:

- Register **fan_duty** programs the duty-cycle output in fractional duty cycle
  - fan_duty[7:0] = 0 ➔ 0/256 = 0.0 percent
  - fan_duty[7:0] = 1 ➔ 1/256 = 0.390625 percent
  - fan_duty[7:0] = 254 ➔ 254/256 = 99.21875 percent
  - fan_duty[7:0] = 255 ➔ overrides to 100 percent
- PMBus command FAN_COMMAND_x programs the duty-cycle output in fractional duty cycle of FANx, where X = 1, 2
  - FAN_COMMAND_x = C000h ➔ 0/256 = 0.0 percent
  - FAN_COMMAND_x = C001h ➔ 1/256 = 0.390625 percent
  - FAN_COMMAND_x = C0FEh ➔ 254/256 = 99.21875 percent
  - FAN_COMMAND_x = C0FFh ➔ overrides to 100 percent

FAN_CONFIG_1_2[6] = 0 is required for FAN1 and FAN_CONFIG_1_2[6] = 0 is required for FAN2.

13.1.2 Current mode

In current mode, the PWM duty cycle is computed as the ratio of the output current to the maximum fan current defined by register **fan_imax**. The following registers:

- **fan_imin**
- **fan_duty_min**

are provided to give additional control of the current to duty-cycle response as shown in Figure 96 and Figure 97. It should be noted that one or both of these registers may be set to 0.

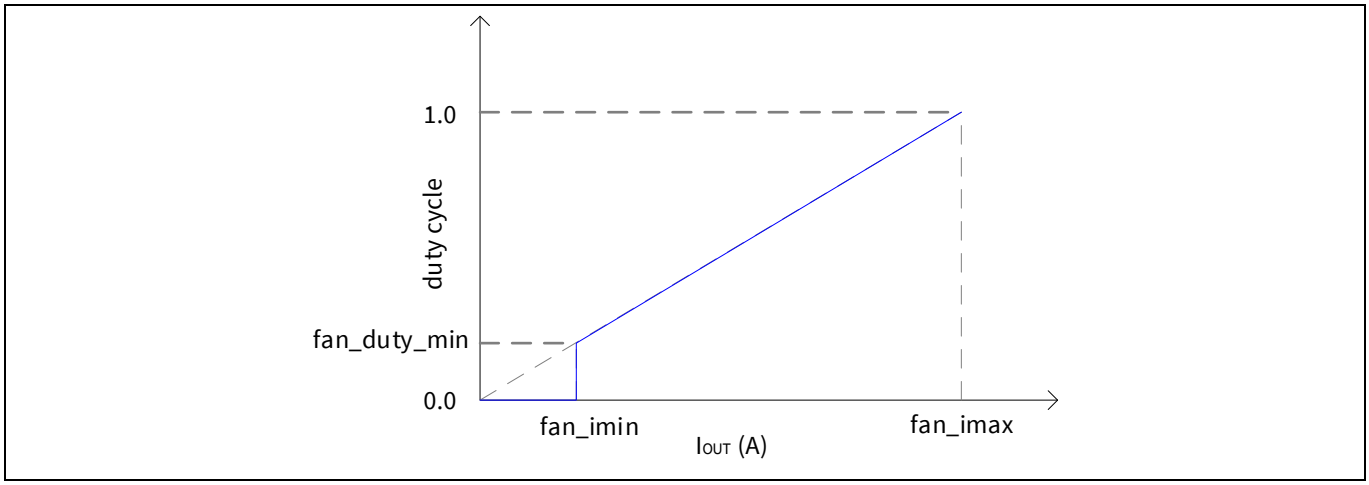

Figure 96 Current mode duty cycle with fan_duty_min less than or equal to (fan_imin/fan_imax)

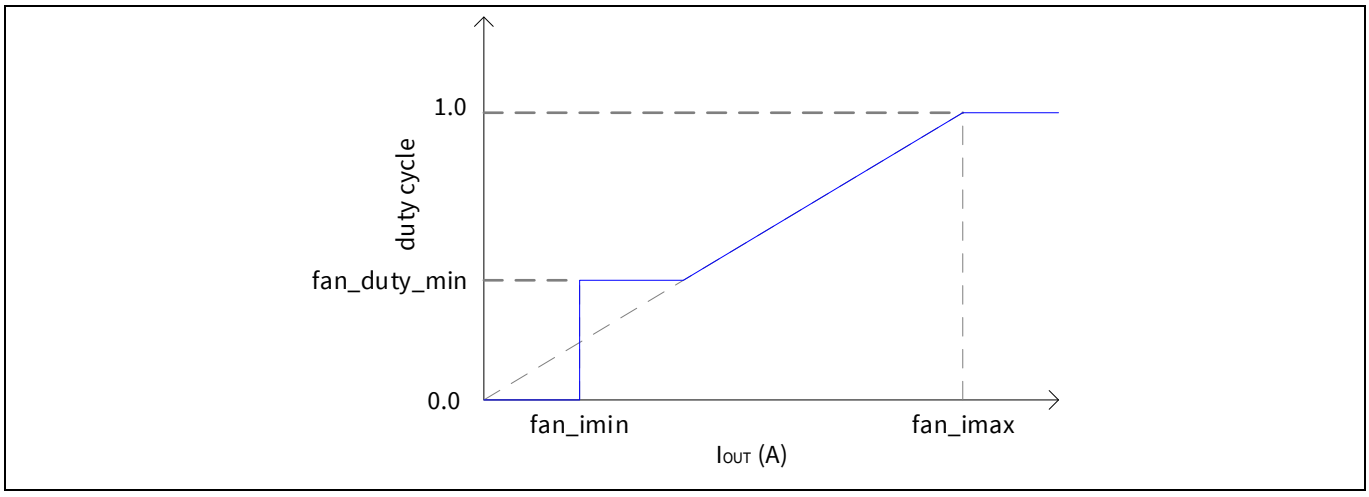

Figure 97 Current mode duty cycle with fan_duty_min greater than (fan_imin/fan_imax)

13.2 Fan speed input

XDPP1100 provides HW support for up to two fan speed tachometer inputs, identified as:

- FAN1_TACH
- FAN2_TACH

The pin programming options for these fan tachometer inputs are shown in Table 81.

Table 81 FAN1_TACH, FAN2_TACH pin programming options

FAN1_TACH pin	FAN2_TACH pin	Register programming
IMON	–	common.imon_func = 4
PWM12	–	common.pwm12_func = 4
–	FAULT2	common.fault2_func = 4
–	PWM8	common.pwm8_func = 4

For fan speed input, the following fan features are supported:

- Fan speeds in the range 1K to 25K RPM, where the speed in RPM is computed by counting incoming TACH pulses
- Tachometer pulses per revolution between 1 and 4

The tachometer pulse per revolution programming can be performed through:

- Register **fan_tach_ppr** as shown in Table 82
- PMBus command FAN_CONFIG_1_2[5:4] for FAN1 and FAN_CONFIG_1_2[1:0] for FAN2

Table 82 Fan tachometer input pulse per revolution (PPR) programming

fan_tach_ppr	Fan tachometer PPR
0	1
1	2
2	3
3	4

Fan speed is reported on:

- Register **fan_speed** with range 0 to 32K RPM and 8 RPM resolution
- PMBus commands:
  - READ_FAN_SPEED_1
  - READ_FAN_SPEED_2

These PMBus commands are in LINEAR11 format with exponent defined by PMBus command FW_CONFIG_TELEMETRY.READ_FAN_EXP. The recommended exponents are based on expected maximum speed range:

- EXP = 3d for speed range 0 to 8K RPM, 8 RPM resolution
- EXP = 4d for speed range 0 to 16K RPM, 16 RPM resolution
- EXP = 5d for speed range 0 to 32K RPM, 32 RPM resolution

13.3 Fan registers

The relevant fan-related registers and their descriptions are provided in Table 83.

Table 83 Fan-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
fan	fan_imax	RW	7000_4400h (Fan 1) 7000_4800h (Fan 2)	[7:0]	Fan maximum current reference when operating in current mode. When the output current exceeds fan_imax the output duty cycle will be 100 percent. See fan_imin for the current mode current to duty-cycle equation. LSB = 1 A, range = 0 to 255 A
fan	fan_imin	RW	7000_4400h (Fan 1) 7000_4800h (Fan 2)	[15:8]	Fan minimum current reference when operating in current mode. When the output current is less than fan_imin the output duty cycle will be zero. When the output current exceeds fan_imin the output duty cycle is given by the equation Duty(%) = 100 * min(1.0, max(fan_duty_min(U0.8), I_{OUT}(A) / fan_imax(A))) LSB = 1 A, range = 0 to 255 A
fan	fan_duty_min	RW	7000_4400h (Fan 1) 7000_4800h (Fan 2)	[23:16]	Fan minimum duty cycle when operating in current mode. See fan_imin for the current mode current to duty-cycle equation. LSB = 1/256, range = 0 to 255/256
fan	fan_freq	RW	7000_4400h (Fan 1) 7000_4800h (Fan 2)	[25:24]	Fan switching frequency. 0: 25 kHz 1: 50 kHz

Peripheral	Field name	Access	Address	Bits	Description
					2: 100 kHz 3: 200 kHz
fan	fan_tach_ppr	RW	7000_4400h (Fan 1) 7000_4800h (Fan 2)	[27:26]	Tachometer pulses per revolution (ppr). 0: 1 ppr 1: 2 ppr 2: 3 ppr 3: 4 ppr
fan	fan_mode	RW	7000_4400h (Fan 1) 7000_4800h (Fan 2)	[28]	Fan operating mode. 0: Duty-cycle mode 1: Current mode
fan	fan_iout_sel	RW	7000_4400h (Fan 1) 7000_4800h (Fan 2)	[29]	Fan I_{OUT} source when operating in current mode. 0: Loop 0 I_{OUT} 1: Loop 1 I_{OUT}
fan	fan_duty	RW	7000_4404h (Fan 1) 7000_4804h (Fan 2)	[7:0]	Fan-commanded duty cycle when operating in duty-cycle mode. Note fan_duty = 0xFF overrides to 100 percent. **Computed by FW from PMBus command as follows:** fan_duty[7:0] = FAN_COMMAND[10:0] * 2^(FAN_COMMAND[15:11]+8) LSB = 1/256, range = 0 to 255/256

fan	fan_speed	R	7000_4408h (Fan 1)
			7000_4808h (Fan 2)
				[11:0]	Fan speed in RPM as measured at the fan_tach_i input.
					PMBus command computed by FW as follows:
					if (fan_speed[11])
					READ_FAN_SPEED = {6'd10,fan_speed[11:2]}
					else if (fan_speed[10])
					READ_FAN_SPEED = {6'd8,fan_speed[10:1]}
					else
					READ_FAN_SPEED = {6'd6,fan_speed[9:0]}
					LSB = 8 RPM, range = 0 to 32760 RPM

13.4 Fan PMBus commands

Table 84 Fan PMBus commands

Peripheral	Field name	Access	Address	Bits	Description
FAN_CONFIG_1_2	RW	Byte	3Ah	[7]	FAN1 enable. 0: FAN1 disabled 1: FAN1 enabled
				[6]	FAN1 control mode select. Not supported by default FW patch. Set to 0 for correct functioning of FAN_COMMAND_1 when in duty-cycle mode as selected by register fan_mode.
				[5:4]	FAN1 tachometer pulses per revolution (ppr). 0: 1 ppr 1: 2 ppr 2: 3 ppr 3: 4 ppr
				[3]	FAN2 enable. 0: FAN2 disabled 1: FAN2 enabled
				[2]	FAN2 control mode select. Not supported by default FW patch. Set to 0 for correct functioning of FAN_COMMAND_2 when in duty-cycle mode as selected by register fan_mode.
				[1:0]	FAN2 tachometer pulses per revolution (ppr). 0: 1 ppr 1: 2 ppr 2: 3 ppr 3: 4 ppr
FAN_COMMAND_1	RW	Word	3Bh	[15:0]	FAN1 fractional duty cycle when in duty-cycle mode as selected by register fan_mode. Format is LINEAR11 with exponent of -8 recommended. C000h = 0/256 = 0.0 percent C001h = 1/256 = 0.390625 percent C002h = 2/256 = 0.78125 percent … C0FEh = 254/256 = 99.21875 percent C0FFh = 256/256 = 100 percent (note FFh case overrides to 100 percent duty)
FAN_COMMAND_2	RW	Word	3Ch	[15:0]	FAN2 fractional duty cycle when in duty-cycle mode as selected by register fan_mode. Format is LINEAR11 with

Peripheral	Field name	Access	Address	Bits	Description
					exponent of -8 recommended. C000h = 0/256 = 0.0 percent C001h = 1/256 = 0.390625 percent C002h = 2/256 = 0.78125 percent … C0FEh = 254/256 = 99.21875 percent C0FFh = 256/256 = 100 percent (note FFh case overrides to 100 percent duty)
READ_FAN_SPEED_1	R	Word	90h	[15:0]	Returns the measured FAN1 speed in RPM in the LINEAR11 format with exponent defined by FW_CONFIG_TELEMETRY.READ_FAN_EXP.
READ_FAN_SPEED_2	R	Word	91h	[15:0]	Returns the measured FAN2 speed in RPM in the LINEAR11 format with exponent defined by FW_CONFIG_TELEMETRY.READ_FAN_EXP.
FW_CONFIG_TELEMETRY	RW	Block 21 bytes	C6h	[107:104]	READ_FAN_EXP: Defines the LINEAR11 exponent for commands READ_FAN_SPEED_1 and READ_FAN_SPEED_2.

14 IO muxing

This chapter describes the programming of the multiple digital IO pins and describes the relevant register settings in more detail.

XDPP1100 contains several programmable multipurpose digital IO pins, and depending on the variant the pin numbers are:

- 21, for the Q040 variant
- 11, for the Q024 variant

The IO muxing module is responsible for the programming of these digital IO pins.

14.1 Multipurpose digital IO muxing

The multipurpose digital IO pins are programmed through registers. These registers are named as **<pin name>_func**, for example:

- **en_func**
- **fault1_func**
- **pwm3_func**, etc.

Table 85 shows the programming options for the multipurpose digital IO pins, showing the differences in pin numbering for the variants XDPP1100-Q040 and XDPP1100-Q024. The programming is in general as follows:

- **<pin name>_func** = 0 selects the function matching the pin name. The exception is the SYNC pin, which has no assigned function for sync_func = 0
- **<pin name>_func** = 1 or 2 selects a bit from one of the CPU's two GPIO buses, GPIO0 and GPIO1
- **<pin name>_func** = 3 assigns a digital IO pin for the external sync function (Section 7.1.2). All digital IO pins, except SDA and SCL, can be used. The SDA and SCA are permanently assigned to the I^2C/PMBus port
- **<pin name>_func** = 4 and above allow the selection of various other functions, such as fan PWM output and tach input (Section 13), UART TX output and RX input (Section 15), secondary I^2C port SCL2 input and SDA2 IO (Section 15)

Table 85 Multipurpose IO pin function programming

			<pin_name>_func[2:0]							
Pin name	**XDPP1100-Q040 pin number**	**XDPP1100-Q024 pin number**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
FAULT1	1	N/A	FAULT1/GPIO0[2]	GPIO0[2]	GPIO1[2]	External sync	FAN2_PWM	SDA2	UART_RX	N/A
FAULT2	4	N/A	FAULT2/GPIO1[2]	GPIO0[2]	GPIO1[2]	External sync	FAN2_TACH	SCL2	UART_TX	N/A
IMON	6	4	IMON	GPIO0[3]	GPIO1[3]	External sync	FAN1_TACH	N/A	N/A	N/A
PWM11	21	N/A	PWM11	GPIO0[6]	GPIO1[6]	External sync	FAN1_PWM	N/A	N/A	N/A
PWM12	22	N/A	PWM12	GPIO0[7]	GPIO1[7]	External sync	FAN1_TACH	N/A	N/A	N/A
PWM1	23	13	PWM1	GPIO0[5]	GPIO1[5]	External sync	N/A	N/A	N/A	N/A
NPWM2	24	14	PWM2	GPIO0[7]	GPIO1[7]	External sync	N/A	N/A	N/A	N/A
PWM3	25	15	PWM3	GPIO0[1]	GPIO1[1]	External sync	N/A	N/A	N/A	N/A
PWM4	26	16	PWM4	GPIO0[2]	GPIO1[2]	External sync	N/A	N/A	N/A	N/A
PWM5	27	17	PWM5	GPIO0[3]	GPIO1[3]	External sync	UART_RX	N/A	N/A	N/A
PWM6	28	18	PWM6	GPIO0[4]	GPIO1[4]	External sync	UART_TX	N/A	N/A	N/A
PWM9	29	N/A	PWM9	GPIO0[4]	GPIO1[4]	External sync	N/A	N/A	N/A	N/A
PWM10	30	N/A	PWM10	GPIO0[5]	GPIO1[5]	External sync	N/A	N/A	N/A	N/A
PWM7	31	N/A	PWM7	GPIO0[5]	GPIO1[5]	External sync	FAN2_PWM	N/A	N/A	N/A
PWM8	32	N/A	PWM8	GPIO0[6]	GPIO1[6]	External sync	FAN2_TACH	N/A	N/A	N/A
SMBALERT#	35	21	SMBALERT#	GPIO0[6]	GPIO1[6]	External sync	GPIO0[7]	GPIO1[7]	N/A	N/A
BEN	36	N/A	BEN/GPIO1[0]	GPIO0[0]	GPIO1[0]	External sync	UART_RX	SDA2	N/A	N/A
BPWRGD	37	N/A	BPWRGD/GPIO1[1]	GPIO0[1]	GPIO1[1]	External sync	UART_TX	SCL2	N/A	N/A
SYNC	38	22	N/A	GPIO0[7]	GPIO1[7]	External sync	FAN1_PWM	N/A	N/A	N/A
EN	39	23	EN/GPIO0[0]	GPIO0[0]	GPIO1[0]	External sync	N/A	N/A	N/A	N/A
PWRGD	40	24	PWRGD/GPIO0[1]	GPIO0[1]	GPIO1[1]	External sync	N/A	N/A	N/A	N/A

14.2 Digital input priority

Some functions on the XDPP1100 can be independently mapped to multiple IO pins. For the output functions this is generally not problematic, but for the input functions a priority must be assigned so that only one IO pin provides the actual input. The XDPP1100 input prioritization is listed in Table 86 for all multi-sourced input functions.

Table 86 Input prioritizations for all multi-sourced input functions

Function	Highest priority IO pin	Lowest priority IO pin
External sync	FAULT1	FAULT2, IMON, PWM9, PWM10, PWM11, PWM12, PWM1, PWM2, PWM3, PWM4, PWM5, PWM6, PWM7, PWM8, SMBALERT#, BEN, BPWRGD, SYNC, EN, PWRGD
FAN1_TACH	IMON	PWM12
FAN2_TACH	FAULT2	PWM8
UART_RX	BEN	FAULT1, PWM5
SCL2	FAULT2	BPWRGD
SDA2	FAULT1	BEN
GPIO0[0]	BEN	EN
GPIO0[1]	PWM3	BPWRGD, PWRGD
GPIO0[2]	FAULT1	FAULT2, PWM4
GPIO0[3]	IMON	PWM5
GPIO0[4]	PWM9	PWM6
GPIO0[5]	PWM10	PWM1, PWM7
GPIO0[6]	PWM11	PWM8, SMBALERT#
GPIO0[7]	PWM12	PWM2, SMBALERT#, SYNC
GPIO1[0]	BEN	EN
GPIO1[1]	PWM3	BPWRGD, PWRGD
GPIO1[2]	FAULT1	FAULT2, PWM4
GPIO1[3]	IMON	PWM5
GPIO1[4]	PWM9	PWM6
GPIO1[5]	PWM10	PWM1, PWM7
GPIO1[6]	PWM11	PWM8, SMBALERT#
GPIO1[7]	PWM12	PWM2, SMBALERT#, SYNC

14.3 GPIO deglitch

The GPIO deglitch function only applies to GPIO0[0] and GPIO1[0] assigned to EN and BEN by default ROM FW. This is due to an erratum affecting all currently available XDPP1100 versions. Therefore, it should be noted that this section describes actual behavior rather than the intended behavior.

A deglitcher is provided on the GPIO0 and GPIO1 inputs. An individual GPIO input bit must be stable for the time defined by register **gpio_dly** for a transition to pass through the deglitcher. A setting of 0 on **gpio_dly** disables the deglitcher on all GPIO inputs.

Register **gpio0_dben** enables the deglitcher on GPIO0 and currently only the following settings are supported:

- 00h: GPIO0[0] deglitch disabled
- 01h: GPIO0[0] deglitch enabled

Register **gpio1_dben** enables the deglitcher on GPIO1 and currently only the following settings are supported:

- 00h: GPIO1[0] deglitch disabled
- 01h: GPIO1[0] deglitch enabled

14.4 Digital IO buffer programming

The digital IO buffer programming depends on the pins utilized. For the pins BEN, BPWRGD, EN, FAULT1, FAULT2, IMON, PWRGD, SYNC, PWM1, PWM2, PWM3, PWM4, PWM5, PWM6, PWM7, PWM8, PWM9, PWM10, PWM11, PWM12 the following registers can be used in addition to the registers described in Section 14.4:

- **<pin_name>_pd**, enables a weak pull-down resistor
- **<pin_name>_pu_n**, enables a weak pull-up resistor
- **<pin_name>_ppen**, selects between an open drain or CMOS output stage

The pins SCL, SDA and SMBALERT# have special IO buffers for the I^2C/PMBus interface. The output buffer is open drain only. The following registers can be used to program these pins:

- **<pin_name>_pd**, enables a weak pull-down resistor
- **<pin_name>_en_3v3**, selects between 1.8 V or 3.3 V CMOS levels on the input buffer

The pins PWM6 and PWM11 have special output buffers with tri-state bias resistors intended for use with integrated power stage drivers. For the programming, a register **<pin_name>_static_hiz** can be used to enable the tri-state bias resistors.

14.5 IO muxing registers

The relevant IO muxing-related registers and their descriptions are provided in Table 87.

Table 87 IO muxing-related register descriptions

Peripheral	Field name	Access	Address	Bits	Description
common	ben_func	RW	7000_3004h	[2:0]	Pin BEN function definition. 0: BEN (GPIO1[0]), digital IO 1: GPIO0[0], digital IO 2: GPIO1[0], digital IO 3: SYNC, digital IO 4: UARTRXD, digital Input 5: SDA2, digital IO 6 to 7: Not used
common	ben_pd	RW	7000_3004h	[3]	Pin BEN weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	ben_pu_n	RW	7000_3004h	[4]	Pin BEN weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled

Peripheral	Field name	Access	Address	Bits	Description
common	ben_ppen	RW	7000_3004h	[5]	Pin BEN output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	bpwrgd_func	RW	7000_3004h	[8:6]	Pin BPWRGD function definition. 0: BPWRGD (GPIO1[1]), digital IO 1: GPIO0[1], digital IO 2: GPIO1[1], digital IO 3: SYNC, digital IO 4: UARTTXD, digital output 5: SCL2, digital IO 6 to 7: Not used
common	bpwrgd_pd	RW	7000_3004h	[9]	Pin BPWRGD weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	bpwrgd_pu_n	RW	7000_3004h	[10]	Pin BPWRGD weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	bpwrgd_ppen	RW	7000_3004h	[11]	Pin BPWRGD output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	en_func	RW	7000_3004h	[14:12]	Pin EN function definition. 0: EN (GPIO0[0]), digital IO 1: GPIO0[0], digital IO 2: GPIO1[0], digital IO 3: SYNC, digital IO 4 to 7: Not used
common	en_pd	RW	7000_3004h	[15]	Pin EN weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	en_pu_n	RW	7000_3004h	[16]	Pin EN weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	en_ppen	RW	7000_3004h	[17]	Pin EN output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	fault1_func	RW	7000_3004h	[20:18]	Pin FAULT1 function definition. 0: FAULT1 (GPIO0[2]), digital IO 1: GPIO0[2], digital IO 2: GPIO1[2], digital IO 3: SYNC, digital IO 4: FAN2_PWM, digital output 5: SDA2, digital IO

Peripheral	Field name	Access	Address	Bits	Description
					6: UARTRXD, digital input 7: Not used
common	fault1_pd	RW	7000_3004h	[21]	Pin FAULT1 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	fault1_pu_n	RW	7000_3004h	[22]	Pin FAULT1 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	fault1_ppen	RW	7000_3004h	[23]	Pin FAULT1 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	fault2_func	RW	7000_3004h	[26:24]	Pin FAULT2 function definition. 0: FAULT2 (GPIO1[2]), digital IO 1: GPIO0[2], digital IO 2: GPIO1[2], digital IO 3: SYNC, digital IO 4: FAN2_TACH, digital input 5: SCL2, digital IO 6: UARTTXD, digital output 7: Not used
common	fault2_pd	RW	7000_3004h	[27]	Pin FAULT2 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	fault2_pu_n	RW	7000_3004h	[28]	Pin FAULT2 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	fault2_ppen	RW	7000_3004h	[29]	Pin FAULT2 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	imon_func	RW	7000_3008h	[2:0]	Pin IMON function definition. 0: IMON, analog 1: GPIO0[3], digital IO 2: GPIO1[3], digital IO 3: SYNC, digital IO 4: FAN1_TACH, digital input 5 to 7: Not used
common	imon_pd	RW	7000_3008h	[3]	Pin IMON weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	imon_pu_n	RW	7000_3008h	[4]	Pin IMON weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	imon_ppen	RW	7000_3008h	[5]	Pin IMON output buffer CMOS/open drain select.

Peripheral	Field name	Access	Address	Bits	Description
					0: Open drain output 1: CMOS output
common	pwrgd_func	RW	7000_3008h	[8:6]	Pin PWRGD function definition. 0: PWRGD (GPIO0[1]), digital IO 1: GPIO0[1], digital IO 2: GPIO1[1], digital IO 3: SYNC, digital IO 4 to 7: Not used
common	pwrgd_pd	RW	7000_3008h	[9]	Pin PWRGD weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwrgd_pu_n	RW	7000_3008h	[10]	Pin PWRGD weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwrgd_ppen	RW	7000_3008h	[11]	Pin PWRGD output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	smbalert_func	RW	7000_3008h	[14:12]	Pin SMBALERT function definition. 0: SMBALERT, digital IO 1: GPIO0[6], digital IO 2: GPIO1[6], digital IO 3: SYNC, digital IO 4: GPIO0[7], digital IO 5: GPIO1[7], digital IO 6 to 7: Not used
common	smbalert_pd	RW	7000_3008h	[15]	Pin SMBALERT weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	smbalert_pu_n	RW	7000_3008h	[16]	Function not in use.
common	smbalert_en_3v3	RW	7000_3008h	[17]	Pin SMBALERT 1.8 V/3.3 V input threshold select. 0: 1.8 V CMOS input 1: 3.3 V CMOS input
common	sync_func	RW	7000_3008h	[20:18]	Pin SYNC function definition. 0: Not used 1: GPIO0[7], digital IO 2: GPIO1[7], digital IO 3: SYNC, digital IO 4: FAN1_PWM, digital output 5 to 7: Not used
common	sync_pd	RW	7000_3008h	[21]	Pin SYNC weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled

Peripheral	Field name	Access	Address	Bits	Description
common	sync_pu_n	RW	7000_3008h	[22]	Pin SYNC weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	sync_ppen	RW	7000_3008h	[23]	Pin SYNC output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	scl_pd	RW	7000_3008h	[24]	Pin SCL weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	scl_en_3v3	RW	7000_3008h	[25]	Pin SCL 1.8 V/3.3 V input threshold select. 0: 1.8 V CMOS input 1: 3.3 V CMOS input
common	sda_pd	RW	7000_3008h	[26]	Pin SDA weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	sda_en_3v3	RW	7000_3008h	[27]	Pin SDA 1.8 V/3.3 V input threshold select. 0: 1.8 V CMOS input 1: 3.3 V CMOS input
common	sync_deglitch_en	RW	7000_3008h	[28]	Deglitch enable for digital sync function when used as an input. 0: Sync deglitch disabled 1: Sync deglitch enabled
common	pwm1_func	RW	7000_300Ch	[2:0]	Pin PWM1 function definition. 0: PWM1, digital output 1: GPIO0[5], digital IO 2: GPIO1[5], digital IO 3: SYNC, digital IO 4 to 7: Not used
common	pwm1_pd	RW	7000_300Ch	[3]	Pin PWM1 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm1_pu_n	RW	7000_300Ch	[4]	Pin PWM1 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm1_ppen	RW	7000_300Ch	[5]	Pin PWM1 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm2_func	RW	7000_300Ch	[8:6]	Pin PWM2 function definition. 0: PWM2, digital output 1: GPIO0[7], digital IO 2: GPIO1[7], digital IO

Peripheral	Field name	Access	Address	Bits	Description
					3: SYNC, digital IO 4 to 7: Not used
common	pwm2_pd	RW	7000_300Ch	[9]	Pin PWM2 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm2_pu_n	RW	7000_300Ch	[10]	Pin PWM2 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm2_ppen	RW	7000_300Ch	[11]	Pin PWM2 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm3_func	RW	7000_300Ch	[14:12]	Pin PWM3 function definition. 0: PWM3, digital output 1: GPIO0[1], digital IO 2: GPIO1[1], digital IO 3: SYNC, digital IO 4 to 7: Not used
common	pwm3_pd	RW	7000_300Ch	[15]	Pin PWM3 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm3_pu_n	RW	7000_300Ch	[16]	Pin PWM3 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm3_ppen	RW	7000_300Ch	[17]	Pin PWM3 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm4_func	RW	7000_300Ch	[20:18]	Pin PWM4 function definition. 0: PWM4, digital output 1: GPIO0[2], digital IO 2: GPIO1[2], digital IO 3: SYNC, digital IO 4 to 7: Not used
common	pwm4_pd	RW	7000_300Ch	[21]	Pin PWM4 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm4_pu_n	RW	7000_300Ch	[22]	Pin PWM4 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm4_ppen	RW	7000_300Ch	[23]	Pin PWM4 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output

Peripheral	Field name	Access	Address	Bits	Description
common	pwm5_func	RW	7000_300Ch	[26:24]	Pin PWM5 function definition. 0: PWM5, digital output 1: GPIO0[3], digital IO 2: GPIO1[3], digital IO 3: SYNC, digital IO 4: UARTRXD, digital input 5 to 7: Not used
common	pwm5_pd	RW	7000_300Ch	[27]	Pin PWM5 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm5_pu_n	RW	7000_300Ch	[28]	Pin PWM5 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm5_ppen	RW	7000_300Ch	[29]	Pin PWM5 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm6_func	RW	7000_3010h	[2:0]	Pin PWM6 function definition. 0: PWM6, digital output 1: GPIO0[4], digital IO 2: GPIO1[4], digital IO 3: SYNC, digital IO 4: UARTTXD, digital output 5 to 7: Not used
common	pwm6_pd	RW	7000_3010h	[3]	Pin PWM6 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm6_pu_n	RW	7000_3010h	[4]	Pin PWM6 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm6_ppen	RW	7000_3010h	[5]	Pin PWM6 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm7_func	RW	7000_3010h	[8:6]	Pin PWM7 function definition. 0: PWM7, digital output 1: GPIO0[5], digital IO 2: GPIO1[5], digital IO 3: SYNC, digital IO 4: FAN2_PWM, digital output 5 to 7: Not used
common	pwm7_pd	RW	7000_3010h	[9]	Pin PWM7 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled

Peripheral	Field name	Access	Address	Bits	Description
common	pwm7_pu_n	RW	7000_3010h	[10]	Pin PWM7 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm7_ppen	RW	7000_3010h	[11]	Pin PWM7 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm8_func	RW	7000_3010h	[14:12]	Pin PWM8 function definition. 0: PWM8, digital output 1: GPIO0[6], digital IO 2: GPIO1[6], digital IO 3: SYNC, digital IO 4: FAN2_TACH, digital input 5 to 7: Not used
common	pwm8_pd	RW	7000_3010h	[15]	Pin PWM8 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm8_pu_n	RW	7000_3010h	[16]	Pin PWM8 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm8_ppen	RW	7000_3010h	[17]	Pin PWM8 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm9_func	RW	7000_3010h	[20:18]	Pin PWM9 function definition. 0: PWM9, digital output 1: GPIO0[4], digital IO 2: GPIO1[4], digital IO 3: SYNC, digital IO 4 to 7: Not used
common	pwm9_pd	RW	7000_3010h	[21]	Pin PWM9 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm9_pu_n	RW	7000_3010h	[22]	Pin PWM9 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm9_ppen	RW	7000_3010h	[23]	Pin PWM9 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm10_func	RW	7000_3010h	[26:24]	Pin PWM10 function definition. 0: PWM10, digital output 1: GPIO0[5], digital IO 2: GPIO1[5], digital IO 3: SYNC, digital IO 4 to 7: Not used

Peripheral	Field name	Access	Address	Bits	Description
common	pwm10_pd	RW	7000_3010h	[27]	Pin PWM10 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm10_pu_n	RW	7000_3010h	[28]	Pin PWM10 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm10_ppen	RW	7000_3010h	[29]	Pin PWM10 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm6_static_hiz	RW	7000_3010h	[30]	Pin PWM6 static HIZ control. Pin PWM6 has a special output buffer with tri-state bias resistors for use with integrated power stage drivers. 0: Tri-state biasing disabled (typical usage) 1: Tri-state biasing enabled (integrated power stage usage)
common	force_pwm56_in_en_n	RW	7000_3010h	[31]	For test use only. Set to 1 for normal operation. 0: Force-enable the input buffers on PWM5, PWM6 (test only) 1: PWM5, PWM6 input buffer enable controlled by functional logic (normal operation)
common	pwm11_func	RW	7000_3014h	[2:0]	Pin PWM11 function definition. 0: PWM11, digital output 1: GPIO0[6], digital IO 2: GPIO1[6], digital IO 3: SYNC, digital IO 4: FAN1_PWM, digital output 5 to 7: Not used
common	pwm11_pd	RW	7000_3014h	[3]	Pin PWM11 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm11_pu_n	RW	7000_3014h	[4]	Pin PWM11 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm11_ppen	RW	7000_3014h	[5]	Pin PWM11 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm12_func	RW	7000_3014h	[8:6]	Pin PWM12 function definition. 0: PWM12, digital output 1: GPIO0[7], digital IO 2: GPIO1[7], digital IO

Peripheral	Field name	Access	Address	Bits	Description
					3: SYNC, digital IO 4: FAN1_TACH, digital input 5 to 7: Not used
common	pwm12_pd	RW	7000_3014h	[9]	Pin PWM12 weak pull-down enable. 0: Pull-down disabled 1: Pull-down enabled
common	pwm12_pu_n	RW	7000_3014h	[10]	Pin PWM12 weak pull-up enable. 0: Pull-up enabled 1: Pull-up disabled
common	pwm12_ppen	RW	7000_3014h	[11]	Pin PWM12 output buffer CMOS/open drain select. 0: Open drain output 1: CMOS output
common	pwm11_static_hiz	RW	7000_3014h	[12]	Pin PWM11 static HIZ control. Pin PWM11 has a special output buffer with tri-state bias resistors for use with integrated power stage drivers. 0: Tri-state biasing disabled (typical usage) 1: Tri-state biasing enabled (integrated power stage usage)
common	sync_dir_out	RW	7000_3014h	[13]	Defines direction of pin mapped to SYNC function. 0: SYNC mapped pin is input 1: SYNC mapped pin is output
common	gpio_dly	RW	7000_3014h	[22:20]	Defines deglitch time on GPIO input buffers. The GPIO input must be stable for the defined time (clocked at 25 MHz) to be passed through the deglitcher. A setting of 0 disables the deglitch function. Due to an erratum, this register applies to GPIO0[0] and GPIO1[0] (typically EN and BEN) only. LSB = 1 µs, range = 2 to 8 µs
common	gpio0_dben	RW	7000_3028h	[23:16]	GPIO0 bus input deglitch enable. Due to an erratum, only bit [0] is currently functional. 00h: GPIO0[0] deglitch disabled 01h: GPIO0[0] deglitch enabled 02-FFh: Not allowed
common	gpio1_dben	RW	7000_3028h	[31:24]	GPIO1 bus input deglitch enable. Due to an erratum, only bit [0] is currently functional. 00h: GPIO1[0] deglitch disabled 01h: GPIO1[0] deglitch enabled 02-FFh: Not allowed

15 Central processing unit subsystem

This chapter describes the CPU subsystem (CPUS) and describes the relevant register settings in more detail.

The XDPP1100 CPUS block and bus interconnections diagram is shown in Figure 98.

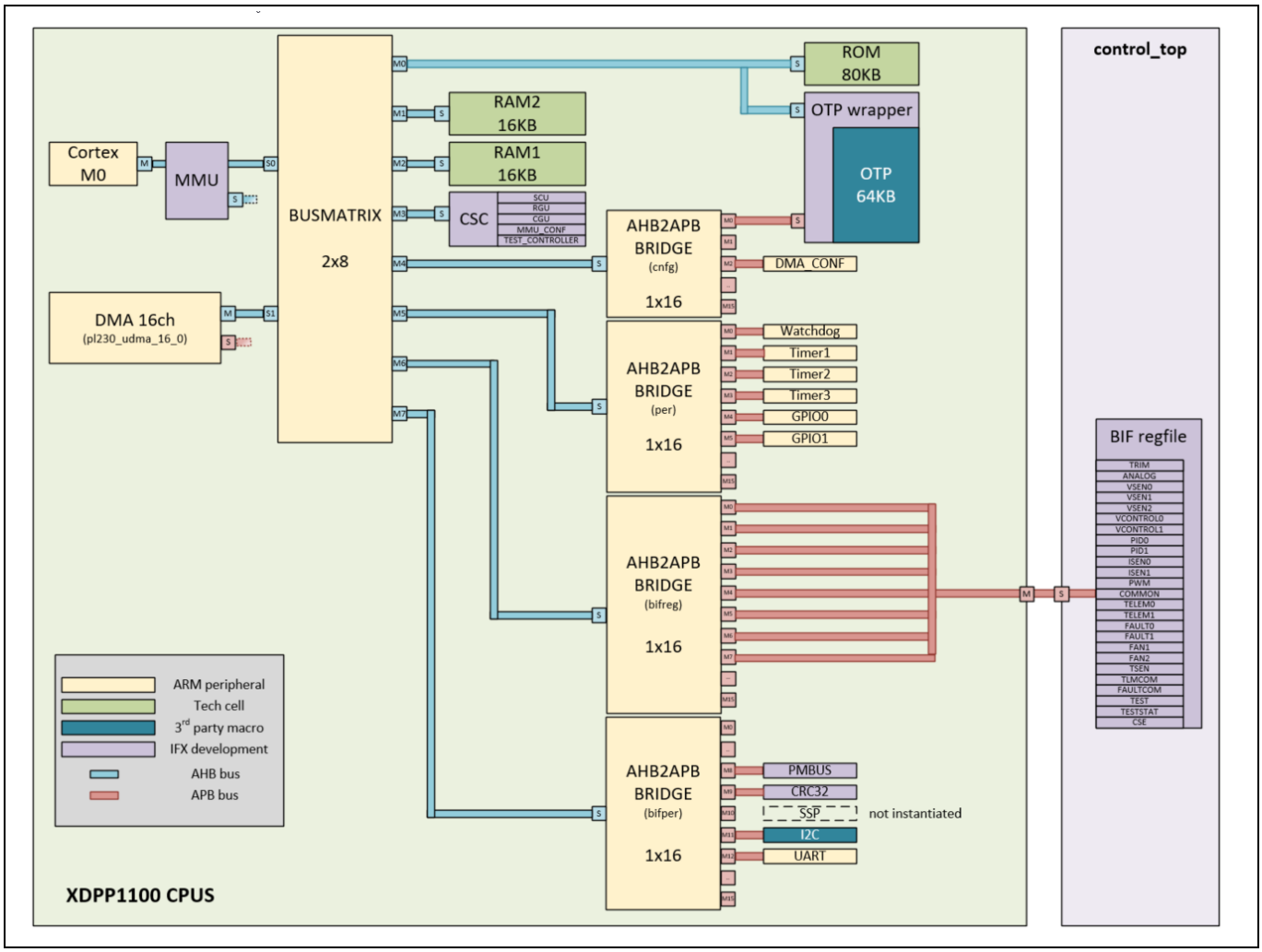

Figure 98 CPUS block diagram

XDPP1100 CPUS is based on a multilayer Arm® AMBA® bus protocol, in which two AHB masters (Cortex®-M0 and DMA) access all peripherals through an Arm® bus matrix. The main features of the bus matrix are:

- It allows concurrent access when the target peripherals are different
- It provides arbitration (round-robin)
- It implements a default slave for out-of-memory accesses
- It is silicon proven on thousands of devices (Arm® IP)

The CPUS uses an AMBA® AHB bus protocol for data access on memories (RAM, ROM, and OTP) and an AMBA® advanced peripheral bus (APB) protocol for peripheral register configuration.

The CPU is an Arm® Cortex®-M0, which executes the application code from ROM/OTP with FW variables in RAM.

A DMA is available to handle autonomous data transfers, and to avoid BW losses on the microcontroller in case of data movements.

AHB to APB Arm® bridges handle the protocol conversion between the two AMBA® layers.

The CPUS includes a memory management unit (MMU) that can remap memory addresses based on a configurable scheme, allowing the implementation of a code-patching mechanism.

15.1 CPUS bus matrix

The CPUS bus matrix implements decoding and arbitration to route all master requests to slaves. The bus matrix has two AHB slave ports, to which Cortex®-M0 and DMA AHB master ports are connected, and eight AHB master ports, to which all the AHB slave peripherals are connected. Master–slave interconnections are shown in Table 88; while the Cortex®-M0 can access all the peripherals, the DMA has some limitations for application stability purposes.

Table 88 Bus matrix internal interconnection

Master port	Peripheral	S0: Cortex®-M0	S1: DMA
M0	ROM/OTP	Yes	Yes
M1	RAM2	Yes	Yes
M2	RAM1	Yes	Yes
M3	CSC	Yes	No
M4	OTP CONF, DMA CONF	Yes	No
M5	WDT, DTIMER1/2/3, GPIO0/1	Yes	No
M6	BIF REGFILE (CONTROL)	Yes	Yes
M7	PMBus/CRC/I^2C	Yes	Yes

15.2 Cortex®-M0 CPU

The Cortex®-M0 processor is a 32-bit reduced instruction set computing (RISC) processor with a von Neumann architecture (single-bus interface).

It uses an instruction set defined in the ARMv6 architecture with Thumb and Thumb-2 support. Thumb-2 technology extended the previous Thumb instruction set to allow all operations to be carried out in one CPU state. The instruction set in Thumb-2 included both 16-bit and 32-bit instructions; most instructions generated by the C compiler use the 16-bit instructions, and the 32-bit instructions are used when the 16-bit version cannot carry out the required operations. This results in high code density and avoids the overhead of switching between two instruction sets. In total, the Cortex®-M0 processor supports 56 base instructions. The Cortex®-M0 processor is highly capable because the Thumb instruction set is highly optimized.

Academically, the Cortex®-M0 processor is classified as load-store architecture, as it has separate instructions for reading and writing to memory, and instructions for arithmetic or logical operations that use registers.

The specifications of the Cortex®-M0 are outlined in a number of Arm® documents. The Cortex®-M0 Devices Generic User Guide covers the programmer's model, instruction set and general information about the architecture. The full details of the instruction set, programmer's model and other topics are specified in a document called the ARMv6-M Architecture Reference Manual. The timing information of the processor core, and implementation-related information are described in a document called the Cortex®-M0 Architecture Reference Manual (TRM). These documents are available from the Arm® website (www.arm.com). Note that downloading the ARMv6-M Architecture Reference Manual requires a registration process.

A simplified block diagram of the Cortex®-M0 is shown in Figure 99.

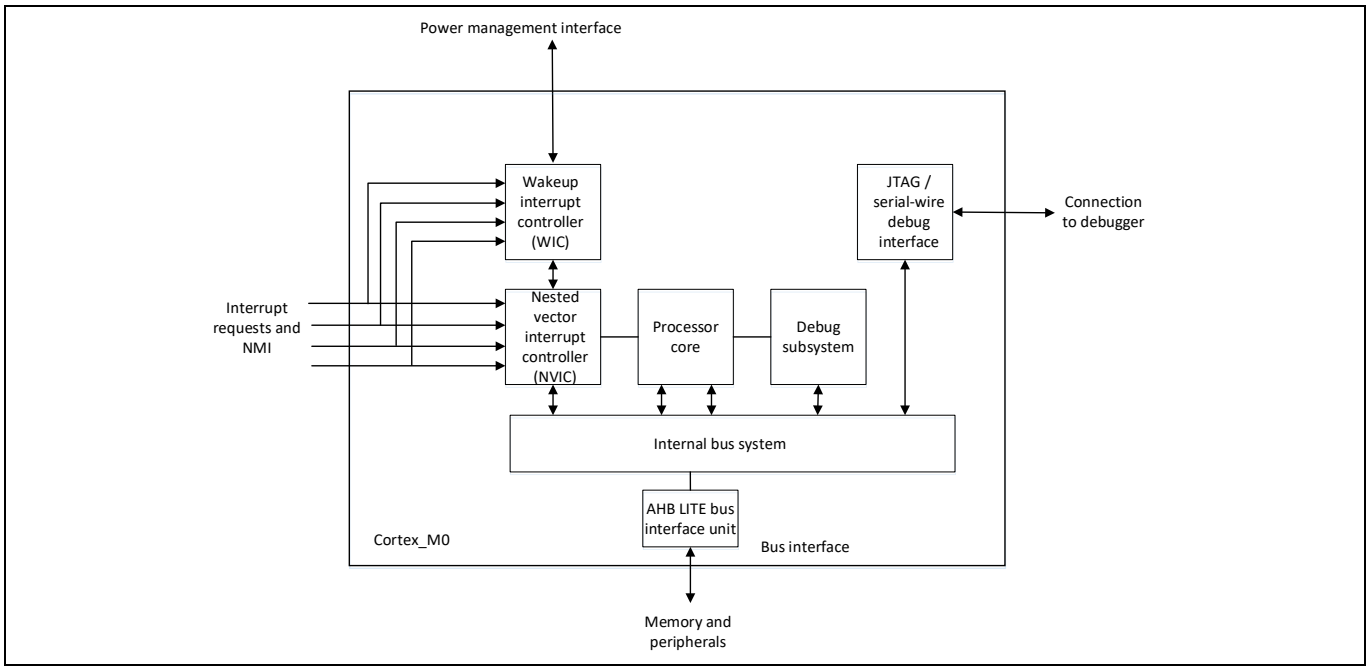

Figure 99 Simplified block diagram of the Cortex®-M0 processor

15.2.1 CPU interrupt sources

The Cortex®-M0 module processor embeds an Arm® Nested Vector Interrupt Controller (NVIC) supporting up to 32 interrupt sources and 1 not-maskable interrupt (NMI) source, with each one having a unique exception number and a dedicated memory address input.
The NVIC interrupt sources comes from several CPUS resources:

- DMA controller
- D_{TIMER} (1, 2, 3)
- WDT
- GPIOs (0, 1)
- PMBus
- I^2C
- UART
- OTP module
- Control module

The following interrupts table (Table 89) is supported by Arm® Cortex®-M0. All interrupt sources are considered to be synchronized with the Arm® clock domain, in particular if some of those are pulsed, then the pulse should be larger than four Arm® clock domain cycles. All interrupt sources need to be enabled in the NVIC before being used, with the exception of the NMI.

Interrupts generated outside the CPUS module are defined as external.

Table 89 Cortex®-M0 interrupt table

Interrupt source	Description	External index	CMSYS interrupt number	Memory address
RESET	Reset	–	–	0000_0004h
NMI	NMI	–	-14	0000_0008h
HARDFAULT	Hard fault	–	-13	0000_000Ch
SVCALL	SV call	–	-5	0000_002Ch
PENDSV	Pend SV	–	-2	0000_0038h
SYSTICK	System timer	–	-1	0000_003Ch
INT0	RAMP0_T1	EXT0	0	0000_0040h
INT1	N/A	–	1	0000_0044h
INT2	N/A	–	2	0000_0048h
INT3	N/A	–	3	0000_004Ch
INT4	PMBus	–	4	0000_0050h
INT5	OTP	–	5	0000_0054h
INT6	RAMP0_T2	EXT1	6	0000_0058h
INT7	RAMP1_T1	EXT2	7	0000_005Ch
INT8	RAMP1_T2	EXT3	8	0000_0060h
INT9	CAL_ALL_DONE	EXT4	9	0000_0064h
INT10	PWM_IN	EXT5	10	0000_0068h
INT11	N/A	EXT6	11	0000_006Ch
INT12	WDT	–	12	0000_0070h
INT13	DTIMER1_1	–	13	0000_0074h
INT14	DTIMER1_2	–	14	0000_0078h
INT15	DTIMER2_1	–	15	0000_007Ch
INT16	DTIMER2_2	–	16	0000_0080h
INT17	DTIMER3_1	–	17	0000_0084h
INT18	DTIMER3_2	–	18	0000_0088h
INT19	GPIO0	–	19	0000_008Ch
INT20	GPIO1	–	20	0000_0090h
INT21	Fault	EXT7	21	0000_0094h
INT22	Test	EXT8	22	0000_0098h
INT23	VC0_VCONTROL	EXT9	23	0000_009Ch
INT24	VC1_VCONTROL	EXT10	24	0000_00A0h
INT25	TLM	EXT11	25	0000_00A4h
INT26	FSW0 logical_or FSW1	EXT12	26	0000_00A8h
INT27	N/A	EXT13	27	0000_00ACh
INT28	DMA	–	28	0000_00B0h
INT29	N/A	–	29	0000_00B4h
INT30	I^2C	–	30	0000_00B8h
INT31	UART	–	31	0000_00BCh

Every interrupt INT31-INT0 can be programmed to be used as NMI, by configuring the NMI source-select register NMI_SRC_EN in SCU.

15.2.2 Cortex®-M0 memory map

The memory map accessible by Cortex®-M0 is described in Table 90.

Table 90 Cortex®-M0 memory map (remap = 0)

Address range	Size	Peripheral	Bus matrix master
0000_0000h - 0001_3FFFh	80 kB	ROM	M0
0001_4000h - 0001_FFFFh	48 kB	Reserved	
0002_0000h - 0002_FFFFh	64 kB	OTP	
0003_0000h - 0003_FFFFh	64 kB	OTP (replica)	
0004_0000h - 0FFF_FFFFh		Reserved	
1000_0000h - 1001_3FFFh	80 kB	ROM	M0
1001_4000h - 1001_FFFFh	48 kB	Reserved	
1002_0000h - 1002_FFFFh	64 kB	OTP	
1003_0000h - 1003_FFFFh	64 kB	OTP (replica)	
1004_0000h - 1004_FFFFh		Reserved	
1005_0000h - 1005_3FFFh	16 kB	RAM1 (replica)	M2
1005_4000h - 1005_7FFFh	16 kB	RAM1 (replica)	
1005_8000h - 1005_BFFFh	16 kB	RAM1 (replica)	
1005_C000h - 1005_FFFFh	16 kB	RAM1	
1006_0000h - 1006_3FFFh	16 kB	RAM2	M1
1006_4000h - 1006_7FFFh	16 kB	RAM2 (replica)	
1006_8000h - 1006_BFFFh	16 kB	RAM2 (replica)	
1006_C000h - 1006_FFFFh	16 kB	RAM2 (replica)	
1007_0000h - 2004_FFFFh		Reserved	
2005_0000h - 2005_3FFFh	16 kB	RAM1 (replica)	M2
2005_4000h - 2005_7FFFh	16 kB	RAM1 (replica)	
2005_8000h - 2005_BFFFh	16 kB	RAM1 (replica)	
2005_C000h - 2005_FFFFh	16 kB	RAM1	
2006_0000h - 2006_3FFFh	16 kB	RAM2	M1
2006_4000h - 2006_7FFFh	16 kB	RAM2 (replica)	
2006_8000h - 2006_BFFFh	16 kB	RAM2 (replica)	
2006_C000h - 2006_FFFFh	16 kB	RAM2 (replica)	
2007_0000h - 2FFF_FFFFh		Reserved	
3000_0000h - 3001_3FFFh	80 kB	ROM	M0
3001_4000h - 3001_FFFFh	48 kB	Reserved	
3002_0000h - 3002_FFFFh	64 kB	OTP	
3003_0000h - 3003_FFFFh	64 kB	OTP (replica)	

Address range	Size	Peripheral	Bus matrix master
3004_0000h - 3FFF_FFFFh		Reserved	
4000_0000h - 400F_FFFFh		CSC	M3
4010_0000h - 4FFF_FFFFh		Reserved	
5000_0000h - 500F_FFFFh		OTP CONF, DMA CONF	M4
5010_0000h - 5FFF_FFFFh		Reserved	
6000_0000h - 600F_FFFFh		WDT, DTIMER1/2/3, GPIO0/1	M5
6010_0000h - 6FFF_FFFFh		Reserved	
7000_0000h - 7007_FFFFh		BIF REGFILE (CONTROL)	M6
7008_0000h - 700F_FFFFh		PMBus/CRC/I^2C	M7
7010_0000h - DFFF_FFFFh		Reserved	
E000_0000h - E00F_FFFFh		Cortex®-M0 private peripherals	
E010_0000h - FFFF_FFFFh		Reserved	

Address decoding is implemented in such a way as to minimize the digital comparators, achieving smaller and faster logic but with the side-effect of building logical replicas into the memory map. Replicas can be used by the FW if needed, or can be considered as reserved space.

In one case replicas are especially useful: it is simpler and more efficient for FW to have contiguous memory space for RAM, but it can be particularly HW-expensive, with 16 kB memory size (18-bit address decoding); allowing the memory replica and selecting by FW the proper space can grant a contiguous memory space (2005_C000h - 2006_3FFFh).

15.2.3 Remapping feature

In microcontroller architectures based on Arm® Cortex®-M0, it is a common feature to have a remap function.

Cortex®-M0 has the interrupt table hard-coded in the lowest address space (0000_0004h - 0000_00BCh) that is normally occupied by a non-volatile memory (NVM). In the case of XDPP1100 this is the 80 kB ROM. This means that all interrupt serving routines (ISRs) are hard-coded and fixed.

Remap is a function that can redirect the lowest address space from ROM to RAM: addresses are rerouted on RAM instead of ROM. In RAM, FW is able to redefine or change (from what is hard-coded in ROM) all the ISRs.

The remap function is activated by the REMAP bit in CPUS_CFG register in SCU module.

Cortex®-M0, reading address 0000_0000h, is accessing the first location of the ROM when the REMAP bit is low; it is accessing the first location of the RAM (writable) when the REMAP bit is high.

During the boot sequence, the FW is using the ISRs in ROM, then it is copying them (or writing a new version) in RAM and finally setting the REMAP bit to high to execute them from the RAM.

The Cortex®-M0 memory map, when REMAP is set, is shown in Table 91.

Table 91 Cortex®-M0 memory map (remap = 1)

Address range	Size	Peripheral	Bus matrix master
0000_0000h - 0000_3FFFh	16 kB	RAM1	M2
0000_4000h - 0000_7FFFh	16 kB	RAM1 (replica)	
0000_8000h - 0000_BFFFh	16 kB	RAM1 (replica)	
0000_C000h - 0000_FFFFh	16 kB	RAM1 (replica)	
0001_0000h - 0FFF_FFFFh		Reserved	
1000_0000h - 1001_3FFFh	80 kB	ROM	M0
1001_4000h - 1001_FFFFh	48 kB	Reserved	
1002_0000h - 1002_FFFFh	64 kB	OTP	
1003_0000h - 1003_FFFFh	64 kB	OTP (replica)	
1004_0000h - 1004_FFFFh		Reserved	
1005_0000h - 1005_3FFFh	16 kB	RAM1 (replica)	M2
1005_4000h - 1005_7FFFh	16 kB	RAM1 (replica)	
1005_8000h - 1005_BFFFh	16 kB	RAM1 (replica)	
1005_C000h - 1005_FFFFh	16 kB	RAM1	
1006_0000h - 1006_3FFFh	16 kB	RAM2	M1
1006_4000h - 1006_7FFFh	16 kB	RAM2 (replica)	
1006_8000h - 1006_BFFFh	16 kB	RAM2 (replica)	
1006_C000h - 1006_FFFFh	16 kB	RAM2 (replica)	
1007_0000h - 2004_FFFFh		Reserved	
2005_0000h - 2005_3FFFh	16 kB	RAM1 (replica)	M2
2005_4000h - 2005_7FFFh	16 kB	RAM1 (replica)	
2005_8000h - 2005_BFFFh	16 kB	RAM1 (replica)	
2005_C000h - 2005_FFFFh	16 kB	RAM1	
2006_0000h - 2006_3FFFh	16 kB	RAM2	M1
2006_4000h - 2006_7FFFh	16 kB	RAM2 (replica)	
2006_8000h - 2006_BFFFh	16 kB	RAM2 (replica)	
2006_C000h - 2006_FFFFh	16 kB	RAM2 (replica)	
2007_0000h - 2FFF_FFFFh		Reserved	
3000_0000h - 3001_3FFFh	80 kB	ROM	M0
3001_4000h - 3001_FFFFh	48 kB	Reserved	
3002_0000h - 3002_FFFFh	64 kB	OTP	
3003_0000h - 3003_FFFFh	64 kB	OTP (replica)	
3004_0000h - 3FFF_FFFFh		Reserved	
4000_0000h - 400F_FFFFh		CSC	M3
4010_0000h - 4FFF_FFFFh		Reserved	
5000_0000h - 500F_FFFFh		OTP CONF, DMA CONF	M4
5010_0000h - 5FFF_FFFFh		Reserved	

Address range	Size	Peripheral	Bus matrix master
6000_0000h - 600F_FFFFh		WDT, DTIMER1/2/3, GPIO0/1	M5
6010_0000h - 6FFF_FFFFh		Reserved	
7000_0000h - 7007_FFFFh		BIF REGFILE (CONTROL)	M6
7008_0000h - 700F_FFFFh		PMBus/CRC/I^{2}C	M7
7010_0000h - DFFF_FFFFh		Reserved	
E000_0000h - E00F_FFFFh		Cortex®-M0 private peripherals	
E010_0000h - FFFF_FFFFh		Reserved	

15.3 Clock and system controller

The clock and system controller (CSC) unit module includes the following units:

- System controller unit (SCU)
- Clock generator unit (CGU)
- Reset generator unit (RGU)

The CSC implements the AHB interface to enable the Cortex®-M0 to access the units in charge to configure the CPUS and to configure and control the clocks and the resets of all peripherals accessed by the processor (Cortex®-M0).

15.3.1 System controller unit

The SCU configures all the miscellaneous functions of the CPUS functions; in particular, it enables/disables:

- Memory map remapping (i.e., it controls the bus matrix remap signal)
- Debug port control
- CPU wakeup from control engine
- NMI source selection

15.3.1.1 SCU registers

The relevant SCU-related registers and their descriptions are shown in Table 92.

Table 92 SCU-related register descriptions

Register name	Field name	Access	Address	Bits	Description
CPUS_CFG	SET_REMAP	RW	4000_0000h	[0]	This bit controls the remap signal used to change the address map implemented by the bus matrix. 0: Remap signal set to 0 1: Remap signal set to 1
CPUS_CFG	DS_DBGPORT	RW	4000_0000h	[2]	Enable debug port. 0: DBG port connection is disabled

Register name	Field name	Access	Address	Bits	Description
					1: DBG port connection is enabled
CPUS_CFG	EN_EXTWKUP	RW	4000_0000h	[3]	Enable external wakeup (WKUPIN) when the CPUS is in the hibernate state. 0: CPUS external wakeup is disabled 1: CPUS external wakeup is enabled
CPUS_CFG	USR_CNFG0	RW	4000_0000h	[4]	Reserved
CPUS_CFG	USR_CNFG1	RW	4000_0000h	[5]	Reserved
CPUS_CFG	USR_CNFG2	RW	4000_0000h	[6]	Reserved
CPUS_CFG	USR_CNFG3	RW	4000_0000h	[7]	Reserved
CPUS_CFG	USR_CNFG4	RW	4000_0000h	[8]	Reserved
CPUS_CFG	USR_CNFG5	RW	4000_0000h	[9]	Reserved
CPUS_CFG	USR_CNFG6	RW	4000_0000h	[10]	Reserved
CPUS_CFG	OTP_KEEP_PWR_ST	RW	4000_0000h	[11]	Controls OTP APB peripheral register reset on a soft reset. 0: Reset OTP APB peripheral register values on soft reset 1: Maintain OTP APB peripheral register values on soft reset
CPUS_CFG	SEL_SRC_DTIMER32_KRN_CLK	RW	4000_0000h	[12]	Select the source of DTIMER32_KERNEL_CLK. 0: Internal CPUS clock source 1: Loop 1 F_{switch} rate clock
CPUS_CFG	SEL_SRC_DTIMER31_KRN_CLK	RW	4000_0000h	[13]	Select the source of DTIMER31_KERNEL_CLK. 0: Internal CPUS clock source 1: Loop 0 F_{switch} rate clock
CPUS_CFG	SEL_SRC_DTIMER22_KRN_CLK	RW	4000_0000h	[14]	Select the source of DTIMER22_KERNEL_CLK. 0: Internal CPUS clock source (required) 1: Invalid setting
CPUS_CFG	SEL_SRC_DTIMER21_KRN_CLK	RW	4000_0000h	[15]	Select the source of DTIMER21_KERNEL_CLK.

Register name	Field name	Access	Address	Bits	Description
					0: Internal CPUS clock source (required) 1: Invalid setting
CPUS_CFG	SEL_SRC_DTIMER12_KRN_CLK	RW	4000_0000h	[16]	Select the source of DTIMER12_KERNEL_CLK. 0: Internal CPUS clock source (required) 1: Invalid setting
CPUS_CFG	SEL_SRC_DTIMER11_KRN_CLK	RW	4000_0000h	[17]	Select the source of DTIMER11_KERNEL_CLK. 0: Internal CPUS clock source (required) 1: Invalid setting
CPUS_CFG	EN_AUX_EXTWKUP	RW	4000_0000h	[20]	Enable auxiliary external wakeup source to wake up CPUS when in the power-down or hibernate state. 0: CPUS EXT-3 wakeup source is disabled 1: CPUS EXT-3 wakeup source is enabled
CPUS_CFG	EN_PMBUS_WKUP	RW	4000_0000h	[21]	Enable internal source PMBUS_IRQ to wake up CPUS when in the power-down or hibernate state. 0: CPUS PMBUS_IRQ wakeup source is disabled 1: CPUS PMBUS_IRQ wakeup source is enabled
CPUS_CFG	EN_GPIO0_WKUP	RW	4000_0000h	[22]	Enable internal source GPIO0_IRQ to wakeup CPUS when in the power-down or hibernate state. 0: CPUS GPIO0_IRQ wakeup source is disabled 1: CPUS GPIO0_IRQ wakeup source is enabled

Register name	Field name	Access	Address	Bits	Description
CPUS_CFG	EN_GPIO1_WKUP	RW	4000_0000h	[23]	Enable internal source GPIO1_IRQ to wake up CPUS when in the power-down or hibernate state. 0: CPUS GPIO1_IRQ wakeup source is disabled 1: CPUS GPIO1_IRQ wakeup source is enabled
CPUS_CFG_SET	SET_REMAP	RW	4000_0004h	[0]	This bit controls the remap signal used to change the address map implemented by the bus matrix. 0: No change to existing value 1: Remap signal set to 1
CPUS_CFG_SET	DS_DBGPORT	RW	4000_0004h	[2]	Disable debug port. 0: No change to existing value 1: DBG port connection is enabled
CPUS_CFG_SET	EN_EXTWKUP	W	4000_0004h	[3]	Enable external wakeup (WKUPIN) when the CPUS is in the hibernate state. 0: No change to existing value 1: CPUS external wakeup is enabled
CPUS_CFG_SET	USR_CNFG0	W	4000_0004h	[4]	Reserved
CPUS_CFG_SET	USR_CNFG1	W	4000_0004h	[5]	Reserved
CPUS_CFG_SET	USR_CNFG2	W	4000_0004h	[6]	Reserved
CPUS_CFG_SET	USR_CNFG3	W	4000_0004h	[7]	Reserved
CPUS_CFG_SET	USR_CNFG4	W	4000_0004h	[8]	Reserved
CPUS_CFG_SET	USR_CNFG5	W	4000_0004h	[9]	Reserved
CPUS_CFG_SET	USR_CNFG6	W	4000_0004h	[10]	Reserved
CPUS_CFG_SET	OTP_KEEP_PWR_ST	W	4000_0004h	[11]	Controls OTP APB peripheral register reset on a soft reset. 0: No change to existing value

Register name	Field name	Access	Address	Bits	Description
					1: Maintain OTP APB peripheral register values on soft reset
CPUS_CFG_SET	SEL_SRC_DTIMER32_KRN_CLK	W	4000_0004h	[12]	Select the source of DTIMER32_KERNEL_CLK. 0: No change to existing value 1: Loop 1 F_{switch} rate clock
CPUS_CFG_SET	SEL_SRC_DTIMER31_KRN_CLK	W	4000_0004h	[13]	Select the source of DTIMER31_KERNEL_CLK. 0: No change to existing value 1: Loop 0 F_{switch} rate clock
CPUS_CFG_SET	SEL_SRC_DTIMER22_KRN_CLK	W	4000_0004h	[14]	Select the source of DTIMER22_KERNEL_CLK. 0: No change to existing value 1: Invalid setting
CPUS_CFG_SET	SEL_SRC_DTIMER21_KRN_CLK	W	4000_0004h	[15]	Select the source of DTIMER21_KERNEL_CLK. 0: No change to existing value 1: Invalid setting
CPUS_CFG_SET	SEL_SRC_DTIMER12_KRN_CLK	W	4000_0004h	[16]	Select the source of DTIMER12_KERNEL_CLK. 0: No change to existing value 1: Invalid setting
CPUS_CFG_SET	SEL_SRC_DTIMER11_KRN_CLK	W	4000_0004h	[17]	Select the source of DTIMER11_KERNEL_CLK. 0: No change to existing value 1: Invalid setting
CPUS_CFG_SET	EN_AUX_EXTWKUP	W	4000_0004h	[20]	Enable auxiliary external wakeup source to wake up CPUS when in the power-down or hibernate state. 0: No change to existing value 1: CPUS EXT-3 wakeup source is enabled
CPUS_CFG_SET	EN_PMBUS_WKUP	W	4000_0004h	[21]	Enable internal source PMBUS_IRQ to wake up

Register name	Field name	Access	Address	Bits	Description
					CPUS when in the power-down or hibernate state. 0: No change to existing value 1: CPUS PMBUS_IRQ wakeup source is enabled
CPUS_CFG_SET	EN_GPIO0_WKUP	W	4000_0004h	[22]	Enable internal source GPIO0_IRQ to wake up CPUS when in the power-down or hibernate state. 0: No change to existing value 1: CPUS GPIO0_IRQ wakeup source is enabled
CPUS_CFG_SET	EN_GPIO1_WKUP	W	4000_0004h	[23]	Enable internal source GPIO1_IRQ to wake up CPUS when in the power-down or hibernate state. 0: No change to existing value 1: CPUS GPIO1_IRQ wakeup source is enabled
CPUS_CFG_CLR	SET_REMAP	W	4000_0008h	[0]	This bit controls the remap signal used to change the address map implemented by the bus matrix. 0: No change to existing value 1: Remap signal set to 0
CPUS_CFG_CLR	DS_DBGPORT	W	4000_0008h	[2]	Disable debug port. 0: No change to existing value 1: DBG port connection is disabled
CPUS_CFG_CLR	EN_EXTWKUP	W	4000_0008h	[3]	Enable external wakeup (WKUPIN) when the CPUS is in the hibernate state.

Register name	Field name	Access	Address	Bits	Description
					0: No change to existing value 1: CPUS external wakeup is disabled
CPUS_CFG_CLR	USR_CNFG0	W	4000_0008h	[4]	Reserved
CPUS_CFG_CLR	USR_CNFG1	W	4000_0008h	[5]	Reserved
CPUS_CFG_CLR	USR_CNFG2	W	4000_0008h	[6]	Reserved
CPUS_CFG_CLR	USR_CNFG3	W	4000_0008h	[7]	Reserved
CPUS_CFG_CLR	USR_CNFG4	W	4000_0008h	[8]	Reserved
CPUS_CFG_CLR	USR_CNFG5	W	4000_0008h	[9]	Reserved
CPUS_CFG_CLR	USR_CNFG6	W	4000_0008h	[10]	Reserved
CPUS_CFG_CLR	OTP_KEEP_PWR_ST	W	4000_0008h	[11]	Controls OTP APB peripheral register reset on a soft reset. 0: No change to existing value 1: Reset OTP APB peripheral register values on soft reset
CPUS_CFG_CLR	SEL_SRC_DTIMER32_KRN_CLK	W	4000_0008h	[12]	Select the source of DTIMER32_KERNEL_CLK. 0: No change to existing state 1: Internal CPUS clock source
CPUS_CFG_CLR	SEL_SRC_DTIMER31_KRN_CLK	W	4000_0008h	[13]	Select the source of DTIMER31_KERNEL_CLK. 0: No change to existing state 1: Internal CPUS clock source
CPUS_CFG_CLR	SEL_SRC_DTIMER22_KRN_CLK	W	4000_0008h	[14]	Select the source of DTIMER22_KERNEL_CLK. 0: No change to existing state 1: Internal CPUS clock source
CPUS_CFG_CLR	SEL_SRC_DTIMER21_KRN_CLK	W	4000_0008h	[15]	Select the source of DTIMER21_KERNEL_CLK. 0: No change to existing state 1: Internal CPUS clock source

Register name	Field name	Access	Address	Bits	Description
CPUS_CFG_CLR	SEL_SRC_DTIMER12_KRN_CLK	W	4000_0008h	[16]	Select the source of DTIMER12_KERNEL_CLK. 0: No change to existing state 1: Internal CPUS clock source
CPUS_CFG_CLR	SEL_SRC_DTIMER11_KRN_CLK	W	4000_0008h	[17]	Select the source of DTIMER11_KERNEL_CLK. 0: No change to existing state 1: Internal CPUS clock source
CPUS_CFG_CLR	EN_AUX_EXTWKUP	W	4000_0008h	[20]	Enable auxiliary external wakeup source to wake up CPUS when in the power-down or hibernate state. 0: No change to existing value 1: CPUS EXT-3 wakeup source is disabled
CPUS_CFG_CLR	EN_PMBUS_WKUP	W	4000_0008h	[21]	Enable internal source PMBUS_IRQ to wake up CPUS when in the power-down or hibernate state. 0: No change to existing value 1: CPUS PMBUS_IRQ wakeup source is disabled
CPUS_CFG_CLR	EN_GPIO0_WKUP	W	4000_0008h	[22]	Enable internal source GPIO0_IRQ to wake up CPUS when in the power-down or hibernate state. 0: No change to existing value 1: CPUS GPIO0_IRQ wakeup source is disabled
CPUS_CFG_CLR	EN_GPIO1_WKUP	W	4000_0008h	[23]	Enable internal source GPIO1_IRQ to wakeup CPUS when in the power-down or hibernate state.

Register name	Field name	Access	Address	Bits	Description
					0: No change to existing value 1: CPUS GPIO1_IRQ wakeup source is disabled
NMI_SRC_EN	EXT0_NMI_EN	RW	4000_000Ch	[0]	External (EXT0_IRQn) NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	CSC_NMI_EN	RW	4000_000Ch	[3]	Reserved
NMI_SRC_EN	PMBUS_NMI_EN	RW	4000_000Ch	[4]	PMBus NMI control. 0: Disabled Enabled
NMI_SRC_EN	OTP1_W_NMI_EN	RW	4000_000Ch	[5]	OTP NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	EXT1_NMI_EN	RW	4000_000Ch	[6]	External (EXT1_IRQn) NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	EXT2_NMI_EN	RW	4000_000Ch	[7]	External (EXT2_IRQn) NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	EXT3_NMI_EN	RW	4000_000Ch	[8]	External (EXT3_IRQn) NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	EXT4_NMI_EN	RW	4000_000Ch	[9]	External (EXT4_IRQn) NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	EXT5_NMI_EN	RW	4000_000Ch	[10]	External (EXT5_IRQn) NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	EXT6_NMI_EN	RW	4000_000Ch	[11]	Reserved
NMI_SRC_EN	WDT_NMI_EN	RW	4000_000Ch	[12]	WDT NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	DTIMER1_0_NMI_EN	RW	4000_000Ch	[13]	DTIMER_11 NMI control. 0: Disabled 1: Enabled

Register name	Field name	Access	Address	Bits	Description
NMI_SRC_EN	DTIMER1_1_NMI_EN	RW	4000_000Ch	[14]	DTIMER_12 NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	DTIMER2_0_NMI_EN	RW	4000_000Ch	[15]	DTIMER_21 NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	DTIMER2_1_NMI_EN	RW	4000_000Ch	[16]	DTIMER_22 NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	DTIMER3_0_NMI_EN	RW	4000_000Ch	[17]	DTIMER_31 NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	DTIMER3_1_NMI_EN	RW	4000_000Ch	[18]	DTIMER_32 NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	GPIO0_NMI_EN	RW	4000_000Ch	[19]	GPIO0 NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	GPIO1_NMI_EN	RW	4000_000Ch	[20]	GPIO1 NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	FAULT_NMI_EN	RW	4000_000Ch	[21]	Fault NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	TEST_NMI_EN	RW	4000_000Ch	[22]	Test NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	VC0_VCONTROL_NMI_EN	RW	4000_000Ch	[23]	$V_{CONTROL0}$ NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	VC1_VCONTROL_NMI_EN	RW	4000_000Ch	[24]	$V_{CONTROL1}$ NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	TLM_NMI_EN	RW	4000_000Ch	[25]	Telemetry NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	FSW0_OR_FSW1_NMI_EN	RW	4000_000Ch	[26]	F_{switch} NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	DMA_NMI_EN	RW	4000_000Ch	[28]	DMA NMI control. 0: Disabled 1: Enabled

Register name	Field name	Access	Address	Bits	Description
NMI_SRC_EN	SSP_NMI_EN	RW	4000_000Ch	[29]	Reserved
NMI_SRC_EN	I2C_NMI_EN	RW	4000_000Ch	[30]	I^2C NMI control. 0: Disabled 1: Enabled
NMI_SRC_EN	UART_NMI_EN	RW	4000_000Ch	[31]	UART NMI control. 0: Disabled 1: Enabled
SPARE_FF	SPARE_FF	RW	4000_0020h	[31:0]	Spare register

15.3.2 Clock generator unit

The clock generation unit (CGU) generates and controls the clock signals of the CPUS section of the XDPP1100 device. It provides the following main system functions:

- Clock schemes and clock generation
- Clock division and control
- Clock muxing and clock gating
- Control signal generation for clock usage

The main input clock for CGU is HOSC_clk (up to 100 MHz) generated in the CGEN block outside the CPUS module. The HOSC_clk provides the primary clock source of the CPUS; the HOSC_clk clock source is optionally divided before mux; the HOSC_clk is also used as a test input clock.

An overview of the CPUS clock domain is shown in Figure 100.

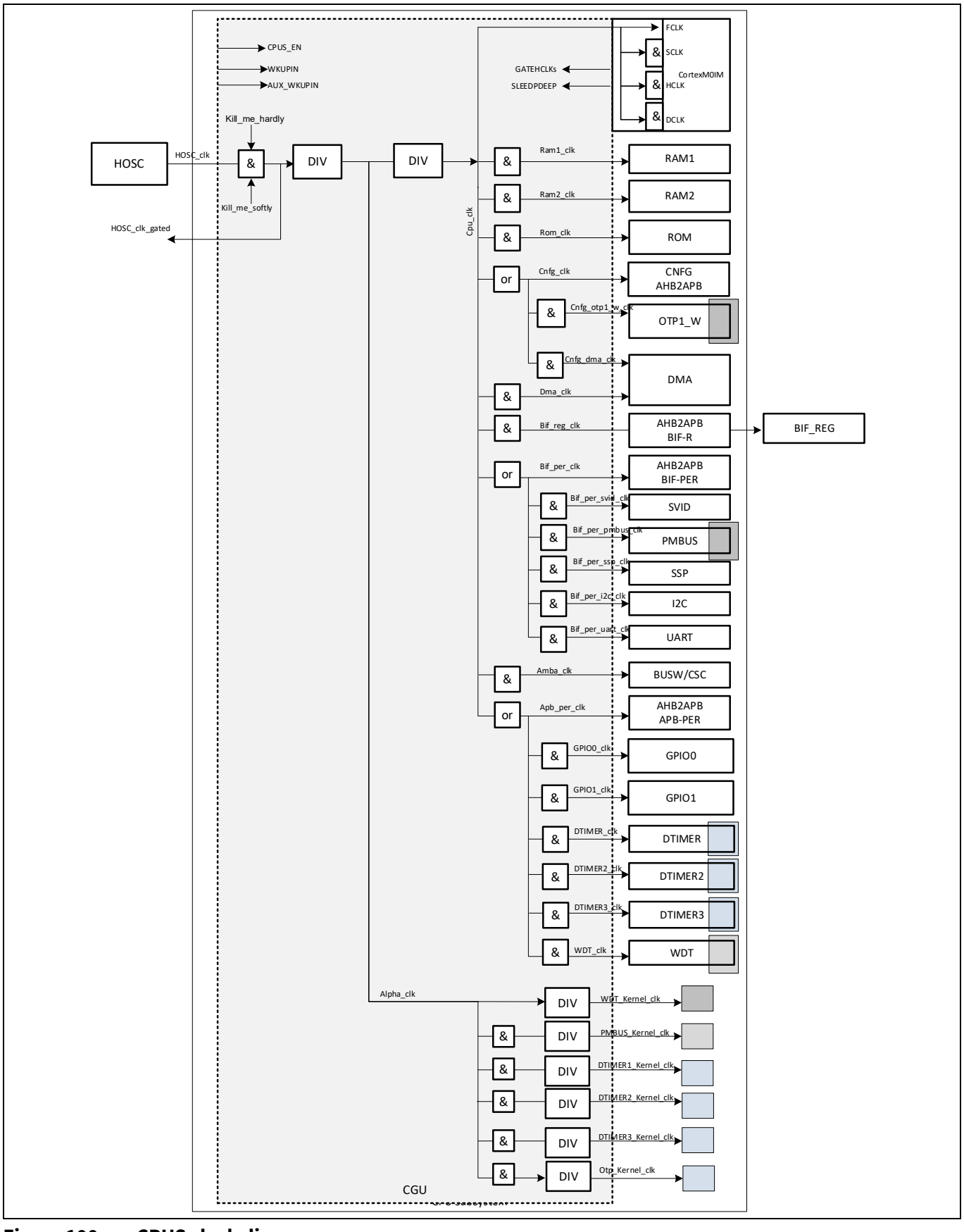

Figure 100 CPUS clock diagram

CPUS clock typical and maximum frequencies and related modules are shown in Table 93.

Table 93 CPUS clock domains

Clock name	Module name	Typ. frequency (MHz)	Max. frequency (MHz)
hosc_clk	CGU	100	100
alpha_clk	CGU	100	100
cpu_clk	CPU (FCLK clock)	100	100
amba_clk	AMBA domain	100	100
rom_clk	ROM	100	100
ram1_clk	RAM1	100	100
ram2_clk	RAM2	100	100
dma_clk	DMA	100	100
bif_reg_clk	BIF REG	100	100
bif_per_clk	BIF PER bridge	100	100
bif_per_svid_clk	CRC32 peripheral	100	100
bif_per_pmbus	BIF PER PMBus peripheral	100	100
bif_per_ssp_clk	Not used	–	–
bif_per_i2c_clk	BIF PER I^2C peripheral	100	100
bif_per_uart_clk	BIF PER UART peripheral	100	100
cnfg_clk	CPU APB PER bridge	100	100
cnfg_dma_clk	DMA config. port	100	100
cnfg_otp1_w_clk	OTP interface peripheral	100	100
otp_kernel_clk	OTP kernel	25	100
apb_per_clk	APB PER bridge	100	100
dtimer1_clk	DTIMER1 peripheral	100	100
dtimer2_clk	DTIMER2 peripheral	100	100
dtimer3_clk	DTIMER3 peripheral	100	100
wdt_clk	WDT interface clock	100	100
gpio0_clk	GPIO0 peripheral	100	100
gpio1_clk	GPIO1 peripheral	100	100
dtimer1_kernel_clk	DTIMER1 kernel	100	100
dtimer2_kernel_clk	DTIMER2 kernel	100	100
dtimer3_kernel_clk	DTIMER3 kernel	100	100
wdt_kernel_clk	WDT kernel	3.125	100
pmbus_kernel_clk	PMBus kernel	25	100

15.3.2.1 Clock dividers

The CPUS clock structure includes eight programmable clock dividers (Table 94).

The alpha clock is the root divider to scale the entire CPUS clock structure.

The CPU clock divider scales the Cortex®-M0 microcontroller and the AMBA® bus infrastructure.

The watchdog and timer clock dividers scale the counter clocks, keeping the interface on the AMBA® bus clock unchanged, and allowing the time constant of the counters to be extended to bigger time intervals. This is particularly effective for the watchdog, for example, where the timer interval is in the range of seconds.

The OTP kernel clock divider enables adjustment of the ratio of the OTP wrapper logic, which should work at a nominal frequency of 25 MHz regardless of alpha clock speed.

The PMBus kernel clock divider enables adjustment of the ratio of the PMBus core logic with respect to alpha clock frequency.

Table 94 CPUS clock dividers

Clock domain	Control register	Divider control	Description
alpha_clk	ALPHA_CLK_DIV_CTRL	alpha_clk_div[4:0]	Root clock divider
cpu_clk	CPU_CLK_DIV_CTRL	cpuclk_div[4:0]	CPU core and bus clock divider
wdt_kernel_clk	KRN_CLK_DIV_CTRL	wdt_kernel_clk_div[4:0]	Watchdog kernel counter clock divider
pmbus_kernel_clk	KRN_CLK_DIV_CTRL	pmbus_kernel_clk_div[4:0]	PMBus logic core clock divider
dtimer1_kernel_clk	KRN_CLK_DIV_CTRL	dtimer1_kernel_clk_div[4:0]	D_{TIMER1} counters clock divider
dtimer2_kernel_clk	KRN_CLK_DIV_CTRL	dtimer2_kernel_clk_div[4:0]	D_{TIMER2} counters clock divider
dtimer3_kernel_clk	KRN_CLK_DIV_CTRL	dtimer3_kernel_clk_div[4:0]	D_{TIMER3} counters clock divider
otp_kernel_clk	KRN_CLK_DIV_CTRL	otp_kernel_clk_div[4:0]	OTP kernel clock divider

All the programmable dividers can take any input divisor vector. The divisor should be programmed according to the module's required clock frequency. When divisor vector = 0, the function of the divisor is "DIV by 1"; when divisor vector = 1, the function of the divisor is "DIV by 2" and so on. In general, xxx_clk_out = clk_in / (vector + 1).

15.3.2.2 Clock gating

All the clock gating cells, with the exception of the primary-source clock gating, are based on the hierarchy shown in Figure 101, where the last stage of gating is controlled by the clock enable register, and the previous stage by the status of the Cortex®-M0 power state, defined by the signals SLEEPING and SLEEPDEEP (only SLEEPDEEP will be used, as the GATEHCLK signal already takes into account the SLEEPING signal), if enabled in the clock sleep and deep sleep mask configuration registers.

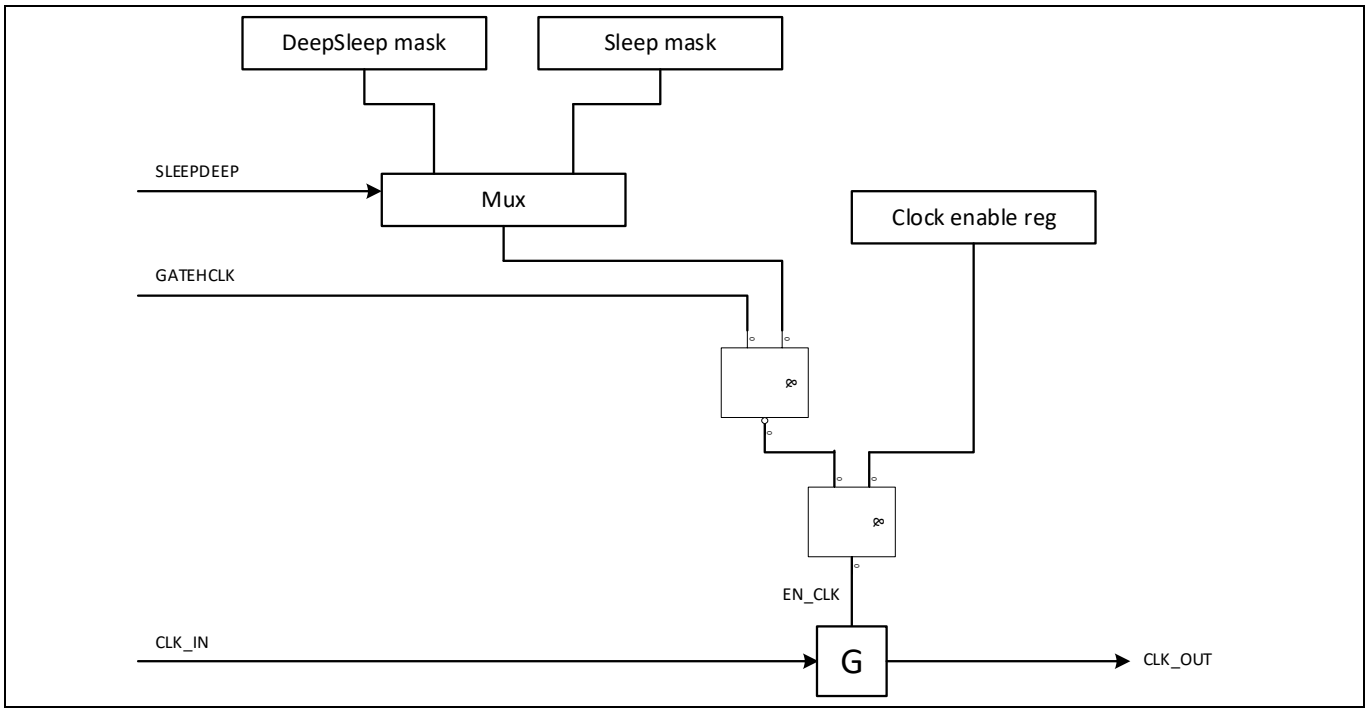

Figure 101 Clock gating structure

15.3.2.3 Primary-source clock gating

The primary-source clock gating has the structure shown in Figure 102, where the last stage of gating is controlled by the HOSC_SW_CLK_GATING_CTRL register or by the HOSC_HW_CLK_GATING_CTRL register while the previous stages have a structure similar to the others.

The HOSC_HW_CLK_GATING_CTRL register enables the input signal CPUS_EN to take control of the CPUS primary input clock source: a low-level CPUS_EN signal will gate the main clock, forcing the CPUS to enter the hibernate state (if it has not already); a high-level CPUS_EN signal will remove the clock gate (HWEN_CLK=1), recovering the previous CPUS functionality.

The HOSC_SW_CLK_GATING_CTRL register enables the SW to take control of the CPUS primary input clock source. Gating the primary-source clock, the CPUS will freeze: external reset input signal assertion or WKUP_IN signal assertion (if previously enabled by setting the proper bit in the SCU configuration register) has to be provided. Asserting the external reset will force CPU to re-boot; while asserting the WKUP_IN signal the processor code will continue to execute code from the last instruction.

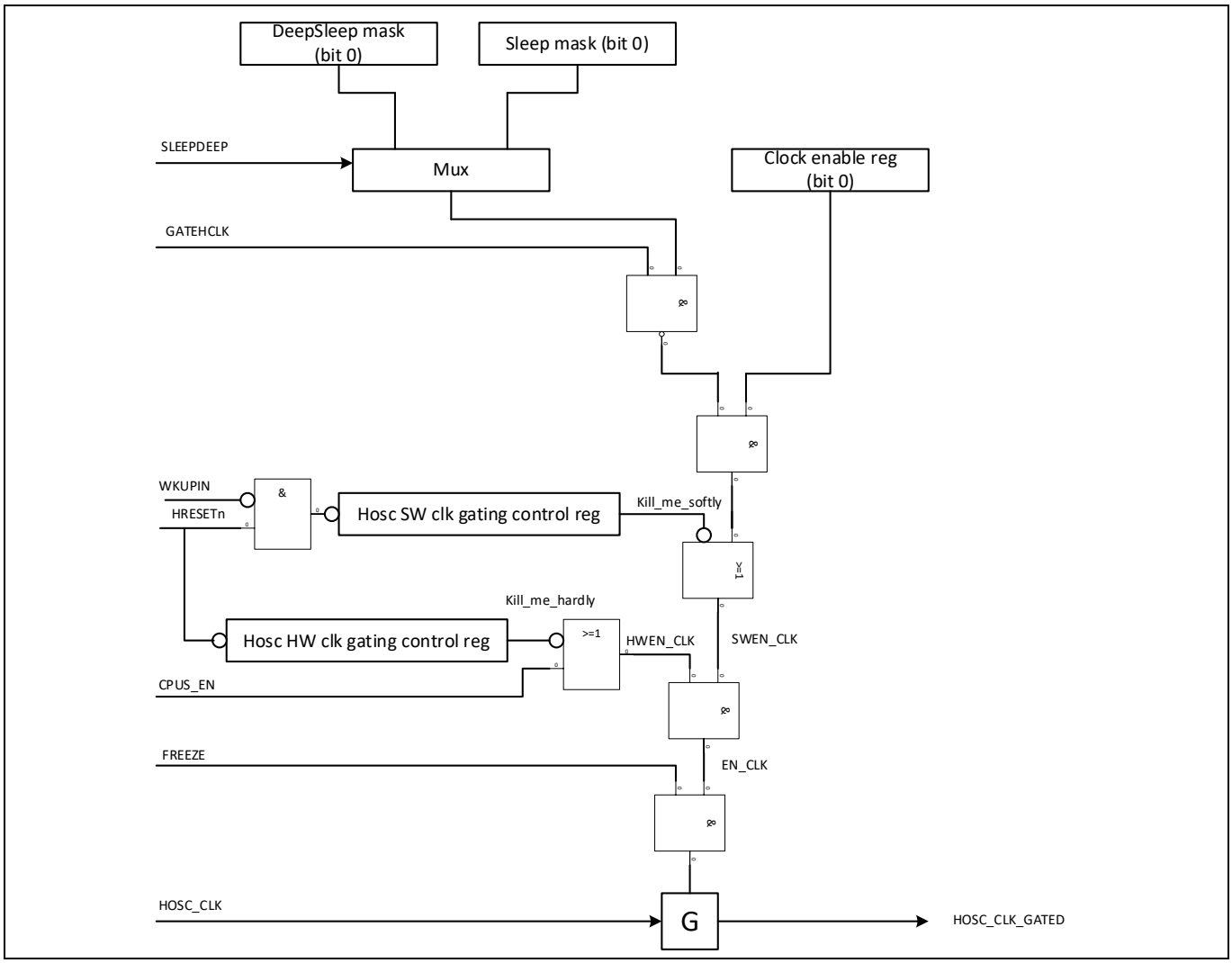

Figure 102 Primary-source clock gating structure

15.3.2.4 CGU registers

The relevant CGU-related registers and their descriptions are provided in Table 95.

Table 95 CGU-related register description

Register name	Field name	Access	Address	Bits	Description
ALPHA_CLK_DIV_CTRL	alpha_clk_div	RW	4000_2000h	[4:0]	This register configures the divider of the main input gated clock source, hosc_clk_gated. The divide ratio is equal to alpha_clk_div[4:0] + 1. For example, the reset setting alpha_clk_div[4:0] = 3 generates a default clock frequency of alpha_clk = hosc_clk/4.
CPU_CLK_DIV_CTRL	cpuclk_div	RW	4000_2004h	[4:0]	This register configures the divider of cpu_clk. The divide ratio is equal to cpuclk_div[4:0] + 1. For

Register name	Field name	Access	Address	Bits	Description
					example, the reset setting cpuclk_div[4:0] = 0 generates a default clock frequency of cpu_clk = alpha_clk/1
KRN_CLK_DIV_CTRL	dtimer1_kernel_clk_div	RW	4000_2008h	[4:0]	This register configures the divider of the D_{TIMER1} kernel clock. The divide ratio is equal to dtimer1_kernel_clk_div[4:0] + 1. For example, the reset setting dtimer1_kernel_clk_div[4:0] = 7 generates a default clock frequency of dtimer1_kernel_clk = alpha_clk/8
KRN_CLK_DIV_CTRL	dtimer2_kernel_clk_div	RW	4000_2008h	[9:5]	This register configures the divider of the D_{TIMER2} kernel clock. The divide ratio is equal to dtimer2_kernel_clk_div[4:0] + 1. For example, the reset setting dtimer2_kernel_clk_div[4:0] = 24 generates a default clock frequency of dtimer2_kernel_clk = alpha_clk/25
KRN_CLK_DIV_CTRL	dtimer3_kernel_clk_div	RW	4000_2008h	[14:10]	This register configures the divider of the D_{TIMER3} kernel clock. The divide ratio is equal to dtimer3_kernel_clk_div[4:0] + 1. For example, the reset setting dtimer3_kernel_clk_div[4:0] = 7 generates a default clock frequency of dtimer3_kernel_clk = alpha_clk/8
KRN_CLK_DIV_CTRL	wdt_kernel_clk_div	RW	4000_2008h	[19:15]	This register configures the divider of the WDT kernel clock. The divide ratio is equal to wdt_kernel_clk_div[4:0] + 1. For example, the reset setting wdt_kernel_clk_div[4:0] = 0 generates a default clock frequency of wdt_kernel_clk = alpha_clk/1
KRN_CLK_DIV_CTRL	pmbus_kernel_clk_div	RW	4000_2008h	[24:20]	This register configures the divider of the PMBus kernel clock. The divide ratio is equal to pmbus_kernel_clk_div[4:0] + 1. For example, the reset setting pmbus_kernel_clk_div[4:0] = 0 generates a default clock

Register name	Field name	Access	Address	Bits	Description
					frequency of pmbus_kernel_clk = alpha_clk/1
KRN_CLK_DIV_CTRL	otp_kernel_clk_div	RW	4000_2008h	[29:25]	This register configures the divider of the OTP kernel clock. The divide ratio is equal to otp_kernel_clk_div[4:0] + 1. For example, the reset setting otp_kernel_clk_div[4:0] = 0 generates a default clock frequency of otp_kernel_clk = alpha_clk/1
CLKOUT_CTRL	clkout_mux	RW	4000_200Ch	[3:0]	Function not in use
CLKOUT_CTRL	clkout_mux_div	RW	4000_200Ch	[7:4]	Function not in use
CLKOUT_CTRL	clkout_g	RW	4000_200Ch	[8]	Function not in use
CLK_EN_CTRL	hosc_clk_g	RW	4000_2010h	[0]	Enable bit for the clock hosc_clk if the hosc_clk primary clock gating control has been enabled. Note: The primary clock gating is performed if this bit is set. The KILL_ME_SOFTLY bit of the HOSC_SW_CLK_GATING_CTRL register is also set. Make sure to enable an external wakeup source before executing the primary clock gating procedure (i.e., entering the hibernate state) in order to avoid permanent loss of the CPUS clock. 0: Clock “hosc_clk” is off only if hosc_clk clock gating control has been enabled 1: Clock “hosc_clk” is live
CLK_EN_CTRL	rom_clk_g	RW	4000_2010h	[1]	Enable bit for the clock rom_clk. 0: Clock “rom_clk” is off 1: Clock “rom_clk” is live
CLK_EN_CTRL	ram1_clk_g	RW	4000_2010h	[2]	Enable bit for the clock ram1_clk. 0: Clock “ram1_clk” is off 1: Clock “ram1_clk” is live
CLK_EN_CTRL	ram2_clk_g	RW	4000_2010h	[3]	Enable bit for the clock ram2_clk. 0: Clock “ram2_clk” is off 1: Clock “ram2_clk” is live
CLK_EN_CTRL	amba_clk_g	RW	4000_2010h	[4]	Enable bit for the clock amba_clk. 0: Clock “amba_clk” is off 1: Clock “amba_clk” is live

Register name	Field name	Access	Address	Bits	Description
CLK_EN_CTRL	dma_clk_g	RW	4000_2010h	[5]	Enable bit for the clock dma_clk. 0: Clock “dma_clk” is off 1: Clock “dma_clk” is live
CLK_EN_CTRL	cnfg_otp1_w_clk_g	RW	4000_2010h	[6]	Enable bit for the clock cnfg_otp1_w_clk and cnfg_clk. 0: Clock “cnfg_otp1_w_clk” is off 1: Clock “cnfg_otp1_w_clk” and cnfg_clk are live
CLK_EN_CTRL	cnfg_dma_clk_g	RW	4000_2010h	[10]	Enable bit for the clock cnfg_dma_clk and cnfg_clk. 0: Clock “cnfg_dma_clk” is off 1: Clock “cnfg_dma_clk” and cnfg_clk are live
CLK_EN_CTRL	bif_reg_clk_g	RW	4000_2010h	[11]	Enable bit for the clock bif_reg_clk. 0: Clock “bif_reg_clk” is off 1: Clock “bif_reg_clk” is live
CLK_EN_CTRL	bif_per_svid_clk_g	RW	4000_2010h	[12]	Enable bit for the clock bif_per_svid_clk and bif_per_clk. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Clock “bif_per_svid_clk” is off 1: Clock “bif_per_svid_clk” and bif_per_clk are live
CLK_EN_CTRL	bif_per_pmbus_clk_g	RW	4000_2010h	[13]	Enable bit for the clock bif_per_pmbus_clk and bif_per_clk. 0: Clock “bif_per_pmbus_clk” is off 1: Clock “bif_per_pmbus_clk” and bif_per_clk are live
CLK_EN_CTRL	bif_per_ssp_clk_g	RW	4000_2010h	[14]	Reserved
CLK_EN_CTRL	bif_per_i2c_clk_g	RW	4000_2010h	[15]	Enable bit for the clock bif_per_i2c_clk and bif_per_clk. 0: Clock “bif_per_i2c_clk” is off 1: Clock “bif_per_i2c_clk” and bif_per_clk are live
CLK_EN_CTRL	bif_per_uart_clk_g	RW	4000_2010h	[16]	Enable bit for the clock bif_per_uart_clk and bif_per_clk. 0: Clock “bif_per_uart_clk” is off 1: Clock “bif_per_uart_clk” and bif_per_clk are live

Register name	Field name	Access	Address	Bits	Description
CLK_EN_CTRL	dtimer1_clk_g	RW	4000_2010h	[17]	Enable bit for the clock dtimer1_clk and apb_per_clk. 0: Clock “dtimer1_clk” is off 1: Clock “dtimer1_clk” and apb_per_clk are live
CLK_EN_CTRL	dtimer2_clk_g	RW	4000_2010h	[18]	Enable bit for the clock dtimer2_clk and apb_per_clk. 0: Clock “dtimer2_clk” is off 1: Clock “dtimer2_clk” and apb_per_clk are live
CLK_EN_CTRL	dtimer3_clk_g	RW	4000_2010h	[19]	Enable bit for the clock dtimer3_clk and apb_per_clk. 0: Clock “dtimer3_clk” is off 1: Clock “dtimer3_clk” and apb_per_clk are live
CLK_EN_CTRL	wdt_clk_g	RW	4000_2010h	[20]	Enable bit for the clock wdt_clk and apb_per_clk. 0: Clock “wdt_clk” is off 1: Clock “wdt_clk” and apb_per_clk are live
CLK_EN_CTRL	gpio0_clk_g	RW	4000_2010h	[21]	Enable bit for the clock gpio0_clk and apb_per_clk. 0: Clock “gpio0_clk” is off 1: Clock “gpio0_clk” and apb_per_clk are live
CLK_EN_CTRL	gpio1_clk_g	RW	4000_2010h	[22]	Enable bit for the clock gpio1_clk and apb_per_clk. 0: Clock “gpio1_clk” is off 1: Clock “gpio1_clk” and apb_per_clk are live
CLK_EN_CTRL	dtimer1_kernel_clk_g	RW	4000_2010h	[23]	Enable bit for the clock dtimer1_kernel_clk. 0: Clock “dtimer1_kernel_clk” is off 1: Clock “dtimer1_kernel_clk” is live
CLK_EN_CTRL	dtimer2_kernel_clk_g	RW	4000_2010h	[24]	Enable bit for the clock dtimer2_kernel_clk. 0: Clock “dtimer2_kernel_clk” is off 1: Clock “dtimer2_kernel_clk” is live
CLK_EN_CTRL	dtimer3_kernel_clk_g	RW	4000_2010h	[25]	Enable bit for the clock dtimer3_kernel_clk.

Register name	Field name	Access	Address	Bits	Description
					0: Clock “dtimer3_kernel_clk” is off 1: Clock “dtimer3_kernel_clk” is live
CLK_EN_CTRL	pmbus_kernel_clk_g	RW	4000_2010h	[26]	Enable bit for the clock pmbus_kernel_clk. 0: Clock “pmbus_kernel_clk” is off 1: Clock “pmbus_kernel_clk” is live
CLK_EN_CTRL	otp_kernel_clk_g	RW	4000_2010h	[27]	Enable bit for the clock otp_kernel_clk. 0: Clock “otp_kernel_clk” is off 1: Clock “otp_kernel_clk” is live
CLK_SLEEP_MSK_CNFG	se_hosc_clk_g	RW	4000_2014h	[0]	Enable hosc_clk clock gating when Cortex®-M0 enters sleep state, if the hosc_clk clock gating control has been enabled. Warning: Only external reset assertion can remove clock gating if no external wakeup sources have been enabled before! 0: Clock “hosc_clk” is not gated by CM0 power state status 1: Clock “hosc_clk” is gated when CM0 is in sleep state if hosc_clk clock gating control has been enabled
CLK_SLEEP_MSK_CNFG	se_rom_clk_g	RW	4000_2014h	[1]	Enable rom_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “rom_clk” is not gated by CM0 power state status 1: Clock “rom_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_ram1_clk_g	RW	4000_2014h	[2]	Enable ram1_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “ram1_clk” is not gated by CM0 power state status 1: Clock “ram1_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_ram2_clk_g	RW	4000_2014h	[3]	Enable ram2_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “ram2_clk” is not gated by CM0 power state status

Register name	Field name	Access	Address	Bits	Description
					1: Clock “ram2_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_amba_clk_g	RW	4000_2014h	[4]	Enable amba_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “amba_clk” is not gated by CM0 power state status 1: Clock “amba_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_dma_clk_g	RW	4000_2014h	[5]	Enable dma_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “dma_clk” is not gated by CM0 power state status 1: Clock “dma_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_cnfg_otp1_w_clk_g	RW	4000_2014h	[6]	Enable cnfg_otp1_w_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “cnfg_otp1_w_clk” is not gated by CM0 power state status 1: Clock “cnfg_otp1_w_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_cnfg_dma_clk_g	RW	4000_2014h	[10]	Enable cnfg_dma_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “cnfg_dma_clk” is not gated by CM0 power state status 1: Clock “cnfg_dma_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_bif_reg_clk_g	RW	4000_2014h	[11]	Enable bif_reg_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “bif_reg_clk” is not gated by CM0 power state status 1: Clock “bif_reg_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_bif_per_svid_clk_g	RW	4000_2014h	[12]	Enable bif_per_svid_clk clock gating when Cortex®-M0 enters sleep state. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Clock “bif_per_svid_clk” is not gated by CM0 power state status

Register name	Field name	Access	Address	Bits	Description
					1: Clock “bif_per_svid_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_bif_per_pmbus_clk_g	RW	4000_2014h	[13]	Enable bif_per_pmbus_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “bif_per_pmbus_clk” is not gated by CM0 power state status 1: Clock “bif_per_pmbus_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_bif_per_ssp_clk_g	RW	4000_2014h	[14]	Reserved
CLK_SLEEP_MSK_CNFG	se_bif_per_i2c_clk_g	RW	4000_2014h	[15]	Enable bif_per_i2c_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “bif_per_i2c_clk” is not gated by CM0 power state status 1: Clock “bif_per_i2c_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_bif_per_uart_clk_g	RW	4000_2014h	[16]	Enable bif_per_uart_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “bif_per_uart_clk” is not gated by CM0 power state status 1: Clock “bif_per_uart_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_dtimer1_clk_g	RW	4000_2014h	[17]	Enable dtimer1_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “dtimer1_clk” is not gated by CM0 power state status 1: Clock “dtimer1_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_dtimer2_clk_g	RW	4000_2014h	[18]	Enable dtimer2_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “dtimer2_clk” is not gated by CM0 power state status 1: Clock “dtimer2_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_dtimer3_clk_g	RW	4000_2014h	[19]	Enable dtimer3_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “dtimer3_clk” is not gated by CM0 power state status

Register name	Field name	Access	Address	Bits	Description
					1: Clock “dtimer3_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_wdt_clk_g	RW	4000_2014h	[20]	Enable wdt_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “wdt_clk” is not gated by CM0 power state status 1: Clock “wdt_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_gpio0_clk_g	RW	4000_2014h	[21]	Enable gpio0_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “gpio0_clk” is not gated by CM0 power state status 1: Clock “gpio0_clk” is gated when CM0 is in sleep state
CLK_SLEEP_MSK_CNFG	se_gpio1_clk_g	RW	4000_2014h	[22]	Enable gpio1_clk clock gating when Cortex®-M0 enters sleep state. 0: Clock “gpio1_clk” is not gated by CM0 power state status 1: Clock “gpio1_clk” is gated when CM0 is in sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_hosc_clk_g	RW	4000_2018h	[0]	Enable hosc_clk clock gating when Cortex®-M0 enters deep sleep state, if the hosc_clk clock gating control has been enabled. Warning: Only external reset assertion can remove clock gating if no external wakeup source has been enabled before! 0: Clock “hosc_clk” is not gated by CM0 power state status 1: Clock “hosc_clk” is gated when CM0 is in deep sleep if hosc_clk clock gating control has been enabled
CLK_DEEP_SLEEP_MSK_CNFG	dse_rom_clk_g	RW	4000_2018h	[1]	Enable rom_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “rom_clk” is not gated by CM0 power state status 1: Clock “rom_clk” is gated when CM0 is in deep sleep state

Register name	Field name	Access	Address	Bits	Description
CLK_DEEP_SLEEP_MSK_CNFG	dse_ram1_clk_g	RW	4000_2018h	[2]	Enable ram1_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “ram1_clk” is not gated by CM0 power state status 1: Clock “ram1_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_ram2_clk_g	RW	4000_2018h	[3]	Enable ram2_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “ram2_clk” is not gated by CM0 power state status 1: Clock “ram2_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_amba_clk_g	RW	4000_2018h	[4]	Enable amba_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “amba_clk” is not gated by CM0 power state status 1: Clock “amba_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_dma_clk_g	RW	4000_2018h	[5]	Enable dma_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “dma_clk” is not gated by CM0 power state status 1: Clock “dma_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_cnfg_otp1_w_clk1_g	RW	4000_2018h	[6]	Enable cnfg_otp1_w_clk1 clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “cnfg_otp1_w_clk1” is not gated by CM0 power state status 1: Clock “cnfg_otp1_w_clk1” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_cnfg_otp1_w_clk2_g	RW	4000_2018h	[7]	Enable cnfg_otp1_w_clk2 clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “cnfg_otp1_w_clk2” is not gated by CM0 power state status

Register name	Field name	Access	Address	Bits	Description
					1: Clock “cnfg_otp1_w_clk2” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_cnfg_dma_clk_g	RW	4000_2018h	[10]	Enable cnfg_dma_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “cnfg_dma_clk” is not gated by CM0 power state status 1: Clock “cnfg_dma_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_bif_reg_clk_g	RW	4000_2018h	[11]	Enable bif_reg_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “bif_reg_clk” is not gated by CM0 power state status 1: Clock “bif_reg_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_bif_per_svid_clk_g	RW	4000_2018h	[12]	Enable bif_per_svid_clk clock gating when Cortex®-M0 enters deep sleep state. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Clock “bif_per_svid_clk” is not gated by CM0 power state status 1: Clock “bif_per_svid_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_bif_per_pmbus_clk_g	RW	4000_2018h	[13]	Enable bif_per_pmbus_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_pmbus_clk” is not gated by CM0 power state status 1: Clock “bif_per_pmbus_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_bif_per_ssp_clk_g	RW	4000_2018h	[14]	Reserved
CLK_DEEP_SLEEP_MSK_CNFG	dse_bif_per_i2c_clk_g	RW	4000_2018h	[15]	Enable bif_per_i2c_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_i2c_clk” is not gated by CM0 power state status

Register name	Field name	Access	Address	Bits	Description
					1: Clock “bif_per_i2c_clk” is gated when CM0 is in deep sleep state.
CLK_DEEP_SLEEP_MSK_CNFG	dse_bif_per_uart_clk_g	RW	4000_2018h	[16]	Enable bif_per_uart_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_uart_clk” is not gated by CM0 power state status 1: Clock “bif_per_uart_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_dtimer1_clk_g	RW	4000_2018h	[17]	Enable dtimer1_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “dtimer1_clk” is not gated by CM0 power state status 1: Clock “dtimer1_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_dtimer2_clk_g	RW	4000_2018h	[18]	Enable dtimer2_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “dtimer2_clk” is not gated by CM0 power state status 1: Clock “dtimer2_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_dtimer3_clk_g	RW	4000_2018h	[19]	Enable dtimer3_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “dtimer3_clk” is not gated by CM0 power state status 1: Clock “dtimer3_clk” is gated when CM0 is in deep sleep state
CLK_DEEP_SLEEP_MSK_CNFG	dse_wdt_clk_g	RW	4000_2018h	[20]	Enable wdt_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “wdt_clk” is not gated by CM0 power state status 1: Clock “wdt_clk” is gated when CM0 is in deep sleep
CLK_DEEP_SLEEP_MSK_CNFG	dse_gpio0_clk_g	RW	4000_2018h	[21]	Enable gpio0_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “gpio0_clk” is not gated by CM0 power state status 1: Clock “gpio0_clk” is gated when CM0 is in deep sleep state

Register name	Field name	Access	Address	Bits	Description
CLK_DEEP_SLEEP_MSK_CNFG	dse_gpio1_clk_g	RW	4000_2018h	[22]	Enable gpio1_clk clock gating when Cortex®-M0 enters deep sleep state. 0: Clock “gpio1_clk” is not gated by CM0 power state status 1: Clock “gpio1_clk” is gated when CM0 is in deep sleep state
HOSC_SW_CLK_GATING_CTRL	kill_me_softly	RW	4000_201Ch	[0]	Enable hosc_clk SW clock gating. Note: The primary clock gating is performed if this bit is set and the hosc_clk_g bit of the CLK_EN_CTRL register is also set or if this bit is set and one of the sleep mask bit-0s are set when executing the WFI/WFE instruction. Be sure to enable an external wakeup source before executing the primary clock gating procedure (i.e., entering the hibernate state) in order to avoid permanent loss of the CPUS clock. 0: Clock “hosc_clk” cannot be gated 1: Clock “hosc_clk” can be gated
HOSC_HW_CLK_GATING_CTRL	kill_me_hardly	RW	4000_2020h	[0]	Enable hosc_clk HW clock gating by CPUS_EN signal. 0: Clock “hosc_clk” cannot be gated 1: Clock “hosc_clk” can be gated
SPARE_FF	SPARE_FF	RW	4000_2024h	[31:0]	Spare register
CLK_EN_CTRL_SET	hosc_clk_g	W	4000_2030h	[0]	Enable bit for the clock hosc_clk if the hosc_clk primary clock gating control has been enabled. Note: The primary clock gating is performed if this bit is set. KILL_ME_SOFTLY bit of the HOSC_SW_CLK_GATING_CTRL register is also set. Be sure to enable an external wakeup source before executing the primary clock gating procedure (i.e., entering the hibernate state) in order to avoid permanent loss of the CPUS clock. 0: Status of “hosc_clk” clock is not affected

Register name	Field name	Access	Address	Bits	Description
					1: Enable “hosc_clk” clock
CLK_EN_CTRL_SET	rom_clk_g	W	4000_2030h	[1]	Enable the rom_clk clock. 0: Status of “rom_clk” clock is not affected 1: Enable “rom_clk” clock
CLK_EN_CTRL_SET	ram1_clk_g	W	4000_2030h	[2]	Enable the ram1_clk clock. 0: Status of “ram1_clk” clock is not affected 1: Enable “ram1_clk” clock
CLK_EN_CTRL_SET	ram2_clk_g	W	4000_2030h	[3]	Enable the ram2_clk clock. 0: Status of ‘ram2_clk’ clock is not affected 1: Enable ‘ram2_clk’ clock
CLK_EN_CTRL_SET	amba_clk_g	W	4000_2030h	[4]	Enable the amba_clk clock. 0: Status of “amba_clk” clock is not affected 1: Enable “amba_clk” clock
CLK_EN_CTRL_SET	dma_clk_g	W	4000_2030h	[5]	Enable the dma_clk clock. 0: Status of “dma_clk” clock is not affected 1: Enable “dma_clk” clock
CLK_EN_CTRL_SET	cnfg_otp1_w_clk_g	W	4000_2030h	[6]	Enable the cnfg_otp1_w_clk clock. 0: Status of “cnfg_otp1_w_clk” clock is not affected 1: Enable “cnfg_otp1_w_clk” clock
CLK_EN_CTRL_SET	cnfg_dma_clk_g	W	4000_2030h	[10]	Enable the cnfg_dma_clk clock. 0: Status of “cnfg_dma_clk” clock is not affected 1: Enable “cnfg_dma_clk” clock
CLK_EN_CTRL_SET	bif_reg_clk_g	W	4000_2030h	[11]	Enable the bif_reg_clk clock. 0: Status of “bif_reg_clk” clock is not affected 1: Enable “bif_reg_clk” clock
CLK_EN_CTRL_SET	bif_per_svid_clk_g	W	4000_2030h	[12]	Enable the bif_per_svid_clk clock. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Status of “bif_per_svid_clk” clock is not affected 1: Enable “bif_per_svid_clk” clock

Register name	Field name	Access	Address	Bits	Description
CLK_EN_CTRL_SET	bif_per_pmbus_clk_g	W	4000_2030h	[13]	Enable the bif_per_pmbus_clk clock. 0: Status of “bif_per_pmbus_clk” clock is not affected 1: Enable “bif_per_pmbus_clk” clock
CLK_EN_CTRL_SET	bif_per_ssp_clk_g	W	4000_2030h	[14]	Reserved
CLK_EN_CTRL_SET	bif_per_i2c_clk_g	W	4000_2030h	[15]	Enable the bif_per_i2c_clk clock. 0: Status of “bif_per_i2c_clk” clock is not affected 1: Enable “bif_per_i2c_clk” clock
CLK_EN_CTRL_SET	bif_per_uart_clk_g	W	4000_2030h	[16]	Enable the bif_per_uart_clk clock. 0: Status of “bif_per_uart_clk” clock is not affected 1: Enable “bif_per_uart_clk” clock
CLK_EN_CTRL_SET	dtimer1_clk_g	W	4000_2030h	[17]	Enable the dtimer1_clk clock. 0: Status of ‘dtimer1_clk’ clock is not affected 1: Enable ‘dtimer1_clk’ clock
CLK_EN_CTRL_SET	dtimer2_clk_g	W	4000_2030h	[18]	Enable the dtimer2_clk clock. 0: Status of “dtimer2_clk” clock is not affected 1: Enable “dtimer2_clk” clock
CLK_EN_CTRL_SET	dtimer3_clk_g	W	4000_2030h	[19]	Enable the dtimer3_clk clock. 0: Status of “dtimer3_clk” clock is not affected 1: Enable “dtimer3_clk” clock
CLK_EN_CTRL_SET	wdt_clk_g	W	4000_2030h	[20]	Enable the wdt_clk clock. 0: Status of “wdt_clk” clock is not affected 1: Enable “wdt_clk” clock
CLK_EN_CTRL_SET	gpio0_clk_g	W	4000_2030h	[21]	Enable the gpio0_clk clock. 0: Status of “gpio0_clk” clock is not affected 1: Enable “gpio0_clk” clock
CLK_EN_CTRL_SET	gpio1_clk_g	W	4000_2030h	[22]	Enable the gpio1_clk clock. 0: Status of “gpio1_clk” clock is not affected 1: Enable “gpio1_clk” clock
CLK_EN_CTRL_SET	dtimer1_kernel_clk_g	W	4000_2030h	[23]	Enable the dtimer1_kernel_clk clock.

Register name	Field name	Access	Address	Bits	Description
					0: Status of “dtimer1_kernel_clk” clock is not affected 1: Enable “dtimer1_kernel_clk” clock
CLK_EN_CTRL_SET	dtimer2_kernel_clk_g	W	4000_2030h	[24]	Enable the dtimer2_kernel_clk clock. 0: Status of “dtimer2_kernel_clk” clock is not affected 1: Enable “dtimer2_kernel_clk” clock
CLK_EN_CTRL_SET	dtimer3_kernel_clk_g	W	4000_2030h	[25]	Enable the dtimer3_kernel_clk clock. 0: Status of “dtimer3_kernel_clk” clock is not affected 1: Enable “dtimer3_kernel_clk” clock
CLK_EN_CTRL_SET	pmbus_kernel_clk_g	W	4000_2030h	[26]	Enable the pmbus_kernel_clk clock. 0: Status of “pmbus_kernel_clk” clock is not affected 1: Enable “pmbus_kernel_clk” clock
CLK_EN_CTRL_SET	otp_kernel_clk_g	W	4000_2030h	[27]	Enable the otp_kernel_clk clock. 0: Status of “otp_kernel_clk” clock is not affected 1: Enable “otp_kernel_clk” clock
CLK_EN_CTRL_CLR	hosc_clk_g	W	4000_2034h	[0]	Disable bit for the clock hosc_clk if the hosc_clk primary clock gating control has been enabled. Note: The primary clock gating is performed if this bit is set. KILL_ME_SOFTLY bit of the HOSC_SW_CLK_GATING_CTRL register is also set. Be sure to enable an external wakeup source before executing the primary clock gating procedure (i.e., entering the hibernate state) in order to avoid permanent loss of the CPUS clock. 0: Status of “hosc_clk” clock is not affected 1: Disable “hosc_clk” clock
CLK_EN_CTRL_CLR	rom_clk_g	W	4000_2034h	[1]	Disable the rom_clk clock.

Register name	Field name	Access	Address	Bits	Description
					0: Status of “rom_clk” clock is not affected 1: Disable “rom_clk” clock
CLK_EN_CTRL_CLR	ram1_clk_g	W	4000_2034h	[2]	Disable the ram1_clk clock. 0: Status of ‘ram1_clk’ clock is not affected 1: Disable ‘ram1_clk’ clock
CLK_EN_CTRL_CLR	ram2_clk_g	W	4000_2034h	[3]	Disable the ram2_clk clock. 0: Status of “ram2_clk” clock is not affected 1: Disable “ram2_clk” clock
CLK_EN_CTRL_CLR	amba_clk_g	W	4000_2034h	[4]	Disable the amba_clk clock. 0: Status of “amba_clk” clock is not affected 1: Disable “amba_clk” clock
CLK_EN_CTRL_CLR	dma_clk_g	W	4000_2034h	[5]	Disable the dma_clk clock. 0: Status of “dma_clk” clock is not affected 1: Disable “dma_clk” clock
CLK_EN_CTRL_CLR	cnfg_otp1_w_clk_g	W	4000_2034h	[6]	Disable the cnfg_otp1_w_clk clock. 0: Status of “cnfg_otp1_w_clk” clock is not affected 1: Disable “cnfg_otp1_w_clk” clock
CLK_EN_CTRL_CLR	cnfg_dma_clk_g	W	4000_2034h	[10]	Disable the cnfg_dma_clk clock. 0: Status of “cnfg_dma_clk” clock is not affected 1: Disable “cnfg_dma_clk” clock
CLK_EN_CTRL_CLR	bif_reg_clk_g	W	4000_2034h	[11]	Disable the bif_reg_clk clock. 0: Status of “bif_reg_clk” clock is not affected 1: Disable “bif_reg_clk” clock
CLK_EN_CTRL_CLR	bif_per_svid_clk_g	W	4000_2034h	[12]	Disable the bif_per_svid_clk clock. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Status of “bif_per_svid_clk” clock is not affected 1: Disable “bif_per_svid_clk” clock
CLK_EN_CTRL_CLR	bif_per_pmbus_clk_g	W	4000_2034h	[13]	Disable the bif_per_pmbus_clk clock.

Register name	Field name	Access	Address	Bits	Description
					0: Status of “bif_per_pmbus_clk” clock is not affected 1: Disable “bif_per_pmbus_clk” clock
CLK_EN_CTRL_CLR	bif_per_ssp_clk_g	W	4000_2034h	[14]	Reserved
CLK_EN_CTRL_CLR	bif_per_i2c_clk_g	W	4000_2034h	[15]	Disable the bif_per_i2c_clk clock. 0: Status of “bif_per_i2c_clk” clock is not affected 1: Disable “bif_per_i2c_clk” clock
CLK_EN_CTRL_CLR	bif_per_uart_clk_g	W	4000_2034h	[16]	Disable the bif_per_uart_clk clock. 0: Status of “bif_per_uart_clk” clock is not affected 1: Disable “bif_per_uart_clk” clock
CLK_EN_CTRL_CLR	dtimer1_clk_g	W	4000_2034h	[17]	Disable the dtimer1_clk clock. 0: Status of “dtimer1_clk” clock is not affected 1: Disable “dtimer1_clk” clock
CLK_EN_CTRL_CLR	dtimer2_clk_g	W	4000_2034h	[18]	Disable the dtimer2_clk clock. 0: Status of “dtimer2_clk” clock is not affected 1: Disable “dtimer2_clk” clock
CLK_EN_CTRL_CLR	dtimer3_clk_g	W	4000_2034h	[19]	Disable the dtimer3_clk clock. 0: Status of “dtimer3_clk” clock is not affected 1: Disable “dtimer3_clk” clock
CLK_EN_CTRL_CLR	wdt_clk_g	W	4000_2034h	[20]	Disable the wdt_clk clock. 0: Status of “wdt_clk” clock is not affected 1: Disable “wdt_clk” clock
CLK_EN_CTRL_CLR	gpio0_clk_g	W	4000_2034h	[21]	Disable the gpio0_clk clock. 0: Status of “gpio0_clk” clock is not affected 1: Disable “gpio0_clk” clock
CLK_EN_CTRL_CLR	gpio1_clk_g	W	4000_2034h	[22]	Disable the gpio1_clk clock. 0: Status of “gpio1_clk” clock is not affected 1: Disable “gpio1_clk” clock
CLK_EN_CTRL_CLR	dtimer1_kernel_clk_g	W	4000_2034h	[23]	Disable the dtimer1_kernel_clk clock. 0: Status of “dtimer1_kernel_clk” clock is not affected

Register name	Field name	Access	Address	Bits	Description
					1: Disable “dtimer1_kernel_clk” clock
CLK_EN_CTRL_CLR	dtimer2_kernel_clk_g	W	4000_2034h	[24]	Disable the dtimer2_kernel_clk clock. 0: Status of “dtimer2_kernel_clk” clock is not affected 1: Disable “dtimer2_kernel_clk” clock
CLK_EN_CTRL_CLR	dtimer3_kernel_clk_g	W	4000_2034h	[25]	Disable the dtimer3_kernel_clk clock. 0: Status of “dtimer3_kernel_clk” clock is not affected 1: Disable “dtimer3_kernel_clk” clock
CLK_EN_CTRL_CLR	pmbus_kernel_clk_g	W	4000_2034h	[26]	Disable the pmbus_kernel_clk clock. 0: Status of “pmbus_kernel_clk” clock is not affected 1: Disable “pmbus_kernel_clk” clock
CLK_EN_CTRL_CLR	otp_kernel_clk_g	W	4000_2034h	[27]	Disable the otp_kernel_clk clock. 0: Status of “otp_kernel_clk” clock is not affected 1: Disable “otp_kernel_clk” clock
CLK_SLEEP_MSK_CNFG_SET	se_hosc_clk_g	W	4000_2038h	[0]	Enable hosc_clk clock gating when Cortex®-M0 will enter sleep state, if the hosc_clk clock gating control has been enabled. Warning: Only external reset assertion can remove clock gating if no external wakeup source has been enabled before. 0: Clock “hosc_clk” sleep state clock gating status unchanged 1: Enable clock “hosc_clk” sleep state gating if hosc_clk clock gating control has been enabled
CLK_SLEEP_MSK_CNFG_SET	se_rom_clk_g	W	4000_2038h	[1]	Enable rom_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “rom_clk” sleep state clock gating status unchanged 1: Enable clock “rom_clk” sleep state clock gating

Register name	Field name	Access	Address	Bits	Description
CLK_SLEEP_MSK_CNFG_SET	se_ram1_clk_g	W	4000_2038h	[2]	Enable ram1_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “ram1_clk” sleep state clock gating status unchanged 1: Enable clock “ram1_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_ram2_clk_g	W	4000_2038h	[3]	Enable ram2_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “ram2_clk” sleep state clock gating status unchanged 1: Enable clock “ram2_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_amba_clk_g	W	4000_2038h	[4]	Enable amba_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “amba_clk” sleep state clock gating status unchanged 1: Enable clock “amba_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_dma_clk_g	W	4000_2038h	[5]	Enable dma_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dma_clk” sleep state clock gating status unchanged 1: Enable clock “dma_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_cnfg_otp1_w_clk_g	W	4000_2038h	[6]	Enable cnfg_otp1_w_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “cnfg_otp1_w_clk” sleep state clock gating status unchanged 1: Enable clock “cnfg_otp1_w_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_cnfg_dma_clk_g	W	4000_2038h	[10]	Enable cnfg_dma_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “cnfg_dma_clk” sleep state clock gating status unchanged 1: Enable clock “cnfg_dma_clk” sleep state clock gating

Register name	Field name	Access	Address	Bits	Description
CLK_SLEEP_MSK_CNFG_SET	se_bif_reg_clk_g	W	4000_2038h	[11]	Enable bif_reg_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_reg_clk” sleep state clock gating status unchanged 1: Enable clock “bif_reg_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_bif_per_svid_clk_g	W	4000_2038h	[12]	Enable bif_per_svid_clk clock gating when the Cortex®-M0 enters sleep state. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Clock “bif_per_svid_clk” sleep state clock gating status unchanged 1: Enable clock “bif_per_svid_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_bif_per_pmbus_clk_g	W	4000_2038h	[13]	Enable bif_per_pmbus_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_per_pmbus_clk” sleep state clock gating status unchanged 1: Enable clock “bif_per_pmbus_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_bif_per_ssp_clk_g	W	4000_2038h	[14]	Reserved
CLK_SLEEP_MSK_CNFG_SET	se_bif_per_i2c_clk_g	W	4000_2038h	[15]	Enable bif_per_i2c_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_per_i2c_clk” sleep state clock gating status unchanged 1: Enable clock “bif_per_i2c_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_bif_per_uart_clk_g	W	4000_2038h	[16]	Enable bif_per_uart_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_per_uart_clk” sleep state clock gating status unchanged 1: Enable clock “bif_per_uart_clk” sleep state clock gating

Register name	Field name	Access	Address	Bits	Description
CLK_SLEEP_MSK_CNFG_SET	se_dtimer1_clk_g	W	4000_2038h	[17]	Enable dtimer1_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dtimer1_clk” sleep state clock gating status unchanged 1: Enable clock “dtimer1_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_dtimer2_clk_g	W	4000_2038h	[18]	Enable dtimer2_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dtimer2_clk” sleep state clock gating status unchanged 1: Enable clock “dtimer2_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_dtimer3_clk_g	W	4000_2038h	[19]	Enable dtimer3_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dtimer3_clk” sleep state clock gating status unchanged 1: Enable clock “dtimer3_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_wdt_clk_g	W	4000_2038h	[20]	Enable wdt_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “wdt_clk” sleep state clock gating status unchanged 1: Enable clock “wdt_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_gpio0_clk_g	W	4000_2038h	[21]	Enable gpio2_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “gpio2_clk” sleep state clock gating status unchanged 1: Enable clock “gpio2_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_SET	se_gpio1_clk_g	W	4000_2038h	[22]	Enable gpio1_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “gpio1_clk” sleep state clock gating status unchanged 1: Enable clock “gpio1_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_hosc_clk_g	W	4000_203Ch	[0]	Disable hosc_clk clock gating when Cortex®-M0 will enter sleep state, if the hosc_clk clock gating control has been enabled.

Register name	Field name	Access	Address	Bits	Description
					Warning: Only external reset assertion can remove clock gating if no external wakeup source has been enabled before. 0: Clock “hosc_clk” sleep state clock gating status unchanged 1: Disable clock “hosc_clk” sleep state gating
CLK_SLEEP_MSK_CNFG_CLR	se_rom_clk_g	W	4000_203Ch	[1]	Disable rom_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “rom_clk” sleep state clock gating status unchanged 1: Disable clock “rom_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_ram1_clk_g	W	4000_203Ch	[2]	Disable ram1_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “ram1_clk” sleep state clock gating status unchanged 1: Disable clock “ram1_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_ram2_clk_g	W	4000_203Ch	[3]	Disable ram2_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “ram2_clk” sleep state clock gating status unchanged 1: Disable clock “ram2_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_amba_clk_g	W	4000_203Ch	[4]	Disable amba_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “amba_clk” sleep state clock gating status unchanged 1: Disable clock “amba_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_dma_clk_g	W	4000_203Ch	[5]	Disable dma_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dma_clk” sleep state clock gating status unchanged 1: Disable clock “dma_clk” sleep state clock gating

Register name	Field name	Access	Address	Bits	Description
CLK_SLEEP_MSK_CNFG_CLR	se_cnfg_otp1_w_clk_g	W	4000_203Ch	[6]	Disable cnfg_otp1_w_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “cnfg_otp1_w_clk” sleep state clock gating status unchanged 1: Disable clock “cnfg_otp1_w_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_cnfg_dma_clk_g	W	4000_203Ch	[10]	Disable cnfg_dma_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “cnfg_dma_clk” sleep state clock gating status unchanged 1: Disable clock “cnfg_dma_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_bif_reg_clk_g	W	4000_203Ch	[11]	Disable bif_reg_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_reg_clk” sleep state clock gating status unchanged 1: Disable clock “bif_reg_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_bif_per_svid_clk_g	W	4000_203Ch	[12]	Disable bif_per_svid_clk clock gating when the Cortex®-M0 enters sleep state. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Clock “bif_per_svid_clk” sleep state clock gating status unchanged 1: Disable clock “bif_per_svid_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_bif_per_pmbus_clk_g	W	4000_203Ch	[13]	Disable bif_per_pmbus_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_per_pmbus_clk” sleep state clock gating status unchanged 1: Disable clock “bif_per_pmbus_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_bif_per_ssp_clk_g	W	4000_203Ch	[14]	Reserved

Register name	Field name	Access	Address	Bits	Description
CLK_SLEEP_MSK_CNFG_CLR	se_bif_per_i2c_clk_g	W	4000_203Ch	[15]	Disable bif_per_i2c_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_per_i2c_clk” sleep state clock gating status unchanged 1: Disable clock “bif_per_i2c_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_bif_per_uart_clk_g	W	4000_203Ch	[16]	Disable bif_per_uart_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “bif_per_uart_clk” sleep state clock gating status unchanged 1: Disable clock “bif_per_uart_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_dtimer1_clk_g	W	4000_203Ch	[17]	Disable dtimer1_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dtimer1_clk” sleep state clock gating status unchanged 1: Disable clock “dtimer1_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_dtimer2_clk_g	W	4000_203Ch	[18]	Disable dtimer2_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dtimer2_clk” sleep state clock gating status unchanged 1: Disable clock “dtimer2_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_dtimer3_clk_g	W	4000_203Ch	[19]	Disable dtimer3_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “dtimer3_clk” sleep state clock gating status unchanged 1: Disable clock “dtimer3_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_wdt_clk_g	W	4000_203Ch	[20]	Disable wdt_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “wdt_clk” sleep state clock gating status unchanged 1: Disable clock “wdt_clk” sleep state clock gating

Register name	Field name	Access	Address	Bits	Description
CLK_SLEEP_MSK_CNFG_CLR	se_gpio0_clk_g	W	4000_203Ch	[21]	Disable gpio2_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “gpio2_clk” sleep state clock gating status unchanged 1: Disable clock “gpio2_clk” sleep state clock gating
CLK_SLEEP_MSK_CNFG_CLR	se_gpio1_clk_g	W	4000_203Ch	[22]	Disable gpio1_clk clock gating when the Cortex®-M0 enters sleep state. 0: Clock “gpio1_clk” sleep state clock gating status unchanged 1: Disable clock “gpio1_clk” sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_hosc_clk_g	W	4000_2040h	[0]	Enable hosc_clk clock gating when Cortex®-M0 will enter the deep sleep state, if the hosc_clk clock gating control has been enabled. Warning: Only external reset assertion can remove clock gating if no external wakeup source has been enabled before. 0: Clock “hosc_clk” deep sleep state clock gating status unchanged 1: Enable clock “hosc_clk” deep sleep state gating if hosc_clk clock gating control has been enabled
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_rom_clk_g	W	4000_2040h	[1]	Enable rom_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “rom_clk” deep sleep state clock gating status unchanged 1: Enable clock “rom_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_ram1_clk_g	W	4000_2040h	[2]	Enable ram1_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “ram1_clk” deep sleep state clock gating status unchanged 1: Enable clock “ram1_clk” deep sleep state clock gating

Register name	Field name	Access	Address	Bits	Description
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_ram2_clk_g	W	4000_2040h	[3]	Enable ram2_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “ram2_clk” deep sleep state clock gating status unchanged 1: Enable clock “ram2_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_amba_clk_g	W	4000_2040h	[4]	Enable amba_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “amba_clk” deep sleep state clock gating status unchanged 1: Enable clock “amba_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_dma_clk_g	W	4000_2040h	[5]	Enable dma_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “dma_clk” deep sleep state clock gating status unchanged 1: Enable clock “dma_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_cnfg_otp1_w_clk_g	W	4000_2040h	[6]	Enable cnfg_otp1_w_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “cnfg_otp1_w_clk” deep sleep state clock gating status unchanged 1: Enable clock “cnfg_otp1_w_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_cnfg_dma_clk_g	W	4000_2040h	[10]	Enable cnfg_dma_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “cnfg_dma_clk” deep sleep state clock gating status unchanged 1: Enable clock “cnfg_dma_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_bif_reg_clk_g	W	4000_2040h	[11]	Enable bif_reg_clk clock gating when the Cortex®-M0 enters deep sleep state.

Register name	Field name	Access	Address	Bits	Description
					0: Clock “bif_reg_clk” deep sleep state clock gating status unchanged 1: Enable clock “bif_reg_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_bif_per_svid_clk_g	W	4000_2040h	[12]	Enable bif_per_svid_clk clock gating when the Cortex®-M0 enters deep sleep state. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock. 0: Clock “bif_per_svid_clk” deep sleep state clock gating status unchanged 1: Enable clock “bif_per_svid_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_bif_per_pmbus_clk_g	W	4000_2040h	[13]	Enable bif_per_pmbus_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_pmbus_clk” deep sleep state clock gating status unchanged 1: Enable clock “bif_per_pmbus_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_bif_per_ssp_clk_g	W	4000_2040h	[14]	Reserved
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_bif_per_i2c_clk_g	W	4000_2040h	[15]	Enable bif_per_i2c_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_i2c_clk” deep sleep state clock gating status unchanged 1: Enable clock “bif_per_i2c_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_bif_per_uart_clk_g	W	4000_2040h	[16]	Enable bif_per_uart_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_uart_clk” deep sleep state clock gating status unchanged 1: Enable clock “bif_per_uart_clk” deep sleep state clock gating

Register name	Field name	Access	Address	Bits	Description
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_dtimer1_clk_g	W	4000_2040h	[17]	Enable dtimer1_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “dtimer1_clk” deep sleep state clock gating status unchanged 1: Enable clock “dtimer1_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_dtimer2_clk_g	W	4000_2040h	[18]	Enable dtimer2_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “dtimer2_clk” deep sleep state clock gating status unchanged 1: Enable clock “dtimer2_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_dtimer3_clk_g	W	4000_2040h	[19]	Enable dtimer3_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “dtimer3_clk” deep sleep state clock gating status unchanged 1: Enable clock “dtimer3_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_wdt_clk_g	W	4000_2040h	[20]	Enable wdt_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “wdt_clk” deep sleep state clock gating status unchanged 1: Enable clock “wdt_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_gpio0_clk_g	W	4000_2040h	[21]	Enable gpio2_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “gpio2_clk” deep sleep state clock gating status unchanged 1: Enable clock “gpio2_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_SET	dse_gpio1_clk_g	W	4000_2040h	[22]	Enable gpio1_clk clock gating when the Cortex®-M0 enters deep sleep state.

Register name	Field name	Access	Address	Bits	Description
					0: Clock “gpio1_clk” deep sleep state clock gating status unchanged 1: Enable clock “gpio1_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_hosc_clk_g	W	4000_2044h	[0]	Disable hosc_clk clock gating when Cortex®-M0 will enter the deep sleep state, if the hosc_clk clock gating control has been enabled. Warning: Only external reset assertion can remove clock gating if no external wakeup source has been enabled before. 0: Clock “hosc_clk” deep sleep state clock gating status unchanged 1: Disable clock “hosc_clk” deep sleep state gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_rom_clk_g	W	4000_2044h	[1]	Disable rom_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “rom_clk” deep sleep state clock gating status unchanged 1: Disable clock “rom_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_ram1_clk_g	W	4000_2044h	[2]	Disable ram1_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “ram1_clk” deep sleep state clock gating status unchanged 1: Disable clock “ram1_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_ram2_clk_g	W	4000_2044h	[3]	Disable ram2_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “ram2_clk” deep sleep state clock gating status unchanged 1: Disable clock “ram2_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_amba_clk_g	W	4000_2044h	[4]	Disable amba_clk clock gating when the Cortex®-M0 enters deep sleep state.

Register name	Field name	Access	Address	Bits	Description
					0: Clock “amba_clk” deep sleep state clock gating status unchanged 1: Disable clock “amba_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_dma_clk_g	W	4000_2044h	[5]	Disable dma_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “dma_clk” deep sleep state clock gating status unchanged 1: Disable clock “dma_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_cnfg_otp1_w_clk_g	W	4000_2044h	[6]	Disable cnfg_otp1_w_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “cnfg_otp1_w_clk” deep sleep state clock gating status unchanged 1: Disable clock “cnfg_otp1_w_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_cnfg_dma_clk_g	W	4000_2044h	[10]	Disable cnfg_dma_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “cnfg_dma_clk” deep sleep state clock gating status unchanged 1: Disable clock “cnfg_dma_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_bif_reg_clk_g	W	4000_2044h	[11]	Disable bif_reg_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “bif_reg_clk” deep sleep state clock gating status unchanged 1: Disable clock “bif_reg_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_bif_per_svid_clk_g	W	4000_2044h	[12]	Disable bif_per_svid_clk clock gating when the Cortex®-M0 enters deep sleep state. Note: The CRC32 peripheral is using the SVID slot so this actually controls the CRC32 clock.

Register name	Field name	Access	Address	Bits	Description
					0: Clock “bif_per_svid_clk” deep sleep state clock gating status unchanged 1: Disable clock “bif_per_svid_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_bif_per_pmbus_clk_g	W	4000_2044h	[13]	Disable bif_per_pmbus_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_pmbus_clk” deep sleep state clock gating status unchanged 1: Disable clock “bif_per_pmbus_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_bif_per_ssp_clk_g	W	4000_2044h	[14]	Reserved
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_bif_per_i2c_clk_g	W	4000_2044h	[15]	Disable bif_per_i2c_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_i2c_clk” deep sleep state clock gating status unchanged 1: Disable clock “bif_per_i2c_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_bif_per_uart_clk_g	W	4000_2044h	[16]	Disable bif_per_uart_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “bif_per_uart_clk” deep sleep state clock gating status unchanged 1: Disable clock “bif_per_uart_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_dtimer1_clk_g	W	4000_2044h	[17]	Disable dtimer1_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “dtimer1_clk” deep sleep state clock gating status unchanged 1: Disable clock “dtimer1_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_dtimer2_clk_g	W	4000_2044h	[18]	Disable dtimer2_clk clock gating when the Cortex®-M0 enters deep sleep state.

Register name	Field name	Access	Address	Bits	Description
					0: Clock “dtimer2_clk” deep sleep state clock gating status unchanged 1: Disable clock “dtimer2_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_dtimer3_clk_g	W	4000_2044h	[19]	Disable dtimer3_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “dtimer3_clk” deep sleep state clock gating status unchanged 1: Disable clock “dtimer3_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_wdt_clk_g	W	4000_2044h	[20]	Disable wdt_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “wdt_clk” deep sleep state clock gating status unchanged 1: Disable clock “wdt_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_gpio0_clk_g	W	4000_2044h	[21]	Disable gpio2_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “gpio2_clk” deep sleep state clock gating status unchanged 1: Disable clock “gpio2_clk” deep sleep state clock gating
CLK_DEEP_SLEEP_MSK_CNFG_CLR	dse_gpio1_clk_g	W	4000_2044h	[22]	Disable gpio1_clk clock gating when the Cortex®-M0 enters deep sleep state. 0: Clock “gpio1_clk” deep sleep state clock gating status unchanged 1: Disable clock “gpio1_clk” deep sleep state clock gating

15.3.3 Reset generator unit

The reset generator unit (RGU) implements the reset scheme of the CPUS. The CPUS reset scheme is shown in Figure 103.

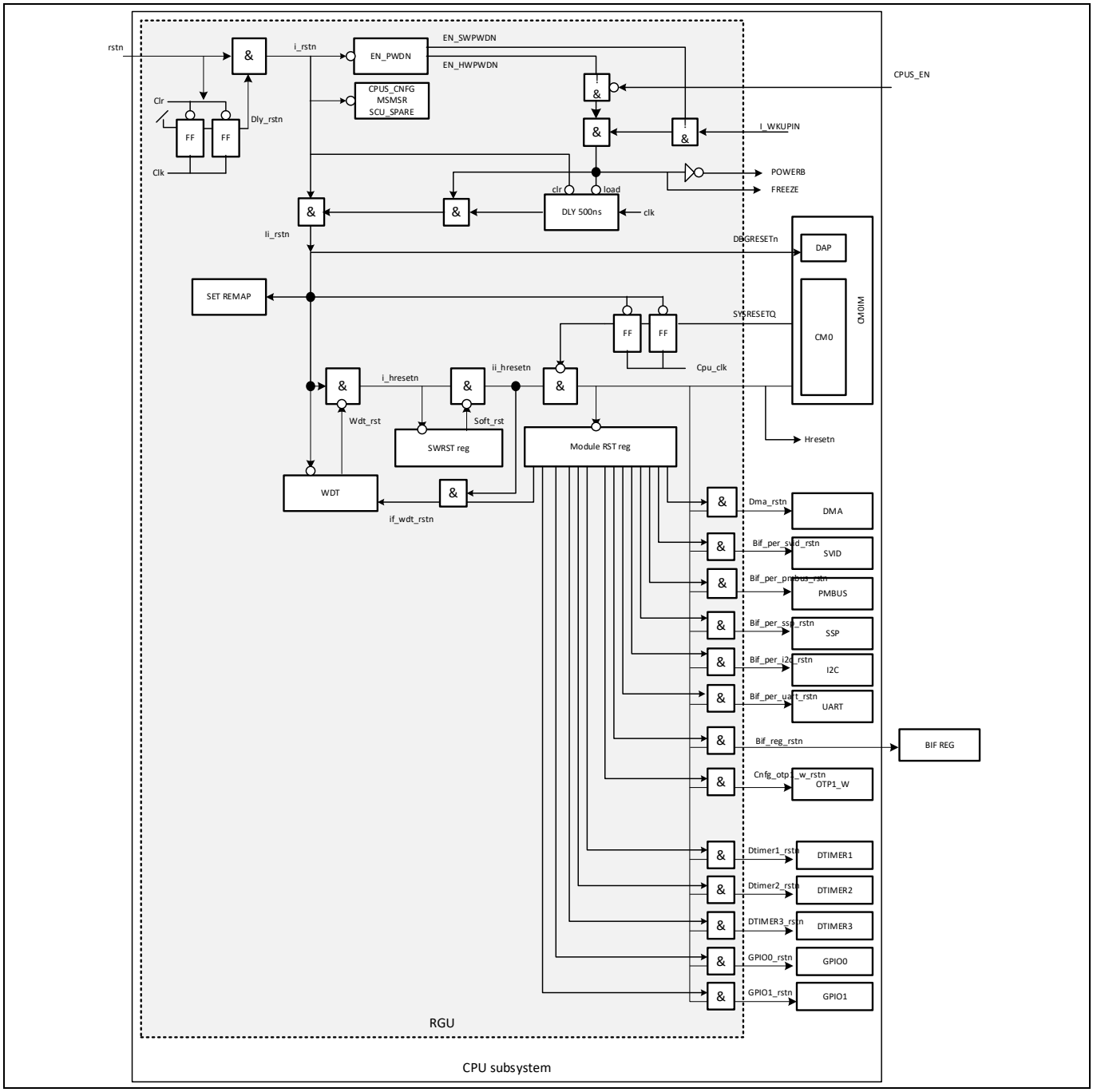

Figure 103 CPUS reset tree

15.3.3.1 Reset sources

The CPUS supports multiple reset sources:

- External reset
- CPUS_EN reset
- SW_PWDN reset
- WDT reset
- CPU system reset
- Soft reset
- Peripheral modules reset

The external reset signal (RSTN) is used to generate the root CPUS signal (II_RSTN), the PWRONB signal, to put the RAM and ROM in power-down, and the FREEZE clock gating signal, to gate the primary input source clock, hosc_clk (Figure 104).

HW_PWD and SW_PWD signals are able to trigger, internally to CPUS, a logic reset state (II_RSTN), with RAMs and ROM in power-down (PWRONB) and input clock hosc_clk gated (FREEZE).

In particular, to guarantee proper hold-time delay to PWRONB signal deassertion, the II_RSTN signal deassertion is stretched for at least 500 ns (more than 50 hosc_clk cycles at 100 MHz).

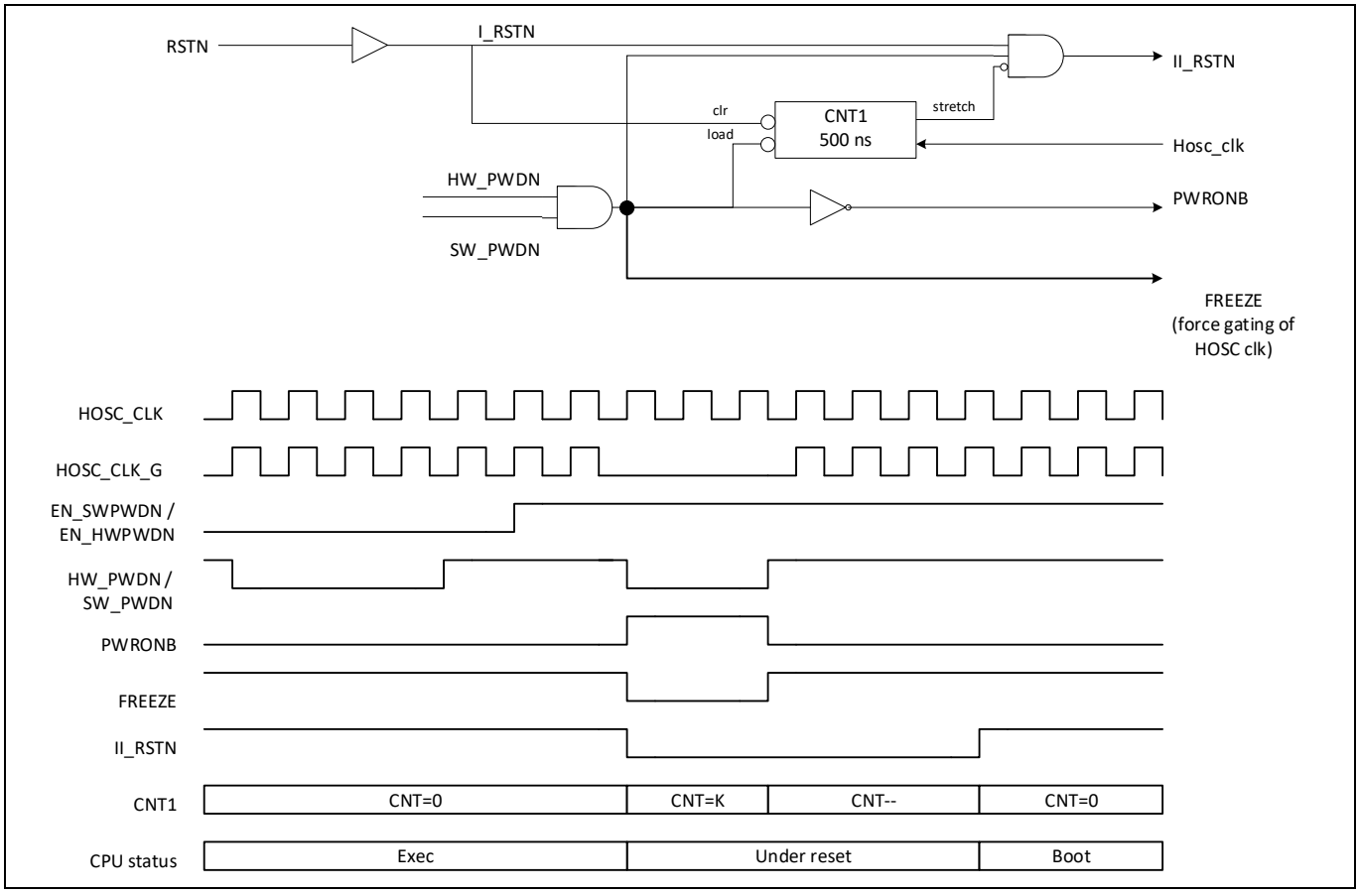

Figure 104 Reset generation scheme

An assertion of the SYSTRESETQ signal by Cortex®-M0 (debugger reset request) will cause the CPUS to be initialized, with the exception of the debugger section and WDT functionality. So the debugger will not be disconnected.

The peripheral reset signals allow the possibility to individually reset the CPU subsystem peripherals or ports (with the exception of the WDT kernel). These are asserted/deasserted by FW by setting appropriate bits in the register RSTMODS.

15.3.3.2 Software power-down

The CPUS can enter a power-down reset state by executing a proper FW instruction (WFE/WFI).

FW power-down needs to be enabled by EN_SWPWDN in the SW_PWDN_CTRL register.

Wakeup is performed by asserting the WKUP_IN signal or the RSTN signal. The proper wakeup source needs to be selected on the SCU CPUS_CFG register.

The software power-down and wakeup logic timing diagram is shown in Figure 105.

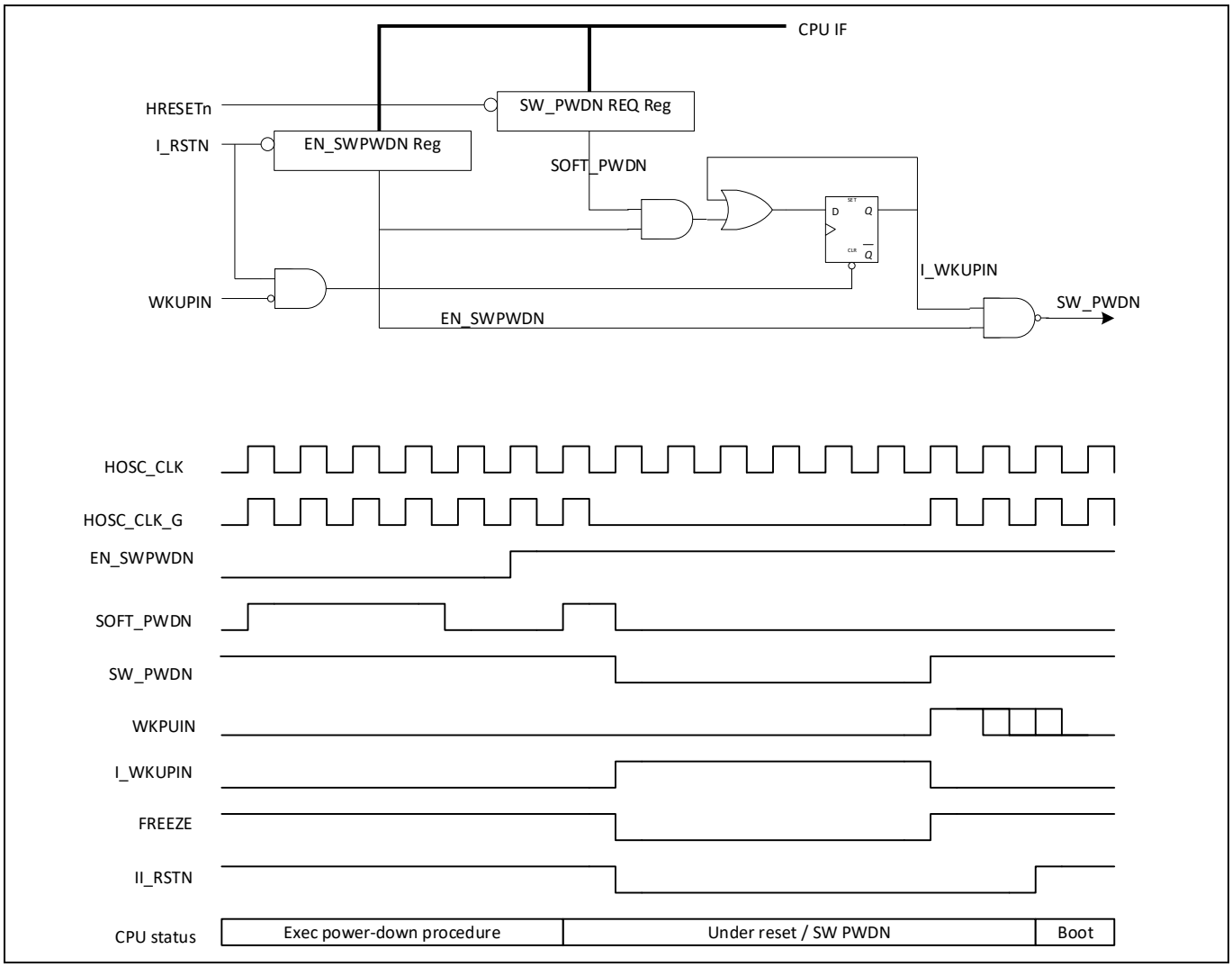

Figure 105 Software power-down and wakeup

15.3.3.3 Software reset

The CPUS software reset is invoked by setting the SWRST bit in the RSTMODS register. To perform the reset, the EN_SWRES bit in SWRST_CTRL register must be set first (enabling CPUS software reset execution), then a SWRST set bit in the RSTMODS register will cause a CPUS reset (Figure 106). When the soft reset procedure is terminated, bit SWRST in register RSTSR (reset status register) is set, indicating that the last reset was a soft reset.

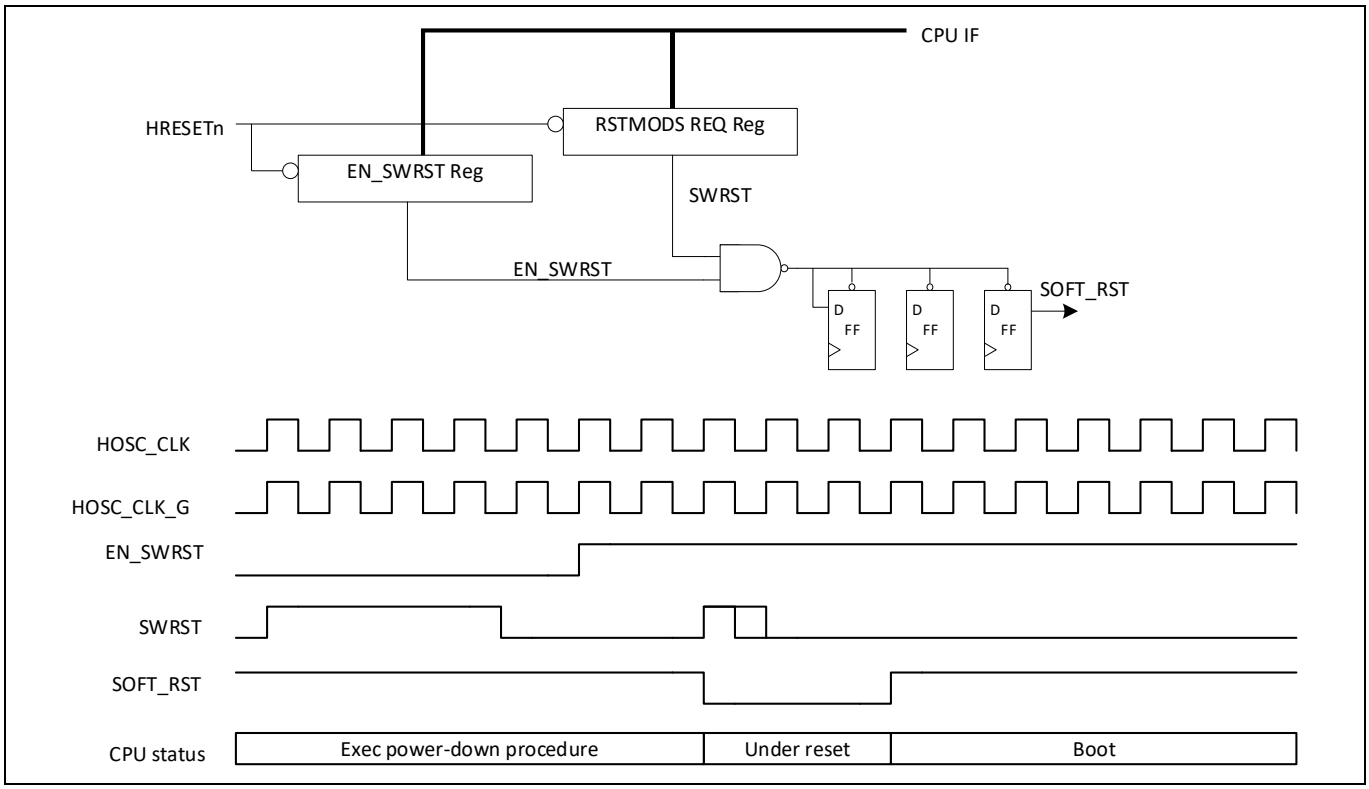

Figure 106 Software reset signal generation

15.3.3.4 RGU registers

The relevant RGU-related registers and their descriptions are provided in Table 96.

Table 96 RGU-related register descriptions

Register name	Field name	Access	Address	Bits	Description
RSTSR	SWPWDNRST	R	4000_1000h	[0]	SW power-down reset flag. 0: The last reset was not generated by SW_PWDN 1: The last reset was generated by SW_PWDN
RSTSR	HWPWDNRST	R	4000_1000h	[7]	HW power-down reset flag. 0: The last reset was not generated by HW_PWDN 1: The last reset was generated by HW_PWDN
RSTSR	SYSRST	R	4000_1000h	[15]	Cortex®-M0 SYSRESETQ flag.

Register name	Field name	Access	Address	Bits	Description
					0: The last reset was not generated by SYSRESETQ 1: The last reset was generated by SYSRESETQ
RSTSR	SWRST	R	4000_1000h	[23]	Soft reset flag. 0: The last reset was not a soft reset 1: The last reset was a soft reset
RSTSR	WDTRST	R	4000_1000h	[31]	Watchdog reset flag. 0: The last reset was not a watchdog reset 1: The last reset was a watchdog reset.
SWRST_CTRL	EN_SWRST	RW	4000_1004h	[0]	The software reset control register is used to enable the CPU to reset the CPUS by issuing a soft reset. A soft reset is triggered by setting the SWRST bit into the RSTMODS register. 0: CPUS cannot be reset by SWRST bit 1: CPUS can be reset by SWRST bit
RSTMODS	SWRST	RW	4000_1008h	[0]	CPUS software reset. 0: Normal operation 1: Perform software reset (also requires EN_SWRST=1) Asserting SWRST causes the CPUS to be initialized with the exception of the debugger section and the watchdog functionalities. FW is rebooting from address 0000_0000h.
RSTMODS	DMARST	RW	4000_1008h	[1]	DMA and DMA wrapper reset bit. To exercise a module reset, FW has to set and to clear the proper bit accordingly. 0: Normal operation of the DMA 1: Performs a reset of the DMA
RSTMODS	CNFGOTP1WRST	RW	4000_1008h	[2]	OTP wrapper reset bit. To exercise a module reset, FW has to set and to clear the proper bit accordingly. 0: Normal operation of the OTP wrapper 1: Performs a reset of the OTP wrapper

Register name	Field name	Access	Address	Bits	Description
RSTMODS	BIFREGRST	RW	4000_1008h	[4]	BIF REG reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the BIF REG 1: Performs a reset of the BIF REG
RSTMODS	BIFPERSVIDRST	RW	4000_1008h	[5]	CRC32 reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the CRC32 1: Performs a reset of the CRC32
RSTMODS	BIFPERPMBUSRST	RW	4000_1008h	[6]	BIF PMBus PER IF reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the BIF PMBUS IF 1: Performs a reset of the BIF PMBUS IF
RSTMODS	BIFPERSSPRST	RW	4000_1008h	[7]	RESERVED
RSTMODS	BIFPERI2CRST	RW	4000_1008h	[8]	BIF I^2C PER IF reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the BIF I^2C IF 1: Performs a reset of the BIF I^2C IF
RSTMODS	BIFPERUARTRST	RW	4000_1008h	[9]	BIF UART PER IF reset bit. To exercise a module reset, FW has to set and to clear the proper bit accordingly. 0: Normal operation of the BIF UART IF 1: Performs a reset of the BIF UART IF
RSTMODS	DTIMER1RST	RW	4000_1008h	[11]	D_{TIMER1} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the D_{TIMER1} module 1: Performs a reset of the D_{TIMER1} module
RSTMODS	DTIMER2RST	RW	4000_1008h	[12]	D_{TIMER2} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the D_{TIMER2} module

Register name	Field name	Access	Address	Bits	Description
					1: Performs a reset of the D_{TIMER2} module
RSTMODS	DTIMER3RST	RW	4000_1008h	[13]	D_{TIMER3} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the D_{TIMER3} module 1: Performs a reset of the D_{TIMER3} module
RSTMODS	WDTIFRST	RW	4000_1008h	[14]	WDT module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the WDT module 1: Performs a reset of the WDT module
RSTMODS	GPIO0RST	RW	4000_1008h	[15]	GPIO0 module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the GPIO0 module 1: Performs a reset of the GPIO0 module
RSTMODS	GPIO1RST	RW	4000_1008h	[16]	GPIO1 module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Normal operation of the GPIO1 module 1: Performs a reset of the GPIO1 module
SW_PWDN_REQ	SOFT_PWDN	RW	4000_100Ch	[0]	The software power-down request register forces the CPUS into its power-down state: the CPUS will be under reset, the ROM and RAMs will be in power-down, and the primary input clock source will be gated. The SW_PWDN_REQ register is used in combination with the SWPWDN_CTRL register: only if the SW_PWDN_CTRL register has been configured does the SW_PWDN_REQ register have effect. 0: No request to enter power-down

Register name	Field name	Access	Address	Bits	Description
					1: Request CPUS to enter power-down. This will be effective if the EN_SWPWDN bit has been set.
SW_PWDN_CTRL	EN_SWPWDN	RW	4000_1010h	[0]	Enable CPUS/RAM/ROM power-down by software. Note: A software power-down procedure is applied in two steps: 1. Set EN_SWPWDN bit to “1” 2. Write SW_PWDN_REQ register Wakeup is performed by asserting the WKUP_IN signal (if it has been enabled) or the RSTN signal. 0: CPUS cannot be powered down by using SW procedure 1: CPUS can be powered down by using SW procedure
HW_PWDN_CTRL	EN_HWPWDN	RW	4000_1014h	[0]	Reserved. CPUS_EN is static set to 1 (XDPP1100 does not support hardware-commanded power-down).
SPARE_FF	SPARE_FF	RW	4000_1018h	[31:0]	Spare register
RSTMODS_SET	SWRST	W	4000_1020h	[0]	CPUS software reset. 0: Status of module reset unchanged 1: Forces reset of CPUS module Asserting SWRST causes the CPUS to be initialized with the exception of the debugger section and the watchdog functionalities. FW is rebooting from address 0000_0000h.
RSTMODS_SET	DMARST	W	4000_1020h	[1]	DMA and DMA wrapper reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	CNFGOTP1WRST	W	4000_1020h	[2]	OTP wrapper reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module

Register name	Field name	Access	Address	Bits	Description
RSTMODS_SET	BIFREGRST	W	4000_1020h	[4]	BIF REG reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	BIFPERSVIDRST	W	4000_1020h	[5]	CRC32 reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	BIFPERPMBUSRST	W	4000_1020h	[6]	BIF PMBUS PER IF reset bit. To exercise a module reset, FW has to set and to clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	BIFPERSSPRST	W	4000_1020h	[7]	Reserved
RSTMODS_SET	BIFPERI2CRST	W	4000_1020h	[8]	BIF I^2C PER IF reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	BIFPERUARTRST	W	4000_1020h	[9]	BIF UART PER IF reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	DTIMER1RST	W	4000_1020h	[11]	D_{TIMER1} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	DTIMER2RST	W	4000_1020h	[12]	D_{TIMER2} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged

Register name	Field name	Access	Address	Bits	Description
					1: Forces reset of module
RSTMODS_SET	DTIMER3RST	W	4000_1020h	[13]	D_{TIMER3} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	WDTIFRST	W	4000_1020h	[14]	WDT module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	GPIO0RST	W	4000_1020h	[15]	GPIO0 module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_SET	GPIO1RST	W	4000_1020h	[16]	GPIO1 module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Forces reset of module
RSTMODS_CLR	SWRST	W	4000_1024h	[0]	Reserved
RSTMODS_CLR	DMARST	W	4000_1024h	[1]	DMA and DMA wrapper reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	CNFGOTP1WRST	W	4000_1024h	[2]	OTP wrapper reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	BIFREGRST	W	4000_1024h	[4]	BIF REG reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged

Register name	Field name	Access	Address	Bits	Description
					1: Release reset
RSTMODS_CLR	BIFPERSVIDRST	W	4000_1024h	[5]	CRC32 reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	BIFPERPMBUSRST	W	4000_1024h	[6]	BIF PMBus PER IF reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	BIFPERSSPRST	W	4000_1024h	[7]	Reserved
RSTMODS_CLR	BIFPERI2CRST	W	4000_1024h	[8]	BIF I^2C PER IF reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	BIFPERUARTRST	W	4000_1024h	[9]	BIF UART PER IF reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	DTIMER1RST	W	4000_1024h	[11]	D_{TIMER1} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	DTIMER2RST	W	4000_1024h	[12]	D_{TIMER2} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	DTIMER3RST	W	4000_1024h	[13]	D_{TIMER3} module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly.

Register name	Field name	Access	Address	Bits	Description
					0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	WDTIFRST	W	4000_1024h	[14]	WDT module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	GPIO0RST	W	4000_1024h	[15]	GPIO0 module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset
RSTMODS_CLR	GPIO1RST	W	4000_1024h	[16]	GPIO1 module reset bit. To exercise a module reset, FW has to set and clear the proper bit accordingly. 0: Status of module reset unchanged 1: Release reset

15.4 Memory

XDPP1100 contains three different memory types: ROM, RAM and OTP. They provide the infrastructure to:

- Store the application code
- Provide an execution space
- Support configurability and trimming parameters
- Support a code patching mechanism

ROM and RAM are Infineon Technologies macros, highly optimized for area and power.

OTP is a NVM external IP provided by a third party.

15.4.1 Read-only memory

The read-only memory (ROM) module embeds an AHB wrapper, the ROM memory BIST and the ROM memory macro.

The ROM macro features 80 kB and is organized into 20K lines x 32 bits.

The ROM memory BIST is a HW engine used during chip production to verify the proper functionality of the module (self-check functional test). It reads the entire content of the ROM, generating a signature that is checked for correctness.

The ROM AHB wrapper implements the protocol layer to the AMBA® AHB bus. ROM access time is zero wait state.

The ROM module is used by the Arm® Cortex®-M0 for code boot repository and execution. On power-up, the Cortex®-M0 starts to execute code from the first ROM location.

15.4.2 Random-access memory

The random-access memory (RAM) space consists of two identical instances of RAM modules. Each module has its own AHB access port, enabling simultaneous access to different memories, through the bus matrix, by the two system masters (Cortex®-M0 and DMA).

Each RAM module embeds an AHB wrapper, the SRAM BIST, and the SRAM memory macro.

Each SRAM macro features a 16 kB SRAM; it is organized into 4K lines x 32 bits and it allows byte, half-word and word access.

The SRAM memory BIST is a HW engine used during production to verify the proper functionality of the module (self-check functional test). It reads and writes every memory cell macro using a specific algorithm (checkboard, MARCH2) to detect physical defects.

The RAM AHB wrapper implements the protocol layer to the AMBA® AHB bus. RAM access time is zero wait state on single READ/WRITE and single-cycle wait state on a sequence WRITE followed by a READ.

The RAM space is used by the Arm® Cortex®-M0 for handling FW variables, fast code execution and data storage.

15.4.3 One-time programmable memory

The one-time programmable (OTP) module is a block to access, configure, and test the SiPROM macro. TheSiPROM macrocell is an embedded NVM IP core providing OTP or emulated multiple-time programmable capability. OTP macro size is 512K bits, with a 128-bit data bus access (4K x 128-bit data organization).

OTP module interconnections are Arm® Cortex®-M0 oriented (32-bit data bus), with an AHB direct interface (not burst capable, with read-only access to OTP macro data) and an APB register interface for configuration, indirect access (to read and write/program OTP) and test. AHB and APB bus interfaces are defined according to Arm® AMBA® specification Rev 2.0 (ARM IHI 0011A).

The OTP block diagram is shown in Figure 107.

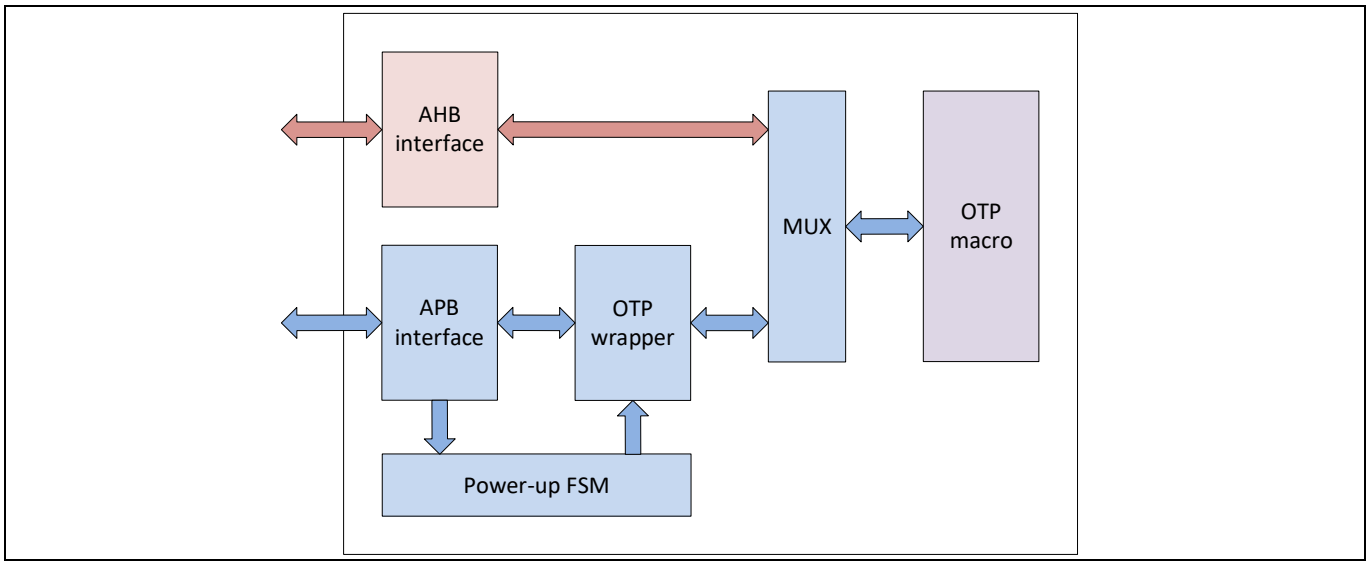

Figure 107 OTP module block diagram

15.4.3.1 OTP memory map

OTP data can be read by FW starting from address 1002_0000h, but to simplify the address decoding logic, it can also be accessed at addresses 0002_0000h, 0003_0000h and 1003_0000h. All these regions point to the same physical memory space (memory replicas).

Configuration registers can be accessed by the microcontroller, starting from address 0x5002000.

The OTP memory map is shown in Table 97.

Table 97 OTP memory map

Address	Size	Description
0000_0000h 0001_FFFFh	–	–
0002_0000h 0002_FFFFh	64 kB	OTP
0003_0000h 0003_FFFFh	64 kB	OTP (replica)
0004_0000h 1001_FFFFh	–	–
1002_0000h 1002_FFFFh	64 kB	OTP (replica)
1003_0000h 1003_FFFFh	64 kB	OTP (replica)
1004_0000h 5001_FFFFh	–	–
5002_0000h 5002_0078h	–	OTP configuration registers

15.4.3.2 OTP configuration interface

It is possible for FW to configure OTP block behavior or access the OTP macro indirectly or perform a specific test procedure (BIST) using the registers mapped on APB space starting from 5002_0000h.

OTP macro indirect access allows FW to perform specific commands like a single 128-bit read of a defined address, a single program of an OTP line (128 bits), a configuration registers change (MRA-MRB-MR) or a specific transaction (precharge, compare) execution.

Indirect access is mainly useful to program OTP without stalling the microcontroller for the entire duration of the write. An OTP line (128 bits) program can take from 50 µs up to 1600 µs in case of correct programming. In case there are any reading mismatches during the verify operation, unsuccessful bits are written again using a soaking pulse (500 µs) and for a maximum of 16 attempts.

Waiting for an instruction completion can be achieved with two different methods:

- Polling: looping on the APB_BUSY flag on the STAT register, waiting to find it at zero (operation completed)
- Interrupt: arming the instruction done interrupt (MASK_INSTR_DONE = 1b1 on INT_MASK register) and waiting until the OTP shell triggers an interrupt, then verifying that the right interrupt occurred (checking INSTR_DONE field set on INT_ACTIVE register) and clearing it (write into INT_ACTIVE_CLR)

OTP module configuration is based on a set of registers that can control specific timings such as the duration of the READ pulse, the duration of the programming pulse or the duration of power-up delay and reset hold-time. They can also influence some static behaviors, defining values for MRA-MRB-MR to be used during a READ, a READ1 (read after programming) or READ2 (read after soaking).

15.4.3.3 Indirect OTP access

The basic registers to perform an OTP indirect access are INSTR, EXEC and STAT.

Register INSTR contains two fields:

- OTP_ADDR, which selects the OTP row (address alignment is 128-bit data)
- INSTRUCTION, which select the specific operation that must be performed

Register EXEC is a write-only register; what is written on this register is not meaningful, but a write operation itself triggers the indirect access to start. Reading from this register always returns to zero and has no other effects.

An instruction completion is monitored on the STAT register APB_BUSY field. APB_BUSY high signals that an operation is ongoing; it is not possible to start another operation when APB_BUSY is high. It is also not possible to access OTP AHB direct space while APB_BUSY is high, in case an AHB bus error is triggered to the microcontroller (HARDFAULT).

Supported instructions are described in Table 98.

Table 98 OTP instructions for indirect access

Instruction value	Instruction	Source/destination register	Description
5h01	PROG and VERIFY	Data taken from DATAW0-4	Program and verify 128 bits into OTP (complete write sequence: write, read-back and, if necessary, soak)
5h02	READ	Data stored into OTP_Q0-3	Read 128 bits from OTP
5h03	PROG (basic transaction)	Data taken from DATAW0-4	Perform a 128-bit WRITE basic operation (no read-back, no soaking)
5h04	WRITE OTP data register	Data taken from DATAW0-4	Write OTP DR
5h05	SETMODE MRA	Data taken from READ_MRAB	Write OTP MRA
5h06	CMP	–	Perform a COMPARE basic operation
5h07	PCH	–	Perform a PRECHARGE basic operation

Instruction value	Instruction	Source/destination register	Description
5h08	BIST RESET	–	Execute a BIST reset
5h09	BIST STRESS TEST	–	Start a BIST stress test
5h0a	BIST CLEAN ARRAY TEST	–	Start a BIST clean array test
5h0b	SETMODE MRB	Data taken from READ_MRAB	Write OTP MRB
5h0c	SETMODE MR	Data taken from READ_MR	Write OTP MR

This is the register sequence to execute a 128-bit read:

- Write OTP_ADDR and INSTRUCTION (0x2, READ) on INSTR register
- Write something on EXEC register
- Monitor STAT register:
  - APB_BUSY should rise and fall back to zero when the operation is completed
  - Read OTP returned value on OTP_Q0-3 registers

This is the register sequence to execute a 128-bit (OTP line) write and verify:

- Write OTP_ADDR and INSTRUCTION (0x1, PROG and VERIFY) on which INSTR register
- Write the data to be programmed on OTP into DATAW0-4 registers
- Write something on EXEC register
- Monitor STAT register:
  - APB_BUSY should rise and fall back to zero when the operation is completed
  - Error flags (SOAK_FAIL, RD2_FL and RD1_FL) must be low

There are two different operations to perform an OTP write: “PROG” and “PROG and VERIFY”.

“PROG and VERIFY” operation executes a complete write sequence:

- Writing the data into the OTP with a WRITE pulse with a timing defined in PROG_PULSE_REG field of OTP_PROG_C register
- Reading back the data with MRA-MRB-MR defined in READ1_MRAB and READ1_MR registers
- In case of a mismatch (some bit not correctly programmed), start a soaking process on the unprogrammed bits:
  - Writing them using a WRITE pulse with a timing defined in PROG_SOAK_PULSE_REG field of OTP_PROG_C register
  - Reading back the data with MRA-MRB-MR defined in READ2_MRAB and READ2_MR registers
  - In case of mismatch, repeating this sequence MAXCNT_SOAK times (defined in register OTP_READ_C)

This entire sequence is executed automatically by “PROG and VERIFY” instruction, and error flags (RD1_FL, RD2_FL, SOAK_FAIL and SOAK_CNT) are set corresponding to any failure that occurs during the programming process.

"PROG" is just the atomic WRITE instruction:

- Writing the data into the OTP with a WRITE pulse with a timing defined in PROG_PULSE_REG field of OTP_PROG_C register

Everything else (READ1, soak and READ2) needs to be done manually with appropriate instructions. This can be useful in case of a debug requirement for which specific timings, sequences or MRA-MRB-MR registers need to be used.

15.4.3.4 OTP timing configuration

The OTP module has specific registers used to customize OTP macro interface timings. Table 99 summarizes the OTP timing requirements.

Table 99 OTP timing requirement

Register	Field	Description	Requirement
OTP_PWRUP_C	TRESR_REG	Reset recovery time	849.5 ns
	TPUR_PSR_REG	Power-up time TPUR + TPSR	300 µs
OTP_PROG_C	PROG_PULSE_REG	Programming pulse	50 µs
	PROG_SOAK_PULSE_REG	Programming pulse for soaking	500 µs
OPT_CP_C	VPP_WARMUP_REG	Charge pump warm-up time	10 µs
	VPP_WARMDOWN_REG	Charge pump warm-down time	10 µs
OTP_READ_C	MAXCNT_SOAK	MAX soaking pulses	16 pulses
	PROG_RECOVERY_TIME	Program recovery time	1 µs
	BIST_READ_TIMEOUT	BIST READ pulse	Programmable
	READ_TIMEOUT	READ timeout (internally terminated)	Programmable

Counters associated with those timings are all in the OTP kernel clock domain.

TRESR_REG and TPUR_PSR_REG characterize power-up and reset release.

PROG_PULSE_REG, PROG_SOAK_PULSE_REG and BIST_READ_TIMEOUT control the READ pulse width during standard programming, soaking and the BIST procedure, respectively.

VPP_WARMUP_REG and VPP_WARMDOWN_REG are used to define timings of the internal charge pump during a program phase: VPP_WARMUP_REG defines how long the charge pump needs to be activated for before the programming start; VPP_WARMDOWN_REG defines how long the charge pump needs to stay off for after a bit programming and before the next bit programming.

MAXCNT_SOAK is an integer number defining how many attempts need to be made in case of a programming mismatch before giving up.

PROG_RECOVERY_TIME characterizes the idle time between two bits programming.

READ_TIMEOUT controls a watchdog used during a READ verify sequence on a programming phase. During the programming of an internally terminated READ it is used to verify the written data, so the programming FSM is generating a READ pulse and waiting for the OTP macro to assert the READ termination (STATUS going high). If STATUS is not asserted within this watchdog time, to avoid deadlock, the program is aborted and an error is flagged (STAT register RD2_FL or RD1_FL flags).

15.4.3.5 OTP direct access

The OTP block allows direct access to the OTP memory space through the AHB interface; access is limited to read-only capability to avoid holding the main bus for the entire duration of a WRITE. A bus error (HRESP high) is triggered in case of a WRITE attempt during a READ and an error is logged on the ERR_RPT register for debugging purposes: the AHB_ILL_WR flag is set high and the OTP address used for the access is latched on the OTP_ADDR field.

READ access is performed using an AHB FSM that works as a wrapper between the OTP macro and the AHB bus interface (connected to the microcontroller). Because the OTP macro is organized in 128-bit word length and AHB in 32-bit word length, each AHB access based on the address alignment can cache up to four words.

To achieve the best-performing AHB FSM uses an externally terminated READ sequence on the OTP macro interface; OTP timing waveforms for this operation are described in Figure 108.

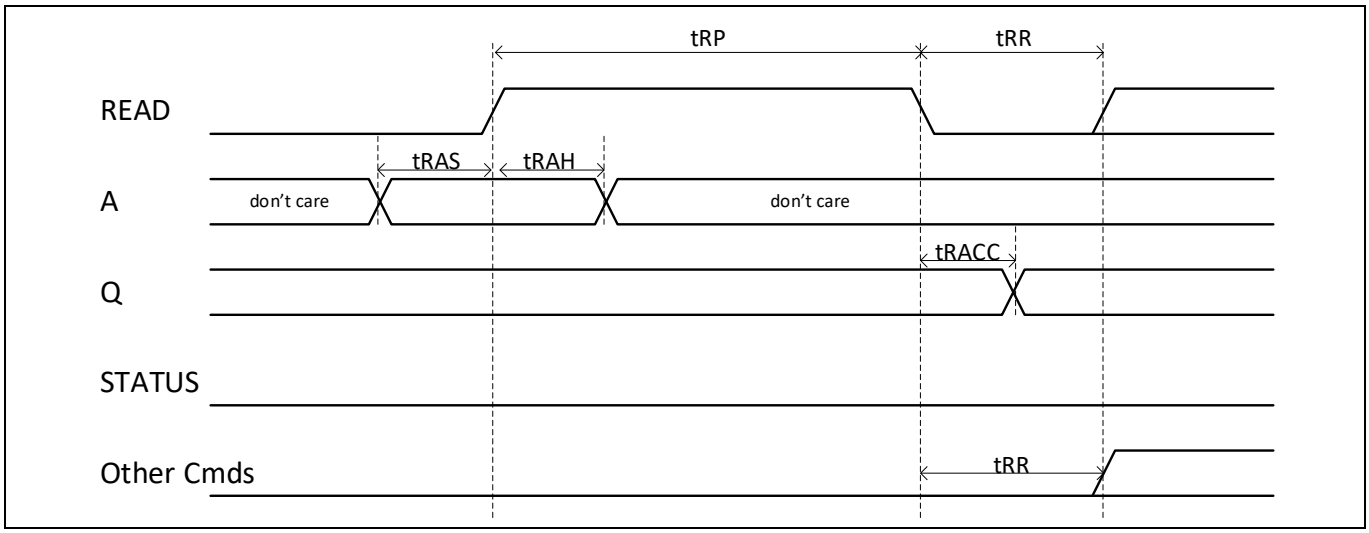

Figure 108 OTP timing diagram

OTP timing requirements are specified in Table 100.

Table 100 OTP timing requirements

Timing	Description	Requirement [ns]
t_{RAS}	READ address setup time	4.4
t_{RAH}	READ address hold-time	1.0
t_{RP}	READ pulse width	37
t_{RR}	READ recovery time	14.8
t_{RACC}	READ access time	11.2

On the AHB side, once a read request is initiated, it takes seven H_{CLK} cycles to complete an access; AHB FSM holds H_{READY} low (inserting wait cycles on the bus) until the OTP macro READ completion, then address consecutive accesses have zero wait state up to the following three reads (if the first address was 128-bit aligned); an access to a different OTP word again triggers a seven-cycle wait state.

The OTP module implements a prefetch FSM to mitigate the penalties on the access time. After any access, prefetch FSM automatically starts a read on the following OTP word, caching 128-bit data that correspond to the following four AHB 32-bit words.

The prefetching feature is enabled by default but can be statically disabled by setting the DIS_PF field on the CONF register to high.

Prefetch FSM access time is the same as AHB FSM (seven cycles), but as it is running while AHB is still consuming the main access data, it enables saving some cycles on a hit (access to consecutive addresses, see Figure 109).

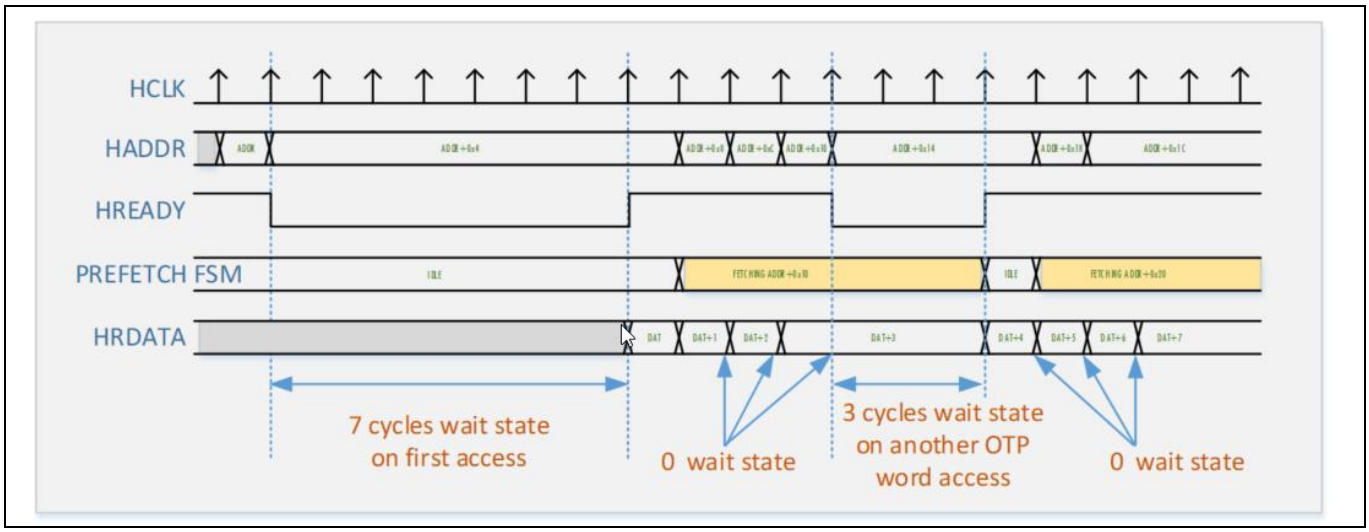

Figure 109 Access to two OTP 128-bit consecutive words

When accessing two consecutive words, four wait state cycles are saved on the AHB bus thanks to the prefetching. This improvement is less if the starting address is not aligned on 128 bits, or if not all of the data of the first read is used (there is less time available for the prefetch unit to complete its read).

But the worst-case scenario is when the microcontroller asks for an address different from the prefetching one while the prefetch FSM is executing the caching access (Figure 110). In this case prefetch FSM must abort its transfer and allow AHB FSM to proceed with the new access. The bus penalty in this situation rises to nine wait cycles (two for stopping prefetch FSM and the normal seven for the AHB FSM read).

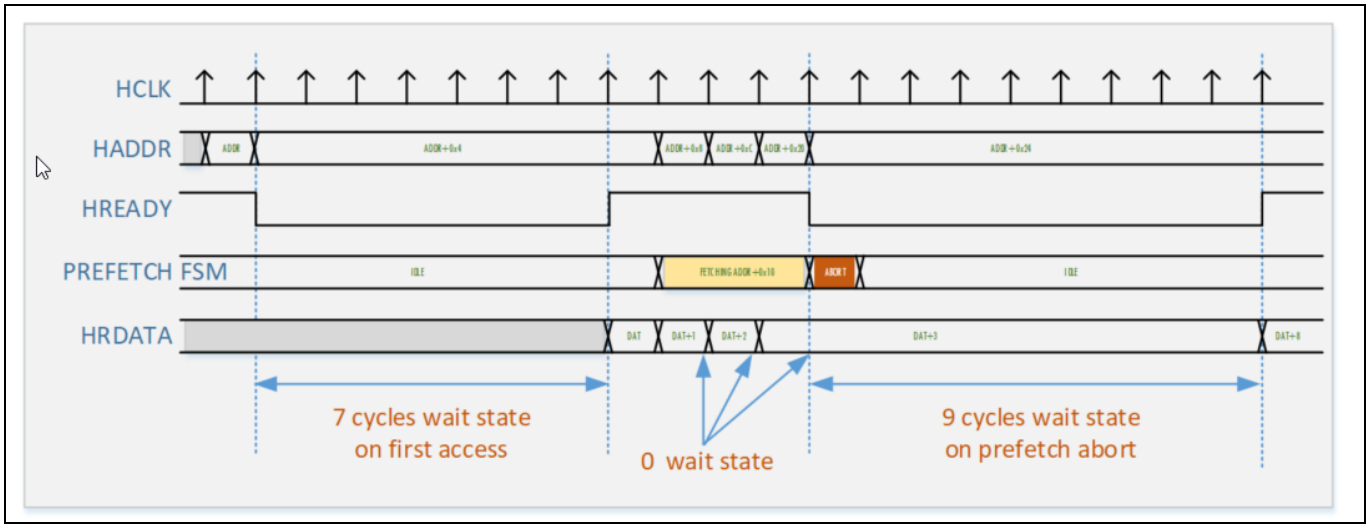

Figure 110 Prefetch abort scenario

Those results are quite ideal, but the Arm® Cortex®-M0 is based on a von Neumann architecture for which instruction and data interfaces are on the same bus. It also does not support BURST memory access (which can generate a burst read with a single instruction), so it should always interleave a code-fetch and a memory access. In this scenario (with code in ROM or RAM), the prefetch engine is able to minimize the wait state insertion where accessing consecutive OTP addresses would generate a single wait cycle during the prefetch hit phase (Figure 111).

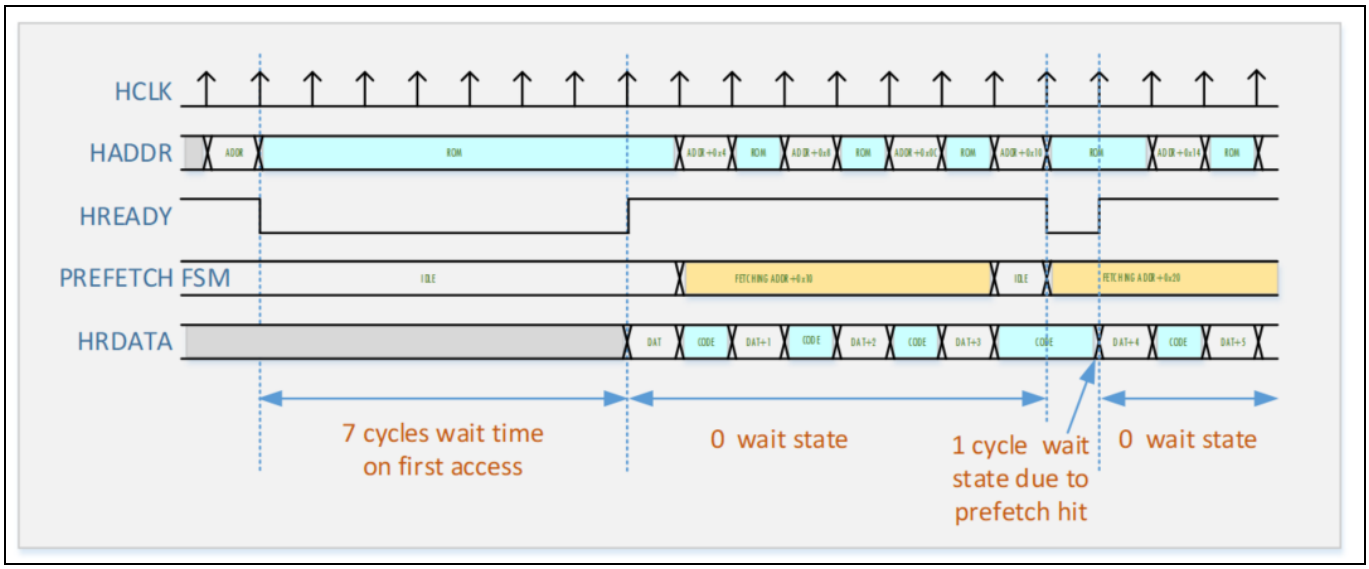

Figure 111 Optimal performance with code in ROM/RAM

An exception is the case in which the execution code is in OTP, for which the Cortex®-M0 can really access every cycle to the OTP memory space; this would be the worst-case performance scenario, and if not acceptable from a system point of view, it needs to be mitigated, for example by copying the OTP program onto the RAM and executing it from the RAM.

15.4.3.6 OTP module registers

The relevant OTP-related registers and their descriptions are provided in Table 101.

Table 101 OTP module-related register description

Register name	Field name	Access	Address	Bits	Description
STAT	PWR_STAT	R	5002_0000h	[0]	OTP power status. 0: OTP power off 1: OTP power on
STAT	BUSY	R	5002_0000h	[4]	Instruction status. 0: Completed 1: Busy
STAT	CMP_STAT	R	5002_0000h	[8]	Compare status (valid after direct access compare instruction)
STAT	PRECH_STAT	R	5002_0000h	[9]	Precharge status (valid after direct access compare instruction)
STAT	RD1_FL	R	5002_0000h	[16]	READ1 fail indicator.

Register name	Field name	Access	Address	Bits	Description
					0: No READ1 failure during programming 1: READ1 failure during programming
STAT	RD2_FL	R	5002_0000h	[17]	READ2 fail indicator. 0: No READ2 failure during programming 1: READ2 failure during programming
STAT	SOAK_FAIL	R	5002_0000h	[18]	SOAK fail indicator. 0: No SOAK failure during programming 1: SOAK failure during programming
STAT	SOAK_CNT	R	5002_0000h	[24]	Max. SOAK counter. For debug in case any of the bits requested to be programmed had to be soaked
CONF	PWRUP	RW	5002_0004h	[0]	OTP power command. 0: Power-down OTP 1: Power-up OTP
CONF	AHB_CLK_RATIO	RW	5002_0004h	[5:4]	AHB clock ratio with respect to the OTP kernel clock (25 MHz). 0: AHB = 100 MHz 1: AHB = 50 MHz 2: AHB = 25 MHz 3: AHB = 12.5 MHz
CONF	FL_EVEN	RW	5002_0004h	[8]	Even row “flip” control. 0: Do not flip even rows 1: Flip even rows
CONF	FL_ODD	RW	5002_0004h	[9]	Odd row “flip” control. 0: Do not flip odd rows 1: Flip odd rows
CONF	DIS_PF	RW	5002_0004h	[10]	Disable automatic prefetching. 0: Automatic prefetching enabled 1: Automatic prefetching disabled
CONF	RED_EN	RW	5002_0004h	[11]	Enable OTP redundancy. 0: Disable OTP redundancy 1: Enable OTP redundancy
INSTR	OTP_INSTR	RW	5002_0008h	[4:0]	Indirect OTP access instruction. 1: PROG and VERIFY

Register name	Field name	Access	Address	Bits	Description
					2: READ 3: PROG (basic transaction) 4: WRITE OTP data register (*) 5: SETMODE MRA (**) 6: CMP 7: PCH 8: BIST RESET 9: BIST STRESS TEST 10: BIST CLEAN ARRAY TEST 11: SETMODE MRB (**) 12: SETMODE MR (***) (*) Data is taken from DATAW0-4 (**) Data is taken from READ_MRAB register (***) Data is taken from READ_MR register
INSTR	OTP_ADDR	RW	5002_0008h	[28:18]	OTP address used by OTP_INSTR
EXEC	START_EXEC	R	5002_000Ch	[31:0]	Write anything to start the instruction, OTP_INSTR. Read always returns 0.
DATAW0	DATA	RW	5002_0010h	[31:0]	DATA[31:0] written to OTP by PROG and PROG+Verify instructions
DATAW1	DATA	RW	5002_0014h	[31:0]	DATA[64:32] written to OTP by PROG and PROG+Verify instructions
DATAW2	DATA	RW	5002_0018h	[31:0]	DATA[95:64] written to OTP by PROG and PROG+Verify instructions.
DATAW3	DATA	RW	5002_001Ch	[31:0]	DATA[127:96] written to OTP by PROG and PROG+Verify instructions
READ_MRAB	MRB_READ	RW	5002_0020h	[15:0]	MRB register setting to be used for OTP READ
READ_MRAB	MRA_READ	RW	5002_0020h	[31:16]	MRA register setting to be used for OTP READ
READ_MR	MR_READ	RW	5002_0024h	[15:0]	MR register setting to be used for OTP READ
READ1_MRAB	MRB_READ1	RW	5002_0028h	[31:16]	MRB register setting to be used for OTP READ1 during OTP programming

Register name	Field name	Access	Address	Bits	Description
READ1_MRAB	MRA_READ1	RW	5002_0028h	[15:0]	MRA register setting to be used for OTP READ1 during OTP programming
READ1_MR	MR_READ1	RW	5002_002Ch	[31:16]	MR register setting to be used for OTP READ1 during OTP programming
READ2_MRAB	MRB_READ2	RW	5002_0030h	[15:0]	MRB register setting to be used for OTP READ2 during OTP programming
READ2_MRAB	MRA_READ2	RW	5002_0030h	[31:16]	MRA register setting to be used for OTP READ2 during OTP programming
READ2_MR	MR_READ2	RW	5002_0034h	[15:0]	MR register setting to be used for OTP READ2 during OTP programming
OPT_MRAB	OTP_MRA	R	5002_0038h	[31:16]	MRA register output from OTP (current configuration)
OPT_MRAB	OTP_MRB	R	5002_0038h	[15:0]	MRB register output from OTP (current configuration)
OPT_MR	OTP_MR	R	5002_003Ch	[31:16]	MR register output from OTP (current configuration)
OTP_Q0	DATA	R	5002_0040h	[31:0]	DATA[31:0] output of OTP macro (used for READ)
OTP_Q1	DATA	R	5002_0044h	[31:0]	DATA[63:2] output of OTP macro (used for READ)
OTP_Q2	DATA	R	5002_0048h	[31:0]	DATA[95:64] output of OTP macro (used for READ)
OTP_Q3	DATA	R	5002_004Ch	[31:0]	DATA[127:96] output of OTP macro (used for READ)
INT_RAW	PWR_STAT	R	5002_0050h	[0]	Raw (unmasked) power-up status. 0: OTP power-up not complete 1: OTP power-up complete
INT_RAW	RAW_INSTR_DONE	R	5002_0050h	[1]	Raw (unmasked) instruction status. 0: OTP instruction not complete 1: OTP instruction complete
INT_RAW	RAW_READ_FAULT	R	5002_0050h	[2]	Raw (unmasked) READ fault status. 0: Fault did not occur during OTP indirect reading

Register name	Field name	Access	Address	Bits	Description
					1: Fault occurred during OTP indirect reading (timeout of internally terminated READ)
INT_RAW	RAW_PROG_FAULT	R	5002_0050h	[3]	Raw (unmasked) PROG fault status. 0: Fault did not occur during OTP programming 1: Fault occurred during OTP programming
INT_MASK	MSK_PWR_STAT	RW	5002_0054h	[0]	Mask for PWR_STAT. 0: Interrupt masked 1: Interrupt enabled
INT_MASK	MSK_INSTR_DONE	RW	5002_0054h	[1]	Mask for INSTR_DONE. 0: Interrupt masked 1: Interrupt enabled
INT_MASK	MSK_READ_FAULT	RW	5002_0054h	[2]	Mask for READ_FAULT. 0: Interrupt masked 1: Interrupt enabled
INT_MASK	MSK_PROG_FAULT	RW	5002_0054h	[3]	Mask for PROG_FAULT. 0: Interrupt masked 1: Interrupt enabled
INT_ACTIVE	PWR_STAT	R	5002_0058h	[0]	Active (masked) power-up status. 0: OTP power-up not complete 1: OTP power-up complete
INT_ACTIVE	INSTR_DONE	R	5002_0058h	[1]	Active (masked) instruction status. 0: OTP instruction not complete 1: OTP instruction complete
INT_ACTIVE	READ_FAULT	R	5002_0058h	[2]	Active (masked) READ fault status. 0: Fault did not occur during OTP indirect reading 1: Fault occurred during OTP indirect reading (timeout of internally terminated READ)
INT_ACTIVE	PROG_FAULT	R	5002_0058h	[3]	Active (masked) PROG fault status. 0: Fault did not occur during OTP programming 1: Fault occurred during OTP programming

Register name	Field name	Access	Address	Bits	Description
INT_ACTIVE_CLR	PWR_STAT	W	5002_005Ch	[0]	Writing 1 clears PWRUP_DONE on INT_ACTIVE register
INT_ACTIVE_CLR	INSTR_DONE	W	5002_005Ch	[1]	Writing 1 clears INSTR_DONE on INT_ACTIVE register
INT_ACTIVE_CLR	READ_FAULT	W	5002_005Ch	[2]	Writing 1 clears READ_FAULT on INT_ACTIVE register
INT_ACTIVE_CLR	PROG_FAULT	W	5002_005Ch	[3]	Writing 1 clears PROG_FAULT on INT_ACTIVE register
OTP_PWRUP_C	TPUR_PSR	RW	5002_0060h	[15:0]	Power-up time TPUR+TPSR (in OTP kernel clocks)
OTP_PWRUP_C	TRESR	RW	5002_0060h	[23:16]	Reset recovery time (in OTP kernel clocks)
OTP_PROG_C	PROG_SOAK_PULSE	RW	5002_0064h	[15:0]	Programming pulse for soaking (in OTP kernel clocks)
OTP_PROG_C	PROG_PULSE	RW	5002_0064h	[31:16]	Programming pulse (in OTP kernel clocks)
OTP_CP_C	VPP_WARMDOWN	RW	5002_0068h	[15:0]	Charge pump warm-down time (in OTP kernel clocks)
OTP_CP_C	VPP_WARMUP	RW	5002_0068h	[24:16]	Charge pump warm-up time (in OTP kernel clocks)
OTP_READ_C	READ_TIMEOUT	RW	5002_006Ch	[7:0]	Timeout for internally terminated READ (a READ FAULT is thrown if this timeout expires before STATUS rise)
OTP_READ_C	BIST_READ_TIMEOUT	RW	5002_006Ch	[15:8]	BIST READ pulse (in OTP kernel clocks)
OTP_READ_C	PROG_RECOVERY_TIME	RW	5002_006Ch	[20:16]	Program recovery time (in OTP kernel clocks)
OTP_READ_C	MAXCNT_SOAK	RW	5002_006Ch	[31:24]	MAX soaking pulse allowed
ERR_RPT	AHB_ILL_APB_ACC	R	5002_0070h	[0]	Read access on OTP shell AHB port while OTP was busy on an APB command – HRESP=ERR generated
ERR_RPT	AHB_ILL_WR	R	5002_0070h	[1]	Write access on OTP shell AHB port detected – HRESP=ERR generated
ERR_RPT	OTP_ADDR	R	5002_0070h	[27:16]	OTP address when the violation occurred
ERR_RPT_CLR	CLR	W	5002_0074h	[31:0]	Write clears all fields on ERR_RPT register
OTP_BIST_C	BMODE_SEL	RW	5002_0078h	[2:0]	OTP memory space select. 0: Main memory space 1: Boot memory space

Register name	Field name	Access	Address	Bits	Description
					2, 3: RDNT memory space 4, 5: Main + boot memory spaces 6, 7: Main + boot + RDNT memory spaces
OTP_BIST_C	BBUSY	R	5002_0078h	[4]	BIST busy flag (BIST running).
OTP_BIST_C	BFAIL	R	5002_0078h	[5]	BIST fail global flag.
OTP_BIST_C	BDONE	R	5002_0078h	[6]	BIST done flag.
OTP_BIST_C	BFAIL0	R	5002_0078h	[9:8]	BIST FAIL flag0. 0: PASS 1: FAIL on stress test 2: FAIL on clean array test 3: N/A
OTP_BIST_C	BFAIL1	R	5002_0078h	[11:10]	BIST FAIL flag1. 0: PASS 1: FAIL on main memory space 2: FAIL on boot memory space 3: FAIL on RDNT memory space
OTP_BIST_C	BFAIL2	R	5002_0078h	[13:12]	BIST FAIL flag2. 0: PASS 1: BIST FAIL – expected data not ZERO
OTP_BIST_C	BIST_ADDR_FAIL	R	5002_0078h	[28:16]	Last OTP address in which BIST detected a FAIL

15.5 Memory management unit

The memory management unit (MMU) module enables a configurable address remapping and a register/peripheral protection.

The MMU manipulates the addresses generated by the Cortex®-M0 before they are fed into the bus matrix. Addresses are decoded and compared inside the MMU using programmable LUTs; matching addresses (if enabled) are remapped based on a programmable offset and then sent to the bus matrix.

The MMU handles the memory space in sectors of 1 kB; for example, 80 kB ROM is represented by 80 LUT entries, one for each of the 1 kB sectors.

In the XDPP1100 MMU there are a total of 176 LUTs:

- 80 ROM LUTs
- 64 OTP LUTs
- 16 RAM1 LUTs
- 16 RAM2 LUTs

Each LUT contains information about how the MMU is performing the remapping and setting up access protection. For example, LUT0, covering addresses from 0000_0000h to 0000_03FFFh, is composed of the following:

- Write protection: If the Cortex®-M0 address is within 0000_0000h to 0000_03FFFh, the operation is a write and this bit is set, the MMU reports an illegal access fault
- Target address block: This remaps LUT0 into another LUT (sector); basically, the original sector address section is replaced by what defined by this parameter
- Target memory space: This remaps the part of the address that was selecting the ROM to another memory space (ROM/OTP/RAM)

If LUT0 is programmed:

- Write protect = 1
- Target address block = 2
- Target memory space = RAM2

Then a write access to 0000_0000h to 0000_03FFFh will generate an illegal access fault.

A read access to 0000_0000h to 0000_03FFFh will generate a read access to the same address offset inside the second sector (0000_0000h to 0000_0BFFFh) of the RAM2.

MMU has a global enable bit, EN_MMU on MMU_CFG register, to activate the address remapping logic. When the EN_MMU bit is reset, MMU is completely transparent (the Cortex®-M0 address is directly propagated to the bus matrix).

The MMU has also a set of global bits, on register MMU_PER_SPACE, to protect the entire peripheral space.

In case of an illegal access fault, the MMU_ERR_RPT register latches the information (target address block and target memory space) of the memory location causing the error.

15.5.1 MMU registers

The relevant MMU-related registers and their descriptions are provided in Table 102.

Table 102 MMU module-related register description

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_ROM0_DATA	PROT	RW	4000_4000h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM0_DATA	BLK_ADR	RW	4000_4000h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped.
MMU_LUT_ROM0_DATA	BASE_ADR	RW	4000_4000h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM1_DATA	PROT	RW	4000_4004h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM1_DATA	BLK_ADR	RW	4000_4004h	[7:1]	Defines the target address block into which the ROM section is remapped.
MMU_LUT_ROM1_DATA	BASE_ADR	RW	4000_4004h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM2_DATA	PROT	RW	4000_4008h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_ROM2_DATA	BLK_ADR	RW	4000_4008h	[7:1]	Defines the target address block into which the ROM section is remapped.
MMU_LUT_ROM2_DATA	BASE_ADR	RW	4000_4008h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM3_DATA	PROT	RW	4000_400Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM3_DATA	BLK_ADR	RW	4000_400Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM3_DATA	BASE_ADR	RW	4000_400Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM4_DATA	PROT	RW	4000_4010h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM4_DATA	BLK_ADR	RW	4000_4010h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM4_DATA	BASE_ADR	RW	4000_4010h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM5_DATA	PROT	RW	4000_4014h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM5_DATA	BLK_ADR	RW	4000_4014h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM5_DATA	BASE_ADR	RW	4000_4014h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM6_DATA	PROT	RW	4000_4018h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM6_DATA	BLK_ADR	RW	4000_4018h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM6_DATA	BASE_ADR	RW	4000_4018h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM7_DATA	PROT	RW	4000_401Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM7_DATA	BLK_ADR	RW	4000_401Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM7_DATA	BASE_ADR	RW	4000_401Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_ROM8_DATA	PROT	RW	4000_4020h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM8_DATA	BLK_ADR	RW	4000_4020h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM8_DATA	BASE_ADR	RW	4000_4020h	[9:8]	Defines the target memory space into which the ROM section is remapped 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM9_DATA	PROT	RW	4000_4024h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM9_DATA	BLK_ADR	RW	4000_4024h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM9_DATA	BASE_ADR	RW	4000_4024h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_ROM10_DATA	PROT	RW	4000_4028h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM10_DATA	BLK_ADR	RW	4000_4028h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM10_DATA	BASE_ADR	RW	4000_4028h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM11_DATA	PROT	RW	4000_402Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM11_DATA	BLK_ADR	RW	4000_402Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM11_DATA	BASE_ADR	RW	4000_402Ch	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM12_DATA	PROT	RW	4000_4030h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM12_DATA	BLK_ADR	RW	4000_4030h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM12_DATA	BASE_ADR	RW	4000_4030h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM13_DATA	PROT	RW	4000_4034h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM13_DATA	BLK_ADR	RW	4000_4034h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped
MMU_LUT_ROM13_DATA	BASE_ADR	RW	4000_4034h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM14_DATA	PROT	RW	4000_4038h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM14_DATA	BLK_ADR	RW	4000_4038h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM14_DATA	BASE_ADR	RW	4000_4038h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM15_DATA	PROT	RW	4000_403Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_ROM15_DATA	BLK_ADR	RW	4000_403Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM15_DATA	BASE_ADR	RW	4000_403Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM16_DATA	PROT	RW	4000_4040h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM16_DATA	BLK_ADR	RW	4000_4040h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM16_DATA	BASE_ADR	RW	4000_4040h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM17_DATA	PROT	RW	4000_4044h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM17_DATA	BLK_ADR	RW	4000_4044h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM17_DATA	BASE_ADR	RW	4000_4044h	[9:8]	Defines the target memory space into which the ROM section is remapped 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM18_DATA	PROT	RW	4000_4048h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM18_DATA	BLK_ADR	RW	4000_4048h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM18_DATA	BASE_ADR	RW	4000_4048h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM19_DATA	PROT	RW	4000_404Ch	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM19_DATA	BLK_ADR	RW	4000_404Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM19_DATA	BASE_ADR	RW	4000_404Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM20_DATA	PROT	RW	4000_4050h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM20_DATA	BLK_ADR	RW	4000_4050h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM20_DATA	BASE_ADR	RW	4000_4050h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_ROM21_DATA	PROT	RW	4000_4054h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM21_DATA	BLK_ADR	RW	4000_4054h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM21_DATA	BASE_ADR	RW	4000_4054h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM22_DATA	PROT	RW	4000_4058h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM22_DATA	BLK_ADR	RW	4000_4058h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM22_DATA	BASE_ADR	RW	4000_4058h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_ROM23_DATA	PROT	RW	4000_405Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM23_DATA	BLK_ADR	RW	4000_405Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM23_DATA	BASE_ADR	RW	4000_405Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM24_DATA	PROT	RW	4000_4060h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM24_DATA	BLK_ADR	RW	4000_4060h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM24_DATA	BASE_ADR	RW	4000_4060h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM25_DATA	PROT	RW	4000_4064h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM25_DATA	BLK_ADR	RW	4000_4064h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM25_DATA	BASE_ADR	RW	4000_4064h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM26_DATA	PROT	RW	4000_4068h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM26_DATA	BLK_ADR	RW	4000_4068h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped
MMU_LUT_ROM26_DATA	BASE_ADR	RW	4000_4068h	[9:8]	Defines the target memory space into which the ROM section is remapped 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM27_DATA	PROT	RW	4000_406Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM27_DATA	BLK_ADR	RW	4000_406Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM27_DATA	BASE_ADR	RW	4000_406Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM28_DATA	PROT	RW	4000_4070h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_ROM28_DATA	BLK_ADR	RW	4000_4070h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM28_DATA	BASE_ADR	RW	4000_4070h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM29_DATA	PROT	RW	4000_4074h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM29_DATA	BLK_ADR	RW	4000_4074h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM29_DATA	BASE_ADR	RW	4000_4074h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM30_DATA	PROT	RW	4000_4078h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM30_DATA	BLK_ADR	RW	4000_4078h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM30_DATA	BASE_ADR	RW	4000_4078h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM31_DATA	PROT	RW	4000_407Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM31_DATA	BLK_ADR	RW	4000_407Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM31_DATA	BASE_ADR	RW	4000_407Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM32_DATA	PROT	RW	4000_4080h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM32_DATA	BLK_ADR	RW	4000_4080h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM32_DATA	BASE_ADR	RW	4000_4080h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM33_DATA	PROT	RW	4000_4084h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM33_DATA	BLK_ADR	RW	4000_4084h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM33_DATA	BASE_ADR	RW	4000_4084h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_ROM34_DATA	PROT	RW	4000_4088h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM34_DATA	BLK_ADR	RW	4000_4088h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM34_DATA	BASE_ADR	RW	4000_4088h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM35_DATA	PROT	RW	4000_408Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM35_DATA	BLK_ADR	RW	4000_408Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM35_DATA	BASE_ADR	RW	4000_408Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_ROM36_DATA	PROT	RW	4000_4090h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM36_DATA	BLK_ADR	RW	4000_4090h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM36_DATA	BASE_ADR	RW	4000_4090h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM37_DATA	PROT	RW	4000_4094h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM37_DATA	BLK_ADR	RW	4000_4094h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM37_DATA	BASE_ADR	RW	4000_4094h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM38_DATA	PROT	RW	4000_4098h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM38_DATA	BLK_ADR	RW	4000_4098h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM38_DATA	BASE_ADR	RW	4000_4098h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM39_DATA	PROT	RW	4000_409Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM39_DATA	BLK_ADR	RW	4000_409Ch	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped
MMU_LUT_ROM39_DATA	BASE_ADR	RW	4000_409Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM40_DATA	PROT	RW	4000_40A0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM40_DATA	BLK_ADR	RW	4000_40A0h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM40_DATA	BASE_ADR	RW	4000_40A0h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM41_DATA	PROT	RW	4000_40A4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_ROM41_DATA	BLK_ADR	RW	4000_40A4h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM41_DATA	BASE_ADR	RW	4000_40A4h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM42_DATA	PROT	RW	4000_40A8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM42_DATA	BLK_ADR	RW	4000_40A8h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM42_DATA	BASE_ADR	RW	4000_40A8h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM43_DATA	PROT	RW	4000_40ACh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM43_DATA	BLK_ADR	RW	4000_40ACh	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM43_DATA	BASE_ADR	RW	4000_40ACh	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM44_DATA	PROT	RW	4000_40B0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM44_DATA	BLK_ADR	RW	4000_40B0h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM44_DATA	BASE_ADR	RW	4000_40B0h	[9:8]	Defines the target memory space into which the ROM section is remapped 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM45_DATA	PROT	RW	4000_40B4h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM45_DATA	BLK_ADR	RW	4000_40B4h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM45_DATA	BASE_ADR	RW	4000_40B4h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM46_DATA	PROT	RW	4000_40B8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM46_DATA	BLK_ADR	RW	4000_40B8h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM46_DATA	BASE_ADR	RW	4000_40B8h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_ROM47_DATA	PROT	RW	4000_40BCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM47_DATA	BLK_ADR	RW	4000_40BCh	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM47_DATA	BASE_ADR	RW	4000_40BCh	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM48_DATA	PROT	RW	4000_40C0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM48_DATA	BLK_ADR	RW	4000_40C0h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM48_DATA	BASE_ADR	RW	4000_40C0h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_ROM49_DATA	PROT	RW	4000_40C4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM49_DATA	BLK_ADR	RW	4000_40C4h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM49_DATA	BASE_ADR	RW	4000_40C4h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM50_DATA	PROT	RW	4000_40C8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM50_DATA	BLK_ADR	RW	4000_40C8h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM50_DATA	BASE_ADR	RW	4000_40C8h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM51_DATA	PROT	RW	4000_40CCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM51_DATA	BLK_ADR	RW	4000_40CCh	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM51_DATA	BASE_ADR	RW	4000_40CCh	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM52_DATA	PROT	RW	4000_40D0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM52_DATA	BLK_ADR	RW	4000_40D0h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped
MMU_LUT_ROM52_DATA	BASE_ADR	RW	4000_40D0h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM53_DATA	PROT	RW	4000_40D4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM53_DATA	BLK_ADR	RW	4000_40D4h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM53_DATA	BASE_ADR	RW	4000_40D4h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM54_DATA	PROT	RW	4000_40D8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_ROM54_DATA	BLK_ADR	RW	4000_40D8h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM54_DATA	BASE_ADR	RW	4000_40D8h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM55_DATA	PROT	RW	4000_40DCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM55_DATA	BLK_ADR	RW	4000_40DCh	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM55_DATA	BASE_ADR	RW	4000_40DCh	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM56_DATA	PROT	RW	4000_40E0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM56_DATA	BLK_ADR	RW	4000_40E0h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM56_DATA	BASE_ADR	RW	4000_40E0h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM57_DATA	PROT	RW	4000_40E4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM57_DATA	BLK_ADR	RW	4000_40E4h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM57_DATA	BASE_ADR	RW	4000_40E4h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM58_DATA	PROT	RW	4000_40E8h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM58_DATA	BLK_ADR	RW	4000_40E8h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM58_DATA	BASE_ADR	RW	4000_40E8h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM59_DATA	PROT	RW	4000_40ECh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM59_DATA	BLK_ADR	RW	4000_40ECh	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM59_DATA	BASE_ADR	RW	4000_40ECh	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_ROM60_DATA	PROT	RW	4000_40F0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM60_DATA	BLK_ADR	RW	4000_40F0h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM60_DATA	BASE_ADR	RW	4000_40F0h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM61_DATA	PROT	RW	4000_40F4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM61_DATA	BLK_ADR	RW	4000_40F4h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM61_DATA	BASE_ADR	RW	4000_40F4h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_ROM62_DATA	PROT	RW	4000_40F8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM62_DATA	BLK_ADR	RW	4000_40F8h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM62_DATA	BASE_ADR	RW	4000_40F8h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM63_DATA	PROT	RW	4000_40FCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM63_DATA	BLK_ADR	RW	4000_40FCh	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM63_DATA	BASE_ADR	RW	4000_40FCh	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM64_DATA	PROT	RW	4000_4100h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM64_DATA	BLK_ADR	RW	4000_4100h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM64_DATA	BASE_ADR	RW	4000_4100h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM65_DATA	PROT	RW	4000_4104h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM65_DATA	BLK_ADR	RW	4000_4104h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped
MMU_LUT_ROM65_DATA	BASE_ADR	RW	4000_4104h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM66_DATA	PROT	RW	4000_4108h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM66_DATA	BLK_ADR	RW	4000_4108h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM66_DATA	BASE_ADR	RW	4000_4108h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM67_DATA	PROT	RW	4000_410Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_ROM67_DATA	BLK_ADR	RW	4000_410Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM67_DATA	BASE_ADR	RW	4000_410Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM68_DATA	PROT	RW	4000_4110h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM68_DATA	BLK_ADR	RW	4000_4110h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM68_DATA	BASE_ADR	RW	4000_4110h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM69_DATA	PROT	RW	4000_4114h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM69_DATA	BLK_ADR	RW	4000_4114h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM69_DATA	BASE_ADR	RW	4000_4114h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM70_DATA	PROT	RW	4000_4118h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM70_DATA	BLK_ADR	RW	4000_4118h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM70_DATA	BASE_ADR	RW	4000_4118h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM71_DATA	PROT	RW	4000_411Ch	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM71_DATA	BLK_ADR	RW	4000_411Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM71_DATA	BASE_ADR	RW	4000_411Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM72_DATA	PROT	RW	4000_4120h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM72_DATA	BLK_ADR	RW	4000_4120h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM72_DATA	BASE_ADR	RW	4000_4120h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_ROM73_DATA	PROT	RW	4000_4124h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM73_DATA	BLK_ADR	RW	4000_4124h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM73_DATA	BASE_ADR	RW	4000_4124h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM74_DATA	PROT	RW	4000_4128h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM74_DATA	BLK_ADR	RW	4000_4128h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM74_DATA	BASE_ADR	RW	4000_4128h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_ROM75_DATA	PROT	RW	4000_412Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM75_DATA	BLK_ADR	RW	4000_412Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM75_DATA	BASE_ADR	RW	4000_412Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM76_DATA	PROT	RW	4000_4130h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM76_DATA	BLK_ADR	RW	4000_4130h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM76_DATA	BASE_ADR	RW	4000_4130h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM77_DATA	PROT	RW	4000_4134h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM77_DATA	BLK_ADR	RW	4000_4134h	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM77_DATA	BASE_ADR	RW	4000_4134h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM78_DATA	PROT	RW	4000_4138h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM78_DATA	BLK_ADR	RW	4000_4138h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the ROM section is remapped
MMU_LUT_ROM78_DATA	BASE_ADR	RW	4000_4138h	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_ROM79_DATA	PROT	RW	4000_413Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_ROM79_DATA	BLK_ADR	RW	4000_413Ch	[7:1]	Defines the target address block into which the ROM section is remapped
MMU_LUT_ROM79_DATA	BASE_ADR	RW	4000_413Ch	[9:8]	Defines the target memory space into which the ROM section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP0_DATA	PROT	RW	4000_4200h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_OTP0_DATA	BLK_ADR	RW	4000_4200h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP0_DATA	BASE_ADR	RW	4000_4200h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP1_DATA	PROT	RW	4000_4204h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP1_DATA	BLK_ADR	RW	4000_4204h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP1_DATA	BASE_ADR	RW	4000_4204h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP2_DATA	PROT	RW	4000_4208h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP2_DATA	BLK_ADR	RW	4000_4208h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP2_DATA	BASE_ADR	RW	4000_4208h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP3_DATA	PROT	RW	4000_420Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP3_DATA	BLK_ADR	RW	4000_420Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP3_DATA	BASE_ADR	RW	4000_420Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP4_DATA	PROT	RW	4000_4210h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP4_DATA	BLK_ADR	RW	4000_4210h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP4_DATA	BASE_ADR	RW	4000_4210h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP5_DATA	PROT	RW	4000_4214h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP5_DATA	BLK_ADR	RW	4000_4214h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP5_DATA	BASE_ADR	RW	4000_4214h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_OTP6_DATA	PROT	RW	4000_4218h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP6_DATA	BLK_ADR	RW	4000_4218h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP6_DATA	BASE_ADR	RW	4000_4218h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP7_DATA	PROT	RW	4000_421Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP7_DATA	BLK_ADR	RW	4000_421Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP7_DATA	BASE_ADR	RW	4000_421Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_OTP8_DATA	PROT	RW	4000_4220h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP8_DATA	BLK_ADR	RW	4000_4220h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP8_DATA	BASE_ADR	RW	4000_4220h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP9_DATA	PROT	RW	4000_4224h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP9_DATA	BLK_ADR	RW	4000_4224h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP9_DATA	BASE_ADR	RW	4000_4224h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP10_DATA	PROT	RW	4000_4228h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP10_DATA	BLK_ADR	RW	4000_4228h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP10_DATA	BASE_ADR	RW	4000_4228h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP11_DATA	PROT	RW	4000_422Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP11_DATA	BLK_ADR	RW	4000_422Ch	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped
MMU_LUT_OTP11_DATA	BASE_ADR	RW	4000_422Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP12_DATA	PROT	RW	4000_4230h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP12_DATA	BLK_ADR	RW	4000_4230h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP12_DATA	BASE_ADR	RW	4000_4230h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP13_DATA	PROT	RW	4000_4234h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_OTP13_DATA	BLK_ADR	RW	4000_4234h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP13_DATA	BASE_ADR	RW	4000_4234h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP14_DATA	PROT	RW	4000_4238h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP14_DATA	BLK_ADR	RW	4000_4238h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP14_DATA	BASE_ADR	RW	4000_4238h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP15_DATA	PROT	RW	4000_423Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP15_DATA	BLK_ADR	RW	4000_423Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP15_DATA	BASE_ADR	RW	4000_423Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP16_DATA	PROT	RW	4000_4240h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP16_DATA	BLK_ADR	RW	4000_4240h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP16_DATA	BASE_ADR	RW	4000_4240h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP17_DATA	PROT	RW	4000_4244h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP17_DATA	BLK_ADR	RW	4000_4244h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP17_DATA	BASE_ADR	RW	4000_4244h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP18_DATA	PROT	RW	4000_4248h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP18_DATA	BLK_ADR	RW	4000_4248h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP18_DATA	BASE_ADR	RW	4000_4248h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_OTP19_DATA	PROT	RW	4000_424Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP19_DATA	BLK_ADR	RW	4000_424Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP19_DATA	BASE_ADR	RW	4000_424Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP20_DATA	PROT	RW	4000_4250h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP20_DATA	BLK_ADR	RW	4000_4250h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP20_DATA	BASE_ADR	RW	4000_4250h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_OTP21_DATA	PROT	RW	4000_4254h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP21_DATA	BLK_ADR	RW	4000_4254h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP21_DATA	BASE_ADR	RW	4000_4254h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP22_DATA	PROT	RW	4000_4258h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP22_DATA	BLK_ADR	RW	4000_4258h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP22_DATA	BASE_ADR	RW	4000_4258h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP23_DATA	PROT	RW	4000_425Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP23_DATA	BLK_ADR	RW	4000_425Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP23_DATA	BASE_ADR	RW	4000_425Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP24_DATA	PROT	RW	4000_4260h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP24_DATA	BLK_ADR	RW	4000_4260h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped
MMU_LUT_OTP24_DATA	BASE_ADR	RW	4000_4260h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP25_DATA	PROT	RW	4000_4264h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP25_DATA	BLK_ADR	RW	4000_4264h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP25_DATA	BASE_ADR	RW	4000_4264h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP26_DATA	PROT	RW	4000_4268h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_OTP26_DATA	BLK_ADR	RW	4000_4268h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP26_DATA	BASE_ADR	RW	4000_4268h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP27_DATA	PROT	RW	4000_426Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP27_DATA	BLK_ADR	RW	4000_426Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP27_DATA	BASE_ADR	RW	4000_426Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP28_DATA	PROT	RW	4000_4270h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP28_DATA	BLK_ADR	RW	4000_4270h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP28_DATA	BASE_ADR	RW	4000_4270h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP29_DATA	PROT	RW	4000_4274h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP29_DATA	BLK_ADR	RW	4000_4274h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP29_DATA	BASE_ADR	RW	4000_4274h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP30_DATA	PROT	RW	4000_4278h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP30_DATA	BLK_ADR	RW	4000_4278h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP30_DATA	BASE_ADR	RW	4000_4278h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP31_DATA	PROT	RW	4000_427Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP31_DATA	BLK_ADR	RW	4000_427Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP31_DATA	BASE_ADR	RW	4000_427Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_OTP32_DATA	PROT	RW	4000_4280h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP32_DATA	BLK_ADR	RW	4000_4280h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP32_DATA	BASE_ADR	RW	4000_4280h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP33_DATA	PROT	RW	4000_4284h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP33_DATA	BLK_ADR	RW	4000_4284h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP33_DATA	BASE_ADR	RW	4000_4284h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_OTP34_DATA	PROT	RW	4000_4288h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP34_DATA	BLK_ADR	RW	4000_4288h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP34_DATA	BASE_ADR	RW	4000_4288h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP35_DATA	PROT	RW	4000_428Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP35_DATA	BLK_ADR	RW	4000_428Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP35_DATA	BASE_ADR	RW	4000_428Ch	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP36_DATA	PROT	RW	4000_4290h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP36_DATA	BLK_ADR	RW	4000_4290h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP36_DATA	BASE_ADR	RW	4000_4290h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP37_DATA	PROT	RW	4000_4294h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP37_DATA	BLK_ADR	RW	4000_4294h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped
MMU_LUT_OTP37_DATA	BASE_ADR	RW	4000_4294h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP38_DATA	PROT	RW	4000_4298h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP38_DATA	BLK_ADR	RW	4000_4298h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP38_DATA	BASE_ADR	RW	4000_4298h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP39_DATA	PROT	RW	4000_429Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_OTP39_DATA	BLK_ADR	RW	4000_429Ch	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP39_DATA	BASE_ADR	RW	4000_429Ch	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP40_DATA	PROT	RW	4000_42A0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP40_DATA	BLK_ADR	RW	4000_42A0h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP40_DATA	BASE_ADR	RW	4000_42A0h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP41_DATA	PROT	RW	4000_42A4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP41_DATA	BLK_ADR	RW	4000_42A4h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP41_DATA	BASE_ADR	RW	4000_42A4h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP42_DATA	PROT	RW	4000_42A8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP42_DATA	BLK_ADR	RW	4000_42A8h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP42_DATA	BASE_ADR	RW	4000_42A8h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP43_DATA	PROT	RW	4000_42ACh	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP43_DATA	BLK_ADR	RW	4000_42ACh	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP43_DATA	BASE_ADR	RW	4000_42ACh	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP44_DATA	PROT	RW	4000_42B0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP44_DATA	BLK_ADR	RW	4000_42B0h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP44_DATA	BASE_ADR	RW	4000_42B0h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_OTP45_DATA	PROT	RW	4000_42B4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP45_DATA	BLK_ADR	RW	4000_42B4h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP45_DATA	BASE_ADR	RW	4000_42B4h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP46_DATA	PROT	RW	4000_42B8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP46_DATA	BLK_ADR	RW	4000_42B8h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP46_DATA	BASE_ADR	RW	4000_42B8h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_OTP47_DATA	PROT	RW	4000_42BCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP47_DATA	BLK_ADR	RW	4000_42BCh	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP47_DATA	BASE_ADR	RW	4000_42BCh	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP48_DATA	PROT	RW	4000_42C0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP48_DATA	BLK_ADR	RW	4000_42C0h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP48_DATA	BASE_ADR	RW	4000_42C0h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP49_DATA	PROT	RW	4000_42C4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP49_DATA	BLK_ADR	RW	4000_42C4h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP49_DATA	BASE_ADR	RW	4000_42C4h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP50_DATA	PROT	RW	4000_42C8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP50_DATA	BLK_ADR	RW	4000_42C8h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped
MMU_LUT_OTP50_DATA	BASE_ADR	RW	4000_42C8h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP51_DATA	PROT	RW	4000_42CCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP51_DATA	BLK_ADR	RW	4000_42CCh	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP51_DATA	BASE_ADR	RW	4000_42CCh	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP52_DATA	PROT	RW	4000_42D0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_OTP52_DATA	BLK_ADR	RW	4000_42D0h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP52_DATA	BASE_ADR	RW	4000_42D0h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP53_DATA	PROT	RW	4000_42D4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP53_DATA	BLK_ADR	RW	4000_42D4h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP53_DATA	BASE_ADR	RW	4000_42D4h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP54_DATA	PROT	RW	4000_42D8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP54_DATA	BLK_ADR	RW	4000_42D8h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP54_DATA	BASE_ADR	RW	4000_42D8h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP55_DATA	PROT	RW	4000_42DCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP55_DATA	BLK_ADR	RW	4000_42DCh	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP55_DATA	BASE_ADR	RW	4000_42DCh	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP56_DATA	PROT	RW	4000_42E0h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP56_DATA	BLK_ADR	RW	4000_42E0h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP56_DATA	BASE_ADR	RW	4000_42E0h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP57_DATA	PROT	RW	4000_42E4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP57_DATA	BLK_ADR	RW	4000_42E4h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP57_DATA	BASE_ADR	RW	4000_42E4h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_OTP58_DATA	PROT	RW	4000_42E8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP58_DATA	BLK_ADR	RW	4000_42E8h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP58_DATA	BASE_ADR	RW	4000_42E8h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP59_DATA	PROT	RW	4000_42ECh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP59_DATA	BLK_ADR	RW	4000_42ECh	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP59_DATA	BASE_ADR	RW	4000_42ECh	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_OTP60_DATA	PROT	RW	4000_42F0h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP60_DATA	BLK_ADR	RW	4000_42F0h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP60_DATA	BASE_ADR	RW	4000_42F0h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP61_DATA	PROT	RW	4000_42F4h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP61_DATA	BLK_ADR	RW	4000_42F4h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP61_DATA	BASE_ADR	RW	4000_42F4h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP62_DATA	PROT	RW	4000_42F8h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP62_DATA	BLK_ADR	RW	4000_42F8h	[7:1]	Defines the target address block into which the OTP section is remapped
MMU_LUT_OTP62_DATA	BASE_ADR	RW	4000_42F8h	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_OTP63_DATA	PROT	RW	4000_42FCh	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_OTP63_DATA	BLK_ADR	RW	4000_42FCh	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the OTP section is remapped
MMU_LUT_OTP63_DATA	BASE_ADR	RW	4000_42FCh	[9:8]	Defines the target memory space into which the OTP section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM10_DATA	PROT	RW	4000_4400h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM10_DATA	BLK_ADR	RW	4000_4400h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM10_DATA	BASE_ADR	RW	4000_4400h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM11_DATA	PROT	RW	4000_4404h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_RAM11_DATA	BLK_ADR	RW	4000_4404h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM11_DATA	BASE_ADR	RW	4000_4404h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM12_DATA	PROT	RW	4000_4408h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM12_DATA	BLK_ADR	RW	4000_4408h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM12_DATA	BASE_ADR	RW	4000_4408h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM13_DATA	PROT	RW	4000_440Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM13_DATA	BLK_ADR	RW	4000_440Ch	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM13_DATA	BASE_ADR	RW	4000_440Ch	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM14_DATA	PROT	RW	4000_4410h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM14_DATA	BLK_ADR	RW	4000_4410h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM14_DATA	BASE_ADR	RW	4000_4410h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM15_DATA	PROT	RW	4000_4414h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM15_DATA	BLK_ADR	RW	4000_4414h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM15_DATA	BASE_ADR	RW	4000_4414h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM16_DATA	PROT	RW	4000_4418h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM16_DATA	BLK_ADR	RW	4000_4418h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM16_DATA	BASE_ADR	RW	4000_4418h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_RAM17_DATA	PROT	RW	4000_441Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM17_DATA	BLK_ADR	RW	4000_441Ch	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM17_DATA	BASE_ADR	RW	4000_441Ch	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM18_DATA	PROT	RW	4000_4420h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM18_DATA	BLK_ADR	RW	4000_4420h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM18_DATA	BASE_ADR	RW	4000_4420h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_RAM19_DATA	PROT	RW	4000_4424h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM19_DATA	BLK_ADR	RW	4000_4424h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM19_DATA	BASE_ADR	RW	4000_4424h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM110_DATA	PROT	RW	4000_4428h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM110_DATA	BLK_ADR	RW	4000_4428h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM110_DATA	BASE_ADR	RW	4000_4428h	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM111_DATA	PROT	RW	4000_442Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM111_DATA	BLK_ADR	RW	4000_442Ch	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM111_DATA	BASE_ADR	RW	4000_442Ch	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM112_DATA	PROT	RW	4000_4430h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM112_DATA	BLK_ADR	RW	4000_4430h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the RAM1 section is remapped
MMU_LUT_RAM112_DATA	BASE_ADR	RW	4000_4430h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM113_DATA	PROT	RW	4000_4434h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM113_DATA	BLK_ADR	RW	4000_4434h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM113_DATA	BASE_ADR	RW	4000_4434h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM114_DATA	PROT	RW	4000_4438h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_RAM114_DATA	BLK_ADR	RW	4000_4438h	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM114_DATA	BASE_ADR	RW	4000_4438h	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM115_DATA	PROT	RW	4000_443Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM115_DATA	BLK_ADR	RW	4000_443Ch	[7:1]	Defines the target address block into which the RAM1 section is remapped
MMU_LUT_RAM115_DATA	BASE_ADR	RW	4000_443Ch	[9:8]	Defines the target memory space into which the RAM1 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM20_DATA	PROT	RW	4000_4500h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM20_DATA	BLK_ADR	RW	4000_4500h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM20_DATA	BASE_ADR	RW	4000_4500h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM21_DATA	PROT	RW	4000_4504h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM21_DATA	BLK_ADR	RW	4000_4504h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM21_DATA	BASE_ADR	RW	4000_4504h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM22_DATA	PROT	RW	4000_4508h	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM22_DATA	BLK_ADR	RW	4000_4508h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM22_DATA	BASE_ADR	RW	4000_4508h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM23_DATA	PROT	RW	4000_450Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM23_DATA	BLK_ADR	RW	4000_450Ch	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM23_DATA	BASE_ADR	RW	4000_450Ch	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2

Register name	Field name	Access	Address	Bits	Description
MMU_LUT_RAM24_DATA	PROT	RW	4000_4510h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM24_DATA	BLK_ADR	RW	4000_4510h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM24_DATA	BASE_ADR	RW	4000_4510h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM25_DATA	PROT	RW	4000_4514h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM25_DATA	BLK_ADR	RW	4000_4514h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM25_DATA	BASE_ADR	RW	4000_4514h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP

Register name	Field name	Access	Address	Bits	Description
					2: RAM1 3: RAM2
MMU_LUT_RAM26_DATA	PROT	RW	4000_4518h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM26_DATA	BLK_ADR	RW	4000_4518h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM26_DATA	BASE_ADR	RW	4000_4518h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM27_DATA	PROT	RW	4000_451Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM27_DATA	BLK_ADR	RW	4000_451Ch	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM27_DATA	BASE_ADR	RW	4000_451Ch	[9:8]	Defines the target memory space into

Register name	Field name	Access	Address	Bits	Description
					which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM28_DATA	PROT	RW	4000_4520h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM28_DATA	BLK_ADR	RW	4000_4520h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM28_DATA	BASE_ADR	RW	4000_4520h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM29_DATA	PROT	RW	4000_4524h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM29_DATA	BLK_ADR	RW	4000_4524h	[7:1]	Defines the target address block into

Register name	Field name	Access	Address	Bits	Description
					which the RAM2 section is remapped
MMU_LUT_RAM29_DATA	BASE_ADR	RW	4000_4524h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM210_DATA	PROT	RW	4000_4528h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM210_DATA	BLK_ADR	RW	4000_4528h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM210_DATA	BASE_ADR	RW	4000_4528h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM211_DATA	PROT	RW	4000_452Ch	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection

Register name	Field name	Access	Address	Bits	Description
					1: Enable write protection
MMU_LUT_RAM211_DATA	BLK_ADR	RW	4000_452Ch	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM211_DATA	BASE_ADR	RW	4000_452Ch	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM212_DATA	PROT	RW	4000_4530h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM212_DATA	BLK_ADR	RW	4000_4530h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM212_DATA	BASE_ADR	RW	4000_4530h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM213_DATA	PROT	RW	4000_4534h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will

Register name	Field name	Access	Address	Bits	Description
					result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM213_DATA	BLK_ADR	RW	4000_4534h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM213_DATA	BASE_ADR	RW	4000_4534h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM214_DATA	PROT	RW	4000_4538h	[0]	Defines the write protection of the target address block in the target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM214_DATA	BLK_ADR	RW	4000_4538h	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM214_DATA	BASE_ADR	RW	4000_4538h	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_LUT_RAM215_DATA	PROT	RW	4000_453Ch	[0]	Defines the write protection of the target address block in the

Register name	Field name	Access	Address	Bits	Description
					target memory space. Any write attempt to a protected block will result in an illegal access fault. 0: Disable write protection 1: Enable write protection
MMU_LUT_RAM215_DATA	BLK_ADR	RW	4000_453Ch	[7:1]	Defines the target address block into which the RAM2 section is remapped
MMU_LUT_RAM215_DATA	BASE_ADR	RW	4000_453Ch	[9:8]	Defines the target memory space into which the RAM2 section is remapped. 0: ROM 1: OTP 2: RAM1 3: RAM2
MMU_PER_SPACE	PER	RW	4000_4600h	[27:0]	Configures the write protection of the peripherals mapped in the peripheral space. A 1 in a bit position enables protection while a 0 disables protection. [0]: Not used [1]: WDT, D_{TIMER}, GPIO [2]: UART, I^2C, PMBus [3]: Trim [4]: Analog [5]: VSP0 [6]: VSP1 [7]: VSP2 [8]: VCTRL0 [9]: VCTRL1 [10]: PID0 [11]: PID1 [12]: ISP0 [13]: ISP1 [14]: PWM

Register name	Field name	Access	Address	Bits	Description
					[15]: COMMON [16]: TELEM0 [17]: TELEM1 [18]: FAULT0 [19]: FAULT1 [20]: FAN1 [21]: FAN2 [22]: TSEN [23]: TLMCOM [24]: FAULTCOM [25]: TEST [25]: TESTSTAT [25]: Reserved
MMU_PER_SPACE_SET	PER	W	4000_4604h	[27:0]	Enables the write protection of the peripherals mapped in the peripheral space. A 1 in a bit position enables protection while a 0 retains the existing protection setting. [0]: Not used [1]: WDT, D_{TIMER}, GPIO [2]: UART, I^2C, SSP, PMBus [3]: Trim [4]: Analog [5]: VSP0 [6]: VSP1 [7]: VSP2 [8]: VCTRL0 [9]: VCTRL1 [10]: PID0 [11]: PID1 [12]: ISP0 [13]: ISP1 [14]: PWM [15]: COMMON [16]: TELEM0 [17]: TELEM1 [18]: FAULT0 [19]: FAULT1

Register name	Field name	Access	Address	Bits	Description
					[20]: FAN1 [21]: FAN2 [22]: TSEN [23]: TLMCOM [24]: FAULTCOM [25]: TEST [25]: TESTSTAT [25]: Reserved
MMU_PER_SPACE_CLR	PER	W	4000_4600h	[27:0]	Disables the write protection of the peripherals mapped in the peripheral space. A 1 in a bit position disables protection while a 0 retains the existing protection setting. [0]: Not used [1]: WDT, D_{TIMER}, GPIO [2]: UART, I^2C, PMBus [3]: Trim [4]: Analog [5]: VSP0 [6]: VSP1 [7]: VSP2 [8]: VCTRL0 [9]: VCTRL1 [10]: PID0 [11]: PID1 [12]: ISP0 [13]: ISP1 [14]: PWM [15]: COMMON [16]: TELEM0 [17]: TELEM1 [18]: FAULT0 [19]: FAULT1 [20]: FAN1 [21]: FAN2 [22]: TSEN [23]: TLMCOM [24]: FAULTCOM [25]: TEST

Register name	Field name	Access	Address	Bits	Description
					[25]: TESTSTAT [25]: Reserved
MMU_CNFG	EN_MMU	RW	4000_4700h	[0]	MMU function enable/disable. 0: Disable MMU function 1: Enable MMU function
MMU_CNFG	EN_LOAD_MMU_TABLE	RW	4000_4700h	[1]	MMU table update enable/disable. 0: MMU table update is not allowed 1: MMU table update is allowed
MMU_ERR_RPT	BLK_ADR	R	4000_4704h	[6:0]	Reports the block address of a memory location causing an error
MMU_ERR_RPT	BASE_ADR	R	4000_4704h	[21:7]	Reports the base address of a memory location causing an error
MMU_ERR_RPT	PROT	R	4000_4704h	[22]	Reflects the status of the protection bit. 0: Register does not reflect an illegal address 1: Register reports the latest illegal address causing a fault
MMU_ERR_RPT_CLR	CLR	W	4000_4708h	[0]	Write-only register that clears MMU_ERR_RPT.

15.6 DMA controller

The DMA controller is an Arm® PrimeCell IP (PL230); extensive documentation can be found in the “Arm® PrimeCell µDMA controller (PL230) Technical Reference Manual”.

The principal features of the DMA controller are:

- It is compatible with AHB-Lite for DMA transfers
- It is compatible with APB for programming the registers
- It has a single AHB-Lite master for transferring data using a 32-bit address bus and 32-bit data bus
- It has a configurable number of DMA channels
- Each DMA channel has dedicated handshake signals
- Each DMA channel has a programmable priority level
- Each priority level arbitrates using a fixed priority that is determined by the DMA channel number

- It supports multiple transfer types:
  - memory-to-memory
  - memory-to-peripheral
  - peripheral-to-memory
- It supports multiple DMA cycle types
- It supports multiple DMA transfer data widths
- Each DMA channel can access a primary, and alternate, channel control data structure
- All the channel control data is stored in system memory (RAM) in little-endian format
- It performs all DMA transfers using the single AHB-Lite burst type
- The destination data width is equal to the source data width
- The number of transfers in a single DMA cycle can be programmed from 1 to 1024
- The transfer address increment can be greater than the data width

15.6.1 DMA block diagram

The DMA block diagram is shown in Figure 112.

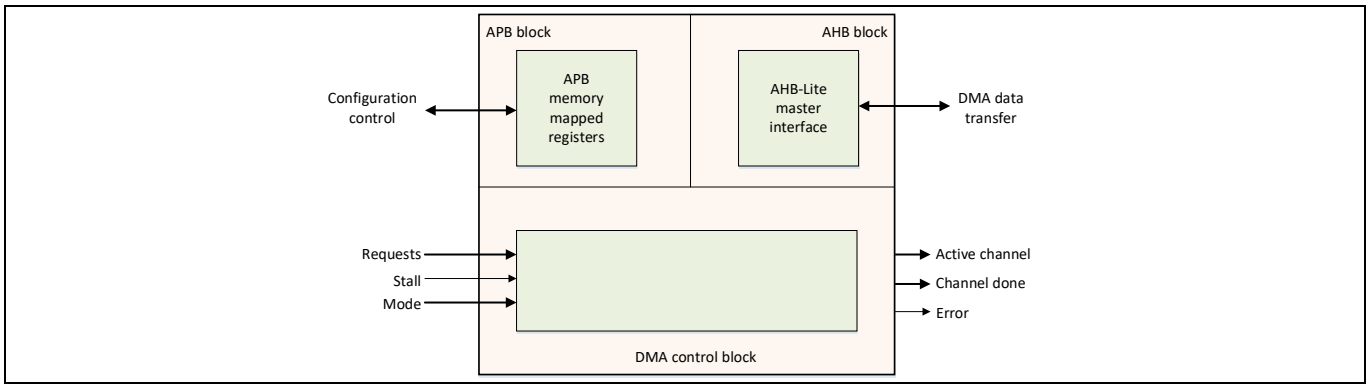

Figure 112 DMA block diagram

The APB block contains the registers that enable the configuration of the controller by using the APB slave interface.

The controller AHB block contains a single AHB-Lite master that enables it to transfer data from a source AHB slave to a destination AHB slave using a 32-bit data bus. The controller is compliant with the AMBA® 3 AHB-Lite protocol. For detailed information about the AHB-Lite interface, see the AMBA® 3 AHB-Lite Protocol v1.0 Specification.

The DMA control block contains the control logic that provides the following features:

- It arbitrates the incoming requests
- It indicates which channel is active
- It indicates when a channel is complete
- It indicates when an error has occurred on the AHB-Lite interface
- It enables slow peripherals to stall the completion of a DMA cycle
- It waits for a request to clear before completing a DMA cycle
- It performs multiple or single DMA transfers for each request
- It performs the supported types of DMA transfers

15.6.2 DMA memory map

The memory map accessible by DMA is described in Table 103.

Table 103 DMA memory map

Address range	Size	Peripheral	Bus matrix master
0000_0000h - 2004_FFFFh	–	Reserved	–
2005_0000h - 2005_3FFFh	16 kB	RAM1 (replica)	M2
2005_4000h - 2005_7FFFh	16 kB	RAM1 (replica)	
2005_8000h - 2005_BFFFh	16 kB	RAM1 (replica)	
2005_C000h - 2005_FFFFh	16 kB	RAM1	
2006_0000h - 2006_3FFFh	16 kB	RAM2	M1
2006_4000h - 2006_7FFFh	16 kB	RAM2 (replica)	
2006_8000h - 2006_BFFFh	16 kB	RAM2 (replica)	
2006_C000h - 2006_FFFFh	16 kB	RAM2 (replica)	
2007_0000h - 2FFF_FFFFh	–	Reserved	–
3000_0000h - 3001_3FFFh	80 kB	ROM	M0
3001_4000h - 3001_FFFFh	48 kB	Reserved	
3002_0000h - 3002_FFFFh	64 kB	OTP	
3003_0000h - 3003_FFFFh	64 kB	OTP (replica)	
3004_0000h - 6FFF_FFFFh	–	Reserved	–
7000_0000h - 7007_FFFFh	–	BIF REGFILE (control)	M6
7008_0000h - 700F_FFFFh	–	PMBus/CRC/I^2C	M7
7010_0000h - FFFF_FFFFh	–	Reserved	–

Remapping does not impact the DMA memory map.

15.6.3 DMA channel assignment

DMA channel assignment is shown in Table 104.

The DMA macro supports single transfer requests (SREQs) and multiple transfer requests (REQs), but multiple transfer requests are currently not supported by the XDPP1100.

DMA channel 0 has the highest priority, channel 15 the lowest.

Table 104 DMA channel assignment table

Request line	DMA SREQ channel	DMA REQ channel
0	N/A	N/A
1	N/A	N/A
2	N/A	N/A
3	N/A	N/A
4	N/A	N/A
5	N/A	N/A

Request line	DMA SREQ channel	DMA REQ channel
6	DTIMER1_1	N/A
7	DTIMER1_2	N/A
8	DTIMER3_1	N/A
9	DTIMER3_2	N/A
10	I2C_TX	N/A
11	I2C_RX	N/A
12	N/A	N/A
13	N/A	N/A
14	UART_TX	N/A
15	UART_RX	N/A

15.6.4 DMA registers

The relevant DMA-related registers and their descriptions are provided in Table 105.

Table 105 DMA-related register description

Register name	Field name	Access	Address	Bits	Description
DMA_STATUS	MASTER_ENABLE	R	5000_0000h	[0]	Enable status of the controller. 0: Controller is disabled 1: Controller is enabled
DMA_STATUS	RES0	R	5000_0000h	[3:1]	Reserved
DMA_STATUS	STATE	R	5000_0000h	[7:4]	Current state of the control state machine. State can be one of the following bit combinations (values not listed are undefined): 0: Idle 1: Reading channel controller date 2: Reading source data end pointer 3: Reading destination data end pointer 4: Reading source data 5: Writing destination data

Register name	Field name	Access	Address	Bits	Description
					6: Waiting for DMA request to clear 7: Writing channel controller data 8: Stalled 9: Done 10: Peripheral scatter-gather transition
DMA_STATUS	RES1	R	5000_0000h	[15:8]	Reserved
DMA_STATUS	CHNLS_MINUS1	R	5000_0000h	[20:15]	Returns the number of available channels minus 1. For example: 0: 1 available channel 15: 16 available channels
DMA_STATUS	RES2	R	5000_0000h	[31:21]	Reserved
DMA_CFG	MASTER_ENABLE	W	5000_0004h	[0]	Enable for the controller. 0: Disable controller 1: Enable controller
DMA_CFG	RES0	W	5000_0004h	[4:1]	Reserved, write as 0
DMA_CFG	CHN1_PROT_CTRL	W	5000_0004h	[7:5]	Sets the AHB-Lite protection by controlling the HPROT[3:1] signal levels as follows: [2] Controls HPROT[3] to indicate if a cacheable access is occurring [1] Controls HPROT[2] to indicate if a bufferable access is occurring [0] Controls HPROT[1] to indicate if a privileged access is occurring Notes: 1. When bit[n]=1, then the

Register name	Field name	Access	Address	Bits	Description
					corresponding H_{PROT} is high 2. When bit[n]=0, then the corresponding H_{PROT} is low 3. The CHN1_PROT_CTRL bits must not be changed when the MASTER_ENABLE bit is set because this may cause a protocol error on the AHB master interface. As the DMA_CFG register is write-only, the user must read the status of the master enable bit from the DMA_STATUS register.
DMA_CFG	RES1	W	5000_0004h	[31:8]	Reserved, write as 0
DMA_CTRL_BASE_PTR	CTRL_BASE_PTR	RW	5000_0008h	[31:9]	Pointer to the base address of the primary data structure
DMA_ALT_CTRL_BASE_PTR	ALT_CTRL_BASE_PTR	R	5000_000Ch	[31:0]	Returns the base address of the alternate data structure
CHNL_WAITONREQ_STATUS	DMA_WAITONREQ_STATUS	R	5000_0010h	[15:0]	Channel wait on request status. For each bit [x]: 0: DMA channel x wait on request is low 1: DMA channel x wait on request is high
DMA_CHNL_SW_REQUEST	CHNL_SW_REQUEST	W	5000_0014h	[15:0]	The write-only CHNL_SW_REQUEST register enables you to generate a software DMA

Register name	Field name	Access	Address	Bits	Description
					request. Set the appropriate bit to generate a software DMA request on the corresponding DMA channel. For each bit [x]: 0: Does not create a DMA request for channel x 1: Creates a DMA request for channel x Note: Writing to a bit where a DMA channel is not implemented does not create a DMA request for that channel.
DMA_CHNL_USEBURST_SET	CHNL_USEBURST_SET	RW	5000_0018h	[15:0]	Returns the use burst status, or disables the single request dma_sreq input from generating DMA requests. For each bit [x], On READ: 0: DMA channel x responds to requests that it receives on dma_req or dma_sreq. The controller performs both 2^R and single bus transfers. 1: DMA channel x does not respond to requests that it receives on dma_sreq. It only responds to dma_req requests and performs 2^R transfers. On WRITE: 0: No effect. Use CHNL_USEBURST_CLR to enable

Register name	Field name	Access	Address	Bits	Description
					dma_sreq requests on channel x. 1: Disables dma_sreq from generating DMA requests on channel x. The controller performs 2^R transfers only.
DMA_CHNL_USEBURST_CLR	CHNL_USEBURST_CLR	W	5000_001Ch	[15:0]	This write-only register enables dma_sreq to generate requests. For each bit [x]: 0: No effect. Use CHNL_USEBURST_SET to disable dma_sreq requests on channel x. 1: Enables dma_sreq to generate DMA requests on channel x
DMA_CHNL_REQ_MASK_SET	CHNL_REQ_MASK_SET	RW	5000_0020h	[15:0]	Returns the request mask status of dma_req and dma_sreq, or disables the corresponding channel from generating DMA requests. For each bit [x], On READ: 0: External requests are enabled for channel x 1: External requests are disabled for channel x On WRITE: 0: No effect. Use the CHNL_REQ_MASK_CLR Register to enable DMA requests. 1: Disables dma_req and dma_sreq on

Register name	Field name	Access	Address	Bits	Description
					channel x from generating DMA requests
DMA_CHNL_REQ_MASK_CLR	CHNL_REQ_MASK_CLR	W	5000_0024h	[15:0]	The write-only register enables DMA request on a per-channel basis. For each bit [x]: 0: No effect. Use the CHNL_REQ_MASK_SET Register to disable dma_req and dma_sreq from generating requests. 1: Enables dma_req or dma_sreq to generate DMA requests on channel x
DMA_CHNL_ENABLE_SET	CHNL_ENABLE_SET	RW	5000_0028h	[15:0]	Returns the enable status of the channels, or enables the corresponding channels. For each bit [x], On READ: 0: Channel x is disabled 1: Channel x is enabled On WRITE: 0: No effect. Use the CHNL_ENABLE_CLR Register to disable a channel. 1: Enables channel x
DMA_CHNL_ENABLE_CLR	CHNL_ENABLE_CLR	W	5000_002Ch	[15:0]	This write-only register disables DMA channels by clearing the enable. For each bit [x]: 0: No effect. Use the CHNL_ENABLE_SET Register to enable DMA channels. 1: Disables channel x

Register name	Field name	Access	Address	Bits	Description
DMA_CHNL_PRI_ALT_SET	CHNL_PRI_ALT_SET	RW	5000_0030h	[15:0]	This register enables you to configure a DMA channel to use the alternate data structure. Reading the register returns the status of which data structure is in use for the corresponding DMA channel. For each bit [x], On READ: 0: DMA channel x is using the primary data structure 1: DMA channel x is using the alternate data structure On WRITE: 0: No effect. Use the CHNL_PRI_ALT_CLR Register to set bit [x] to 0. 1: Selects the alternate data structure for channel x
DMA_CHNL_PRI_ALT_CLR	CHNL_PRI_ALT_CLR	W	5000_0034h	[15:0]	This register enables you to configure a DMA channel to use the primary data structure. For each bit [x]: 0: No effect. Use the CHNL_PRI_ALT_SET Register to select the alternate data structure. 1: Selects the primary data structure for channel x
DMA_CHNL_PRIORITY_SET	CHNL_PRIORITY_SET	RW	5000_0038h	[15:0]	This register enables you to configure a DMA channel to use

Register name	Field name	Access	Address	Bits	Description
					the high priority level. Reading the register returns the status of the channel priority mask. For each bit [x], On READ: 0: DMA channel x is using the default priority level 1: DMA channel x is using a high priority level On WRITE: 0: No effect. Use the CHNL_ENABLE_CLR Register to set channel x to the default priority level. 1: Channel x uses the high priority level
DMA_CHNL_PRIORITY_CLR	CHNL_PRIORITY_CLR	W	5000_003Ch	[15:0]	This write-only register enables you to configure a DMA channel to use the default priority level. For each bit [x]: 0: No effect. Use the CHNL_ENABLE_SET Register to set channel x to the high priority level. 1: Channel x uses the default priority level
DMA_ERR_CLR	ERR_CLR	RW	5000_004Ch	[0]	The read/write ERR_CLR register returns the status of dma_err, and enables you to set dma_err low. On READ: 0: dma_err is low 1: dma_err is high On WRITE:

Register name	Field name	Access	Address	Bits	Description
					0: No effect, status of dma_err is unchanged 1: Sets dma_err low
RIS	DMA_CH_INT_STATUS	R	5000_1000h	[15:0]	Raw DMA channel interrupt status prior to masking by SMA_CH_MASK. For each bit [x]: 0: No interrupt pending on DMA channel x 1: Interrupt pending on DMA channel x
RIS	DMA_ERR_INT_STATUS	R	5000_1000h	[31]	Raw DMA error interrupt status prior to masking by SMA_ERR_MASK. 0: No error interrupt pending 1: Error interrupt pending
IMSC	DMA_CH_MSK	RW	5000_1004h	[15:0]	DMA channel interrupt mask. For each bit [x]: 0: DMA channel x interrupt disabled 0: DMA channel x interrupt enabled
IMSC	DMA_ERR_MSK	RW	5000_1004h	[31]	Mask bit for DMA error interrupt. 0: DMA error interrupt disabled 1: DMA error interrupt enabled
MIS	DMA_CH_INT	R	5000_1008h	[15:0]	DMA channel interrupt status after masking by SMA_CH_MASK. For each bit [x]: 0: DMA channel x interrupt disabled or not pending

Register name	Field name	Access	Address	Bits	Description
					1: DMA channel x interrupt enabled and pending
MIS	DMA_ERR_INT	R	5000_1008h	[31]	DMA error interrupt status after masking by SMA_ERR_MASK. 0: DMA error interrupt disabled or not pending 1: DMA error interrupt enabled and pending
ICR	DMA_CH_INT_CLR	W	5000_100Ch	[15:0]	DMA channel interrupts clear. For each bit [x]: 0: No change to DMA channel x interrupt 1: Clear DMA channel x interrupt
ICR	DMA_ERR_INT_CLR	W	5000_100Ch	[31]	DMA error interrupt clear. 0: No change to DMA error interrupt 1: Clear DMA error interrupt
ISR	DMA_CH_INT_SET	W	5000_1010h	[15:0]	DMA channel interrupts set. For each bit [x]: 0: No change to DMA channel x interrupt 1: Set DMA channel x interrupt
ISR	DMA_ERR_INT_SET	W	5000_1010h	[31]	DMA error interrupt set. 0: No change to DMA error interrupt 1: Set DMA error interrupt

15.7 General-purpose input output (GPIO) module

The CPUS provides a number of GPIO pins, which are organized into blocks of eight GPIO pins each. The XDPP1100 controller contains two GPIO blocks (identical instance, replicated twice): GPIO0 and GPIO1.

Each GPIO register set can be accessed with a different base address:

- GPIO_0: 6004_0000h
- GPIO_1: 6005_0000h

GPIO pin mapping on primary XDPP1100 IOs is summarized in Table 86.

The GPIO block is an Arm® PrimeCell IP (PL061); extensive documentation can be found in the “Arm® PrimeCell General Purpose Input/Output (PL061) Technical Reference Manual”.

The main features of the GPIO block are:

- Eight individually programmable input/output pins, default to input at reset
- HW control capability of GPIO lines for different system configurations
- Bit masking in both read and write operations through address lines
- Programmable interrupt generation capability, from a transition or a level condition, on any number of pins

15.7.1 GPIO registers

The relevant GPIO-related registers and their descriptions are provided in Table 106.

Table 106 GPIO register description

Register name	Field name	Access	Address	Bits	Description
GPIODATA	DATA	RW	6004_0000h 6005_0000h	[7:0]	The GPIODATA register is the data register. In software control mode, values written in the GPIODATA register are transferred onto the GPOUT pins if the respective pins have been configured as outputs through the GPIODIR register. In order to write to GPIODATA, the corresponding bits in the mask, resulting from the address bus PADDR[9:2], must be high. Otherwise, the bit values remain unchanged by the write. Similarly, the values read from this register are determined for each bit by the mask bit derived from the address used to access the data register, PADDR[9:2]. Bits that are 1 in the address mask cause the corresponding bits in GPIODATA to be read, and bits that are 0 in the address mask cause the corresponding bits in GPIODATA to be read as 0, regardless of their value. A read from GPIODATA returns the last bit value written if the respective pins are configured as output, or it returns the value on

Register name	Field name	Access	Address	Bits	Description
					the corresponding input GPIN bit when these are configured as inputs. All bits are cleared by a reset.
GPIODIR	DIR	RW	6004_0400h 6005_0400h	[7:0]	The GPIODIR register is the data direction register. Bits set to high in the GPIODIR configure the corresponding pin to be an output. Clearing a bit configures the pin to be input. All bits are cleared by a reset. Therefore, the GPIO pins are input by default. For each bit [x]: 0: GPIOx configured as input 1: GPIOx configured as output
GPIOIS	INTSENSE	RW	6004_0404h 6005_0404h	[7:0]	The GPIOIS register is the interrupt sense register. Bits set to high in GPIOIS configure the corresponding pins to detect levels. Clearing a bit configures the pin to detect edges. All bits are cleared by a reset. For each bit [x]: 0: GPIOx is edge sensitive 1: GPIOx is level sensitive
GPIOIBE	INTEDGE	RW	6004_0408h 6005_0408h	[7:0]	The GPIOIBE register is the interrupt both edges register. When the corresponding bit in GPIOIS is set to detect edges, bits set to high in GPIOIBE configure the corresponding pin to detect both rising and falling edges, regardless of the corresponding bit in the GPIOIEV (interrupt event register). Clearing a bit configures the pin to be controlled by GPIOIEV. All bits are cleared by a reset. For each bit [x]: 0: GPIOx interrupt controlled by GPIOIEV register 1: GPIOx interrupt on both edges
GPIOIEV	INTEVENT	RW	6004_040Ch 6005_040Ch	[7:0]	The GPIOIEV register is the interrupt event register. Bits set to high in GPIOIEV configure the corresponding pin to detect rising edges or high levels, depending on the corresponding bit value in

Register name	Field name	Access	Address	Bits	Description
					GPIOIS. Clearing a bit configures the pin to detect falling edges or low levels, depending on the corresponding bit value in GPIOIS. All bits are cleared by a reset. For each bit [x]: 0: GPIOx interrupt on rising edge or high level based on GPIOIS 1: GPIOx interrupt on falling edge or low level based on GPIOIS
GPIOIE	INTENA	RW	6004_0410h 6005_0410h	[7:0]	The GPIOIE register is the interrupt enable register. Bits set to high in GPIOIE allow the corresponding pins to trigger their individual interrupts and the combined GPIOINTR line. Clearing a bit disables interrupt triggering on that pin. All bits are cleared by a reset. For each bit [x]: 0: GPIOx interrupt is disabled 1: GPIOx interrupt is enabled
GPIORIS	INTRAW	R	6004_0414h 6005_0414h	[7:0]	The GPIORIS register is the raw interrupt status register. Bits read high in GPIORIS reflect the status of interrupt trigger conditions detected (raw, prior to masking by GPIOIE.INTENA), indicating that all the requirements have been met, before they are finally allowed to trigger by GPIOIE. Bits read as zero indicate that corresponding input pins have not initiated an interrupt. This register is read-only, and bits are cleared by a reset. For each bit [x]: 0: GPIOx interrupt conditions not met prior to masking 1: GPIOx interrupt conditions met prior to masking
GPIOMIS	INTMASK	R	6004_0418h 6005_0418h	[7:0]	The GPIOMIS register is the masked interrupt status register. Bits read high in GPIOMIS reflect the status of input lines triggering an interrupt. Bits read as low indicate that either no interrupt has been generated, or the interrupt is masked (disabled)

Register name	Field name	Access	Address	Bits	Description
					by GPIOIE.INTENA. This register is read-only, and all bits are cleared by a reset. For each bit [x]: 0: GPIOx interrupt disabled or conditions not met 1: GPIOx interrupt enabled and conditions met
GPIOIC	INTCLR	W	6004_041Ch 6005_041Ch	[7:0]	The GPIOIC register is the interrupt clear register. Writing a 1 to a bit in this register clears the corresponding interrupt edge detection logic register. Writing a 0 has no effect. This register is write-only and all bits are cleared by a reset. For each bit [x]: 0: GPIOx interrupt status unchanged 1: GPIOx interrupt edge detect logic cleared
GPIOAFSEL	MSEL	RW	6004_0420h 6005_0420h	[7:0]	The GPIOAFSEL register is the mode control select register. Writing a 1 to any bit in this register selects the hardware control for the corresponding PrimeCell GPIO line. All bits are cleared by a reset, therefore no PrimeCell GPIO line is set to hardware control by default. For each bit [x]: 0: GPIOx software control mode enabled 1: GPIOx hardware control mode enabled
GPIOPeriphID0	PartNumber0	R	6004_0FE0h 6005_0FE0h	[7:0]	PartNumber[7:0]. Together with the upper bits from PartNumber1, PartNumber[11:0] identifies the peripheral. In this case the three-digit product code, 0x061, is returned.
GPIOPeriphID1	PartNumber1	R	6004_0FE4h 6005_0FE4h	[3:0]	PartNumber[11:8]. Together with the lower bits from PartNumber0, PartNumber[11:0] identifies the peripheral. In this case the three-digit product code, 0x061, is returned.

Register name	Field name	Access	Address	Bits	Description
GPIOPeriphID1	Designer0	R	6004_0FE4h 6005_0FE4h	[7:4]	DesignerID[3:0]. Together with the upper bits from Designer1, DesignerID[7:0] identifies the peripheral designer. In this case it returns 0x41, indicating Arm® Ltd.
GPIOPeriphID2	Designer1	R	6004_0FE8h 6005_0FE8h	[3:0]	DesignerID[7:4]. Together with the lower bits from Designer0, DesignerID[7:0] identifies the peripheral designer. In this case it returns 0x41, indicating Arm® Ltd.
GPIOPeriphID2	Revision	R	6004_0FE8h 6005_0FE8h	[7:4]	Returns the peripheral revision number with 0 indicating the initial revision
GPIOPeriphID3	Configuration	R	6004_0FECh 6005_0FECh	[7:0]	Returns the configuration option of the peripheral
GPIOPCellID0	GPIOPCellID0	R	6004_0FF0h 6005_0FF0h	[7:0]	CellID[7:0]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID system. In this case CellID[31:0] = 0xB105F00D
GPIOPCellID1	GPIOPCellID1	R	6004_0FF4h 6005_0FF4h	[7:0]	CellID[15:8]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID system. In this case CellID[31:0] = 0xB105F00D
GPIOPCellID2	GPIOPCellID2	R	6004_0FF8h 6005_0FF8h	[7:0]	CellID[23:16]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID system. In this case CellID[31:0] = 0xB105F00D
GPIOPCellID3	GPIOPCellID3	R	6004_0FFCh 6005_0FFCh	[7:0]	CellID[31:24]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID system. In this case CellID[31:0] = 0xB105F00D

15.8 WDT module

The watchdog block is an Arm® IP cell from the Arm® AMBA® Design Kit; extensive documentation can be found in the “Arm® AMBA® Design Kit Technical Reference Manual”, paragraph 4.4 “Watchdog unit”.

The watchdog module is based around a 32-bit down counter that is initialized from the reload register, WDOGLOAD.

The watchdog clock generates a regular interrupt, WDOGINT, depending on a programmed value. The counter decrements by one on each positive clock edge of WDOGCLK when the clock enable WDOGCLKEN is high. The watchdog asserts an interrupt when the counter reaches zero, then reloads the counter and starts another

down count. If the interrupt is not cleared by the time the counter next reaches zero, then the watchdog module asserts the reset signal.

The watchdog module is intended to be used to apply a reset to a system in the event of software failure, providing a way of recovering from software crashes.

The watchdog unit has a protection mechanism for which registers are locked (not writable), to protect its integrity in case of FW malfunctioning events. Protection is entered using the password register.

The watchdog unit can be enabled or disabled as required.

15.8.1 Watchdog block diagram

The watchdog block structure is shown in Figure 113.

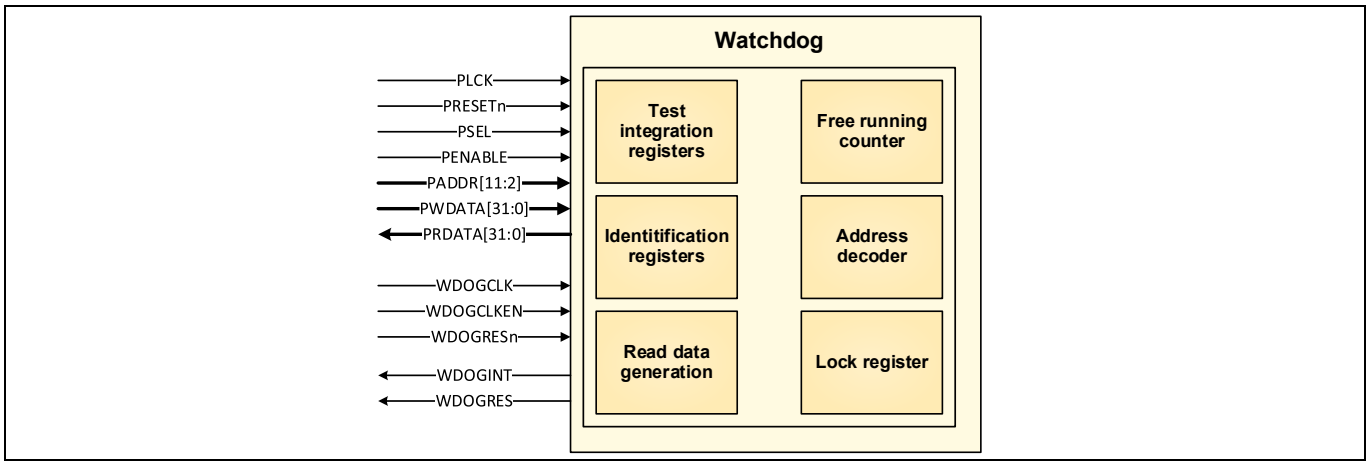

Figure 113 Watchdog block structure

15.8.2 Watchdog registers

The relevant watchdog-related registers and their descriptions are provided in Table 107.

Table 107 WDT-related register descriptions

Register name	Field name	Access	Address	Bits	Description
WDOGLOAD	LOAD	RW	6000_0000h	[31:0]	Watchdog load register. WDT counter decrements from this value. When this register is written to, the counter starts decrementing immediately. The minimum value write value is “1”
WDOGVALUE	VALUE	R	6000_0004h	[31:0]	Returns the current value of the watchdog counter
WDOGCONTROL	INTEN	RW	6000_0008h	[0]	Watchdog interrupt enable. Enables interrupt event, WDOGINT. A low to high transition reloads the watchdog decrement counter with the value configured in the WDOGLOAD register.

Register name	Field name	Access	Address	Bits	Description
					0: Disable the decrement counter and the interrupt 1: Enable the decrement counter and the interrupt
WDOGCONTROL	RESEN	RW	6000_0008h	[1]	Watchdog reset output enable. Enables the watchdog reset output signal. Acts as a mask for the reset output. 0: Disable reset output 1: Enable reset output
WDOGINTCLR	INTCLR	W	6000_000Ch	[31:0]	Interrupt clear. This write-only register clears the watchdog interrupt, and reloads the counter from the value in WDOGLOAD
WDOGRIS	RAWINT	R	6000_0010h	[0]	Raw interrupt status. Reflects the status of the interrupt prior to masking. 0: Interrupt from counter has not been generated 1: Interrupt from counter has been generated
WDOGMIS	INTMASK	R	6000_0014h	[0]	Masked interrupt status. Indicates the masked interrupt status which is the logical AND of the raw interrupt status and WDOGCONTROL.INTEN. 0: Masked interrupt not asserted 1: Masked interrupt asserted
WDOGLOCK	LOCKEN	RW	6000_0C00h	[0]	Watchdog write protection control. When locked, other registers in the watchdog peripheral cannot be written. On read: 0: Unlocked 1: Locked On write: 0x1ACCE551: Unlock Any other value: Lock
WDOGITCR	ITME	RW	6000_0F00h	[0]	Integration test mode enable. When integration test mode is enabled, the masked interrupt output, WDOGINT, and reset output, WDOGRES, are directly controlled by the test output set register, WDOGITOP.

Register name	Field name	Access	Address	Bits	Description
					0: Integration test mode disabled 1: Integration test mode enabled
WDOGITOP	ITRES	W	6000_0F04h	[0]	Integration test WDOGRES value. Sets the value of the WDOGRES signal when in integration test mode. 0: Set WDOGRES low 1: Set WDOGRES high
WDOGITOP	ITINT	W	6000_0F04h	[1]	Integration test WDOGINT value. Sets the value of the WDOGINT signal when in integration test mode. 0: Set WDOGINT low 1: Set WDOGINT high
WDOGPERIPHID0	PartNumber0	R	6000_0FE0h	[7:0]	PartNumber[7:0]. Together with the upper bits from PartNumber1, PartNumber[11:0] identifies the peripheral. In this case the three-digit product code, 0x061, is returned
WDOGPERIPHID1	PartNumber1	R	6000_0FE4h	[3:0]	PartNumber[11:8]. Together with the lower bits from PartNumber0, PartNumber[11:0] identifies the peripheral. In this case the three-digit product code, 0x061, is returned
WDOGPERIPHID1	Designer0	R	6000_0FE4h	[7:4]	DesignerID[3:0]. Together with the upper bits from Designer1, DesignerID[7:0] identifies the peripheral designer. In this case it returns 0x41, indicating Arm® Ltd
WDOGPERIPHID2	Designer1	R	6000_0FE8h	[3:0]	DesignerID[7:4]. Together with the lower bits from Designer0, DesignerID[7:0] identifies the peripheral designer. In this case it returns 0x41, indicating Arm® Ltd
WDOGPERIPHID2	Revision	R	6000_0FE8h	[7:4]	Returns the peripheral revision number with 0 indicating the initial revision
WDOGPERIPHID3	Configuration	R	6000_0FECh	[7:0]	Returns the configuration option of the peripheral
WDOGPCELLID0	WDOGPCELLID0	R	6000_0FF0h	[7:0]	CellID[7:0]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID

Register name	Field name	Access	Address	Bits	Description
					system. In this case CellID[31:0] = 0xB105F00D
WDOGPCELLID1	WDOGPCELLID1	R	6000_0FF4h	[7:0]	CellID[15:8]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID system. In this case CellID[31:0] = 0xB105F00D
WDOGPCELLID2	WDOGPCELLID2	R	6000_0FF8h	[7:0]	CellID[23:16]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID system. In this case CellID[31:0] = 0xB105F00D
WDOGPCELLID3	WDOGPCELLID3	R	6000_0FFCh	[7:0]	CellID[31:24]. Together with the other cell ID registers, CellID[31:0] is used as a standard cross-peripheral ID system. In this case CellID[31:0] = 0xB105F00D

15.9 Dual-timer (D_{TIMER}) modules

XDP1100 includes three identical dual-timer blocks, each of which has the same register set, described in Table 108, with a different base address:

- **D_{TIMER_1}:** 6001_0000h
- **D_{TIMER_2}:** 6002_0000h
- **D_{TIMER_3}:** 6003_0000h

The dual-timer block is an Arm® IP cell from the Arm® AMBA® Design Kit; extensive documentation can be found in the “Arm® AMBA® Design Kit Technical Reference Manual”, paragraph 4.5 “Dual input timer”.

The dual-timer module consists of two programmable 32/16-bit down counters that can generate interrupts on reaching zero. A timer module can be programmed for a 32-bit or 16-bit counter size and one of three timer modes using the control register. The operation of each timer module is identical. It has one of three timer modes:

- **Free-running:** The counter wraps after reaching its zero value and continues to count down from the maximum value. This is the default mode
- **Periodic:** The counter generates an interrupt at a constant interval, reloading the original value after wrapping past zero
- **One-shot:** The counter generates an interrupt once. When the counter reaches zero, it halts until it is reprogrammed by the user. This can be achieved by either clearing the one-shot count bit in the control register, in which case the count proceeds according to the selection of free-running or periodic mode, or by writing a new value to the load value register

The dual-input timers module provides access to two interrupt-generating, programmable 32-bit free-running decrementing counters (FRCs). The FRCs operate from a common timer clock, TIMCLK, with each FRC having its own clock enable input, TIMCLKEN1 and TIMCLKEN2. Each FRC also has a prescaler that can divide the enabled TIMCLK rate by 1, 16 or 256. This enables the count rate for each FRC to be controlled independently using their individual clock enables and prescalers.

15.9.1 Dual-timer block diagram

The dual-timer block diagram is shown in Figure 114.

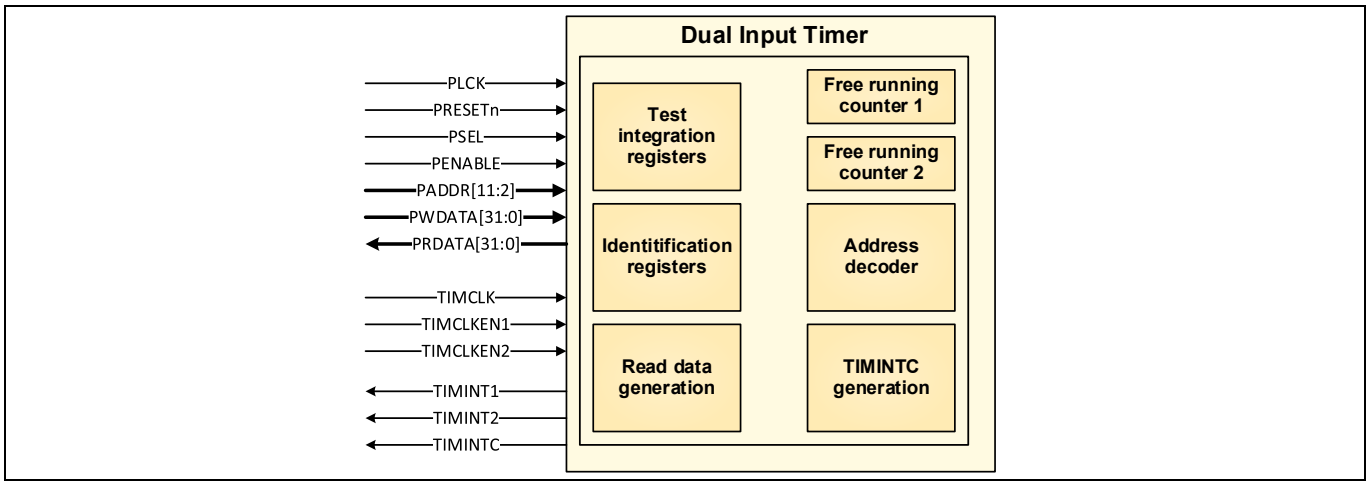

Figure 114 Dual-timer block diagram

15.9.2 Dual-timer registers

The relevant D_{TIMER}-related registers and their descriptions are provided in Table 108.

Table 108 D_{TIMER}-related register descriptions

Register name	Field name	Access	Address	Bits	Description
TIM_SEQ0_TIMERLOAD	LOAD	RW	6001_0000h 6002_0000h 6003_0000h	[31:0]	TIM_SEQ0_TIMERLOAD contains the value from which the counter is to decrement. This is the value used to reload the counter when periodic mode is enabled and the current count reaches zero. When this register is written to directly, the current count is immediately reset to the new value at the rising edge of the enabled clock. The value in this register is also overwritten if the TIM_SEQ0_TIMERBGLOAD register is written to, but the current count is not immediately affected. If values are written to both the TIM_SEQ0_TIMERLOAD and TIM_SEQ0_TIMERBGLOAD

Register name	Field name	Access	Address	Bits	Description
					registers before an enabled rising edge on TIMCLK, the following occurs: On the next enabled clock edge, the value written to the TIM_SEQ0_TIMERLOAD value replaces the current count value. Then each time the counter reaches zero, the current count value is reset to the value written to TIM_SEQ0_TIMERBGLOAD. Reading from the TIM_SEQ0_TIMERLOAD register at any time after the two writes have occurred retrieves the value written to TIM_SEQ0_TIMERBGLOAD. The value read from TIM_SEQ0_TIMERLOAD is always the value that takes effect for periodic mode after the next time the counter reaches zero.
TIM_SEQ0_TIMERVALUE	VALUE	R	6001_0004h 6002_0004h 6003_0004h	[31:0]	Returns the current value of the decrementing counter.
TIM_SEQ0_TIMERCONTROL	ONESHOT	RW	6001_0008h 6002_0008h 6003_0008h	[0]	One-shot count. Select one-shot or wrapping counter mode. 0: Wrapping mode, default 1: One-shot mode
TIM_SEQ0_TIMERCONTROL	TSIZE	RW	6001_0008h 6002_0008h 6003_0008h	[1]	Timer size. Select 16/32-bit counter operation. 0: 16-bit counter, default 1: 32-bit counter
TIM_SEQ0_TIMERCONTROL	TPRE	RW	6001_0008h 6002_0008h 6003_0008h	[3:2]	Timer prescaler. Set the prescaler of the timer. 0: Timer clock divided by 1 (default) 1: Timer clock divided by 16 2: Timer clock divided by 256

Register name	Field name	Access	Address	Bits	Description
					3: Undefined, do not use
TIM_SEQ0_TIMERCONTROL	INTEN	RW	6001_0008h 6002_0008h 6003_0008h	[5]	Timer interrupt enable. 0: Timer interrupt disabled 1: Timer interrupt enabled
TIM_SEQ0_TIMERCONTROL	TMODE	RW	6001_0008h 6002_0008h 6003_0008h	[6]	Timer operation mode. 0: Free-running mode 1: Periodic mode
TIM_SEQ0_TIMERCONTROL	TEN	RW	6001_0008h 6002_0008h 6003_0008h	[7]	Timer enable. 0: Timer disabled 1: Timer enabled
TIM_SEQ0_TIMERINTCLR	CLRINT	W	6001_000Ch 6002_000Ch 6003_000Ch	[31:0]	Interrupt clear. A write of any value to this write-only register clears the counter interrupt.
TIM_SEQ0_TIMERRIS	RAWINT	R	6001_0010h 6002_0010h 6003_0010h	[0]	Raw interrupt status. Reflects the raw interrupt status from the counter prior to masking by INTEN. 0: Raw interrupt not asserted 1: Raw interrupt asserted
TIM_SEQ0_TIMERMIS	STINT	R	6001_0014h 6002_0014h 6003_0014h	[0]	Timer interrupt status. Reflects the counter interrupt status after masking by INTEN. 0: Interrupt disabled or not asserted 1: Interrupt enabled and asserted
TIM_SEQ0_TIMERBGLOAD	BVALUE	RW	6001_0018h 6002_0018h 6003_0018h	[31:0]	TIM_SEQ0_TIMERBGLOAD contains the value used to reload the counter when periodic mode is enabled, and the current count reaches zero. This register provides an alternative method of accessing the TIM_SEQ0_TIMERLOAD register. The difference is that writes to TIM_SEQ0_TIMERBGLOAD do not cause the counter to immediately restart from the new value.

Register name	Field name	Access	Address	Bits	Description
					Reading from this register returns the same value returned from TIM_SEQ0_TIMERLOAD.
TIM_SEQ1_TIMERLOAD	LOAD	RW	6001_0020h 6002_0020h 6003_0020h	[31:0]	TIM_SEQ1_TIMERLOAD contains the value from which the counter is to decrement. This is the value used to reload the counter when periodic mode is enabled and the current count reaches zero. When this register is written to directly, the current count is immediately reset to the new value at the rising edge of the enabled clock. The value in this register is also overwritten if the TIM_SEQ1_TIMERBGLOAD register is written to, but the current count is not immediately affected. If values are written to both the TIM_SEQ1_TIMERLOAD and TIM_SEQ1_TIMERBGLOAD registers before an enabled rising edge on TIMCLK, the following occurs: On the next enabled clock edge, the value written to the TIM_SEQ1_TIMERLOAD value replaces the current count value. Then each time the counter reaches zero, the current count value is reset to the value written to TIM_SEQ1_TIMERBGLOAD. Reading from the TIM_SEQ1_TIMERLOAD register at any time after the two writes have occurred retrieves the value written to

Register name	Field name	Access	Address	Bits	Description
					TIM_SEQ1_TIMERBGLOAD. The value read from TIM_SEQ1_TIMERLOAD is always the value that takes effect for periodic mode after the next time the counter reaches zero.
TIM_SEQ1_TIMERVALUE	VALUE	R	6001_0024h 6002_0024h 6003_0024h	[31:0]	Returns the current value of the decrementing counter.
TIM_SEQ1_TIMERCONTROL	ONESHOT	RW	6001_0028h 6002_0028h 6003_0028h	[0]	One-shot count. Select one-shot or wrapping counter mode. 0: Wrapping mode, default 1: One-shot mode
TIM_SEQ1_TIMERCONTROL	TSIZE	RW	6001_0028h 6002_0028h 6003_0028h	[1]	Timer size. Select 16/32-bit counter operation. 0: 16-bit counter, default 1: 32-bit counter
TIM_SEQ1_TIMERCONTROL	TPRE	RW	6001_0028h 6002_0028h 6003_0028h	[3:2]	Timer prescaler. Set the prescaler of the timer. 0: Timer clock divided by 1 (default) 1: Timer clock divided by 16 2: Timer clock divided by 256 3: Undefined, do not use
TIM_SEQ1_TIMERCONTROL	INTEN	RW	6001_0028h 6002_0028h 6003_0028h	[5]	Timer interrupt enable. 0: Timer interrupt disabled 1: Timer interrupt enabled
TIM_SEQ1_TIMERCONTROL	TMODE	RW	6001_0028h 6002_0028h 6003_0028h	[6]	Timer operation mode. 0: Free-running mode 1: Periodic mode
TIM_SEQ1_TIMERCONTROL	TEN	RW	6001_0028h 6002_0028h 6003_0028h	[7]	Timer enable. 0: Timer disabled 1: Timer enabled
TIM_SEQ1_TIMERINTCLR	CLRINT	W	6001_002Ch 6002_002Ch 6003_002Ch	[31:0]	Interrupt clear. A write of any value to this write-only register clears the counter interrupt.
TIM_SEQ1_TIMERRIS	RAWINT	R	6001_0030h 6002_0030h 6003_0030h	[0]	Raw interrupt status. Reflects the raw interrupt status from the counter prior to masking by INTEN.

Register name	Field name	Access	Address	Bits	Description
					0: Raw interrupt not asserted 1: Raw interrupt asserted
TIM_SEQ1_TIMERMIS	STINT	R	6001_0034h 6002_0034h 6003_0034h	[0]	Timer interrupt status. Reflects the counter interrupt status after masking by INTEN. 0: Interrupt disabled or not asserted 1: Interrupt enabled and asserted
TIM_SEQ1_TIMERBGLOAD	BVALUE	RW	6001_0038h 6002_0038h 6003_0038h	[31:0]	TIM_SEQ1_TIMERBGLOAD contains the value used to reload the counter when periodic mode is enabled, and the current count reaches zero. This register provides an alternative method of accessing the TIM_SEQ1_TIMERLOAD register. The difference is that writes to TIM_SEQ1_TIMERBGLOAD do not cause the counter to immediately restart from the new value. Reading from this register returns the same value returned from TIM_SEQ1_TIMERLOAD.

15.10 PMBus module

This chapter describes the PMBus module embedded in the CPUS.

The PMBus module implements an I^2C slave interface controller compliant with PMBus protocol profile. The PMBus supports an AMBA® APB compliant interface.

Figure 115 shows the top-level block diagram of the PMBus unit.

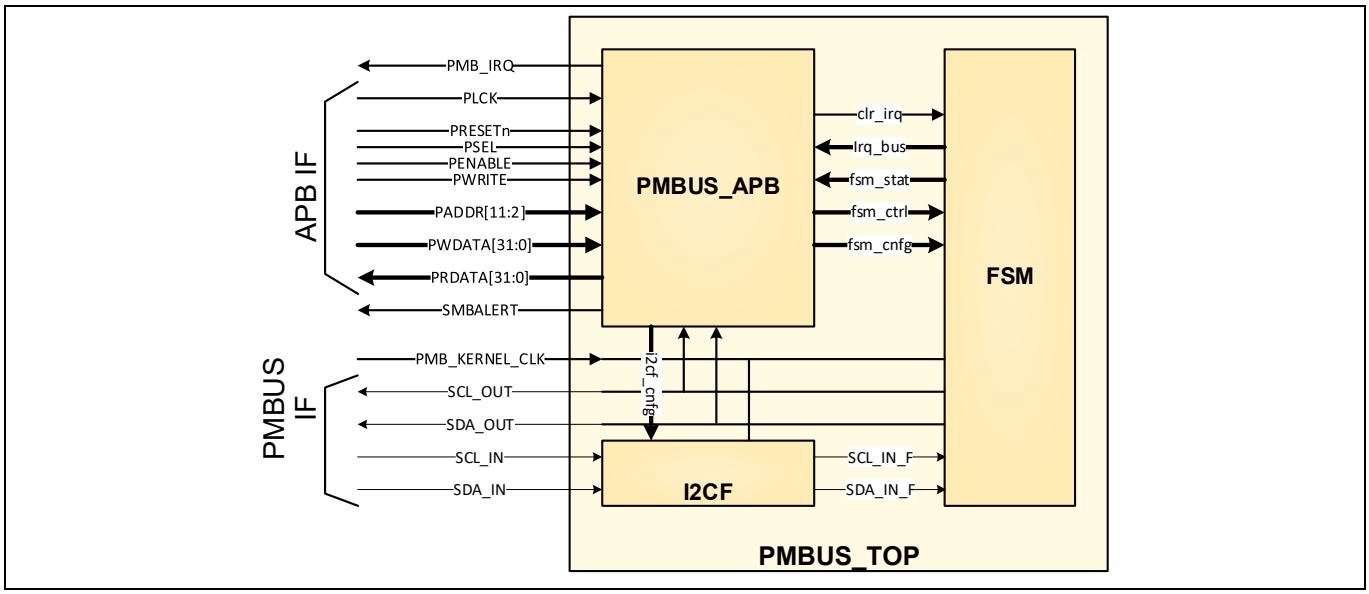

Figure 115 PMBus block diagram

The PMBus module manages the I²C/SMBus/PMBus transactions by using a CPU. The advantage of such an approach is to have a fully programmable, configurable, and extensible SMBus/PMBus interface that can support any kind of commands. The PMBus module is partitioned into three main blocks:

- PMB_I2CF block
- PMB_APB block
- PMB_FSM block

The PMBus module operation is enabled by enabling the dedicated kernel clock (pmbus_kernel_clk), by releasing the dedicated peripheral reset signal (pmbus_rst_n), by properly configuring the module registers through the APB interface, and by enabling the PMBus PHY to operate.

The PMB_I2CF block (I²C input signal formatter) has the duty to provide the debounced input signals, “sda_in” and “scl_in”, and the signals indicating input signal status change, to the PMB_FSM block (PMBUS PHY FSM). The length of the debouncing can be configured through the APB register. The PMB_I2CF also provides the “transaction_in_progress” signal, indicating when an I²C/PMBus transaction is in progress (signal will be asserted from the start event until the stop event). Finally, the PMB_I2CF operates only when the PHY is enabled.

The PMB_APB block implements the APB interface and contains all the CPU registers required to control and manage the PMBus interface (i.e., the PMB_FSM and the PMB_I2CF). The PMB_APB is also in charge of asserting and deasserting the SMBALERT signal. The PMB_APB embeds the logic to manage the service request generated by the PMB_FSM. Finally, the PMB_APB embeds a selectable feature to manage the ACK/NACK response without the need to stretch the clock.

The PMB_FSM block implements the state machine to support the I²C physical layer communication. The PMB_FSM machine is fully controlled by the CPU. The PMB_FSM is also in charge of accommodating the signal crossing clock domains due to the presence of the dedicated PMBus PHY clock (pmbus_kernel_clk). As far as the clock domains go, it is assumed that all required clocks, even with different frequencies, are synchronous with each other. A WDT is provided to monitor the SCL low status per SMBus specification; another WDT is provided to avoid clock stretch stalling; watchdogs are enabled only when a transaction is in progress. The PMB_FSM embeds an HW CRC logic for RX PEC and TX PEC calculations. The FSM is organized into two main

sections: the RX section and the TX section. The RX section must collect the incoming byte data (ADR, DATA), while the TX section has the duty to transmit the byte data provided.

15.10.1 PMBus clock scheme

The PMBus unit has two clock inputs (Figure 116): the APB interface clock (pclk) used to clock the registers inside the PMB_APB clock, and the PMBus kernel clock (pmbus_kernel_clk) used to move forward the PMB_FSM and to filter the PMBus interface input signal in the PMB_I2CF.

The implementation of the PMBus unit assumes all clocks are in phase with all other clocks in the PMBus module. The pmbus_kernel_clk is required to be more than eight times the maximum PMBus interface clock rate. The pclk is required to be equal to or greater than the pmbus_kernel_clk when the APB interface is in operation.

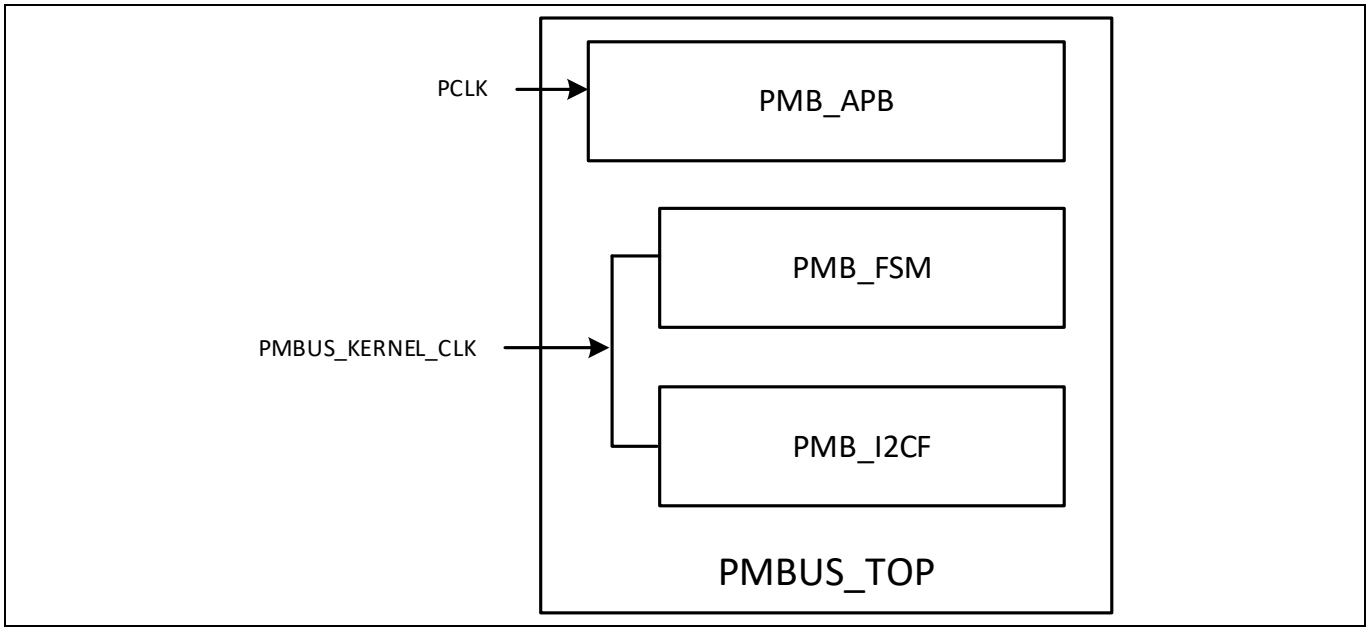

Figure 116 PMBus clock scheme

15.10.2 Interrupt generation

The PMBus interrupt is generated at every I^2C phase completion: it is generated at the detection of a start event (start_irq), when the first byte after start is received (rx_adr_w_irq or rx_adr_r_irq), when further bytes are received (rx_irq), when a byte is requested to be transmitted (tx_irq), when a stop event is detected (stop_irq) or when a watchdog event occurs (wdt_irq).

Interrupt is propagated to the CPU when the mask has been unmasked by writing “1” into the corresponding enable bit of the ISR register. Pending interrupts are cleared by writing “1” into the corresponding status bits in the ISR registers or a module reset or a generic reset is performed. The ISR register holds the value until interrupts are cleared.

FSM-generated interrupts are stored into first-in, first-out (FIFO) memory, having a depth of four locations: the PMB_IRQ interrupt line is asserted when there is at least one interrupt in the FIFO and the mask is released. To pop one element from the FIFO, an interrupt clear operation must be triggered.

15.10.3 PMBus operation

This section describes the procedure required to configure and manage the PMBus transactions with the preemptive mode enabled (CNFG_REG.EN_PREEMPTIVE_MODE = 1) and automatic transmission of the TXPEC.

15.10.3.1 Configuration

To enable the CPU to manage the PMBus transactions, the following actions must be performed (Figure 117):

- Enable the PMBus APB clocks interface (bif_per_pmbus_clk)
- Configure and enable the PMBus KERNEL clocks (pmbus_kernel_clk)
- Release the PMBus APB peripheral reset (bif_per_pmbus_rstn)
- Configure/initialize the PMBus registers:
  - Set the DEBOUNCE_LENGTH, the WDT_SCL_LOW_VAL, the WDT_SCL_STRETCH, the EN_AUTO_TRANSMIT_TX_PEC and the EN_PREEMPTIVE_MODE in the CNFG register
  - Clear the interrupts: Write “1” in the interrupt flags of the ISR register until FIFO_IRQ is empty (ISR_REG.I_CODE == NONE_IRQ and ISR_REG.I_PMB == 0)
  - Unmask the interrupt source: Write “1” on the interrupt source in the ISR register (ISR_REG.IE_PMB)
  - Enable the PMBus PHY to operate: set CNFG_REG.EN_PHY to 1. At this point the PMBus will start to process all the incoming PMBus transactions
- Configure the ARA_CW registers by setting the ARA_ADDR and enabling the ARA_CW
- Configure the ADDR_LUT by setting the SLAVE_ADDR and the type of transaction at which the interface will react, then enable the address screening for all wanted addresses
- Clear the DATA_LUT
- Enable PMBus PHY to operate: set CNFG_REG.EN_PHY to 1

At this point, the PMBus will start to process all the incoming PMBus transactions.

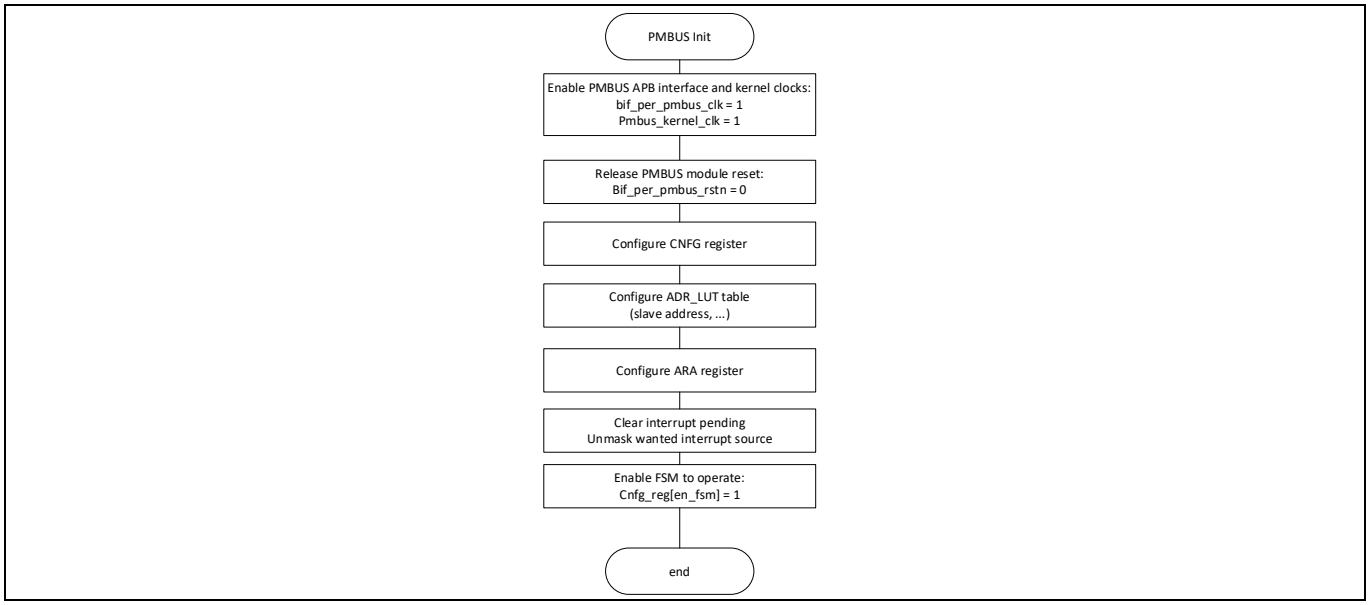

Figure 117 PMBus initialization

15.10.3.2 Write transaction

The following is an example of how to manage an incoming WRITE_BYTE transaction with and without PEC, where the host places data on the I^2C bus and the slave decodes the data. A flow chart to manage a generic PMBus write transaction is shown in Figure 118.

- Wait for PMB IRQ event and I_CODE == START_IRQ. This means there is an ongoing PMBus transaction
- Clear the interrupt event (write “1” into ISR.I_PMB)
- Select the ACK/NACK source to be used for automatic generation; that is, set CTRL_RX.ACK_SRC_SEL to 1 (ADDR_LUT)
- Wait for PMB IRQ event and I_CODE == RX_ADR_W_IRQ
- Clear the interrupt event (write “1” into ISR.I_PMB)
- Read the status of the ACK signal (STATUS_REG.ACK_STATUS)
- If ACK_STATUS == 1, then the incoming address belongs to the device, hence load DATA_LUT with a valid command mask table, read the data byte (this is the ADDRESS), select the proper ACK source for ACK/NACK automatic generation (CTRL_RX.ACK_SRC_SEL to 2), trigger the FSM to move forward (write “1” into FSM_TRIGGER bit, CTRL_RX.RX_TRIGGER = 1)
- If ACK_STATUS == 0, the FSM machine will enter the WAIT_FOR_STOP condition as soon as the trigger is provided (CTRL_RX.RX_TRIGGER = 1); drop transaction
- Wait for PMB IRQ event and I_CODE == RX_IRQ
- Clear the interrupt event (write “1” into ISR.I_PMB)
- Read the status of the ACK signal (STATUS_REG.ACK_STATUS)
- If ACK_STATUS == 1, the command is valid, then depending on the received command, load DATA_LUT with a valid data range mask, read the data byte (this is the CMD_BYTE), select the proper ACK source for comparison (CTRL_RX.ACK_SRC_SEL to 2), trigger the FSM to move forward (write “1” into FSM_TRIGGER bit, CTRL_RX.RX_TRIGGER = 1)
- If ACK_STATUS == 0, trigger (CTRL_RX.RX_TRIGGER = 1), and drop the transaction as the command is not valid (NACK was generated)
- Wait for PMB IRQ event and I_CODE == RX_IRQ
- Clear the interrupt event (write “1” into ISR.I_PMB)
- If ACK_STATUS == 1, the data is valid, read the data byte (this is the DATA BYTE), select the proper ACK source for ACK/NACK prediction in case of PEC reception (CTRL_RX.ACK_SRC_SEL to 3), trigger the FSM to move forward (CTRL_RX.RX_TRIGGER = 1)
- If ACK_STATUS == 0, data was not valid, trigger FSM (CTRL_RX.RX_TRIGGER = 1) and drop the transaction
- Wait for the PMB IRQ event and I_CODE == RX_IRQ or I_CODE == STOP_IRQ
- Clear the interrupt event (write “1” into ISR.I_PMB)
- If I_CODE == STOP_IRQ, execute the received command
- If I_CODE == RX_IRQ and ACK_STATUS == 1, PEC matched, transaction should be terminated, select the proper ACK source for ACK/NACK prediction (CTRL_RX.ACK_SRC_SEL to 0, == NACK), trigger the FSM to move forward (CTRL_RX.RX_TRIGGER = 1)
- If I_CODE == RX_IRQ and ACK_STATUS == 0, PEC did not match; trigger machine (CTRL_RX.RX_TRIGGER = 1), drop transaction
- Wait for PMB IRQ event and I_CODE == STOP_IRQ
- Clear the interrupt event (write “1” into ISR.I_PMB)
- Enable execution of the received transaction (write byte in this case)

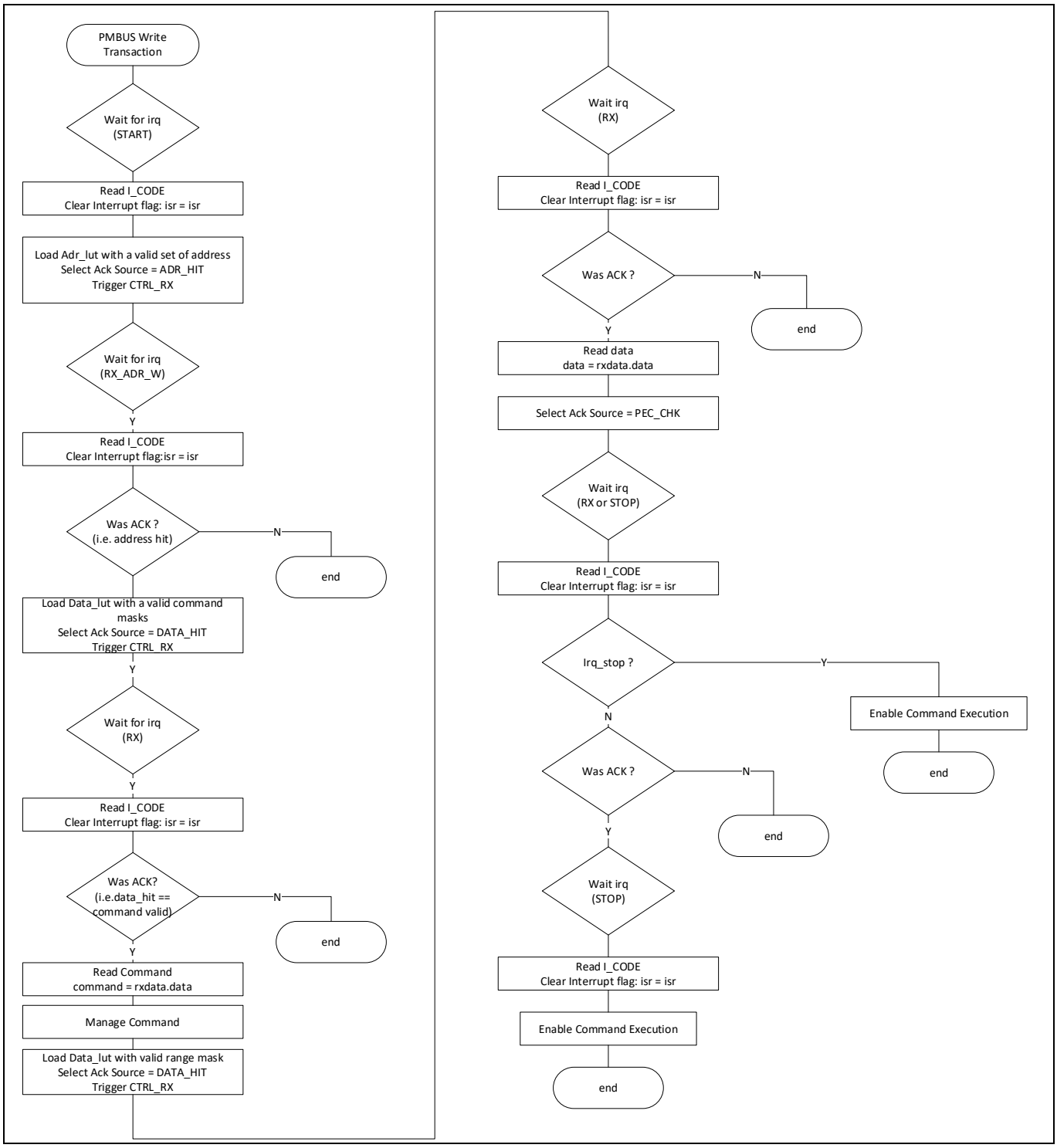

Figure 118 PMBus WRITE transaction

15.10.3.3 Read byte transaction

Here the process of managing an incoming READ_BYTE transaction (Figure 119) with and without PEC is described, where the slave has to place data on the I^2C bus following an initial sequence where the host sends the read request.

- Wait for PMB IRQ event and I_CODE == START_IRQ. This means there is an ongoing PMBus transaction
- Clear the interrupt event (write “1” into ISR.I_PMB)
- Enable at least one control word in the ADR_LUT by setting the EN_ADDR field and the target slave address
- Select the ACK/NACK automatic detection on from ADR_LUT; that is, set CTRL_RX.ACK_SRC_SEL to 1
- Wait for PMB IRQ event and I_CODE == RX_ADR_W_IRQ
- Clear the interrupt event (write “1” into ISR.I_PMB)
- Read the status of the ACK signal (STATUS_REG.ACK_STATUS)
- If ACK_STATUS == 1, then the incoming address belongs to the device, hence load DATA_LUT with a valid command mask table, read the data byte (this is the ADDRESS), select the proper ACK source for ACK/NACK prediction (CTRL_RX.ACK_SRC_SEL to 2), trigger the FSM to move forward (write 1 into FSM_TRIGGER bit, CTRL_RX.RX_TRIGGER = 1)
- If ACK_STATUS == 0, drop transaction
- Wait for PMB IRQ event and I_CODE == RX_IRQ
- Clear the interrupt event (write “1” into ISR.I_PMB)
- Read the status of the ACK signal (STATUS_REG.ACK_STATUS)
- If ACK_STATUS == 1, command is valid, then depending on the received command, load DATA_LUT with a valid data range mask, read the data byte (this is the CMD_BYTE), select the proper ACK source for comparison (CTRL_RX.ACK_SRC_SEL to 2), trigger the FSM to move forward (write “1” into FSM_TRIGGER bit, CTRL_RX.RX_TRIGGER = 1)
- If ACK_STATUS == 0, drop transaction as command is not valid (NACK received)
- Enable irq_rx_after_start
- Wait for irq_rx_after_start interrupt assertion
- Clear the interrupt event (write “1” into ISR.IFx)
- Read the slave address
- Check for address consistency
- If the address belongs to the device, enable reception of irq_rx interrupt event to detect the next incoming byte, set the ACK bit and trigger the FSM to move forward (write “3” into FSM_TRIGGER bit); otherwise enable the reception of the irq_rx_after_start interrupt event again, set NACK (i.e., ~ACK) and trigger FSM to move forward
- Wait for irq_rx interrupt assertion
- Clear the interrupt event (write “1” into ISR.IFx)
- Read the data byte (this is the CMD_BYTE)
- Check the validity of command
- If “valid” then set the expected number of bytes to be received with the commands (0 in this case), enable the reception of the irq_rx interrupt event, set ACK and trigger the FSM to move forward. If “not valid”: drop transaction, enable the reception of the irq_rx_after_start interrupt event again, set NACK (i.e., ~ACK) and trigger FSM to move forward
- Wait for irq_tx_after_start interrupt assertion (other events will trigger different command interpretations)
- Clear the interrupt event (write “1” into ISR.IFx)
- Prepare the TX_BUFFER with the expected data, set the number of bytes to be transmitted, then trigger the FSM to move forward
- Wait for irq_tx or irq_stop interrupt assertion
- If irq_tx, clear the interrupt event (write “1” into ISR.IFx)

- Read the calculated TX PEC data byte, load the TXDATA register with this byte, then trigger the FSM to move forward
- If irq_stop, clear the interrupts

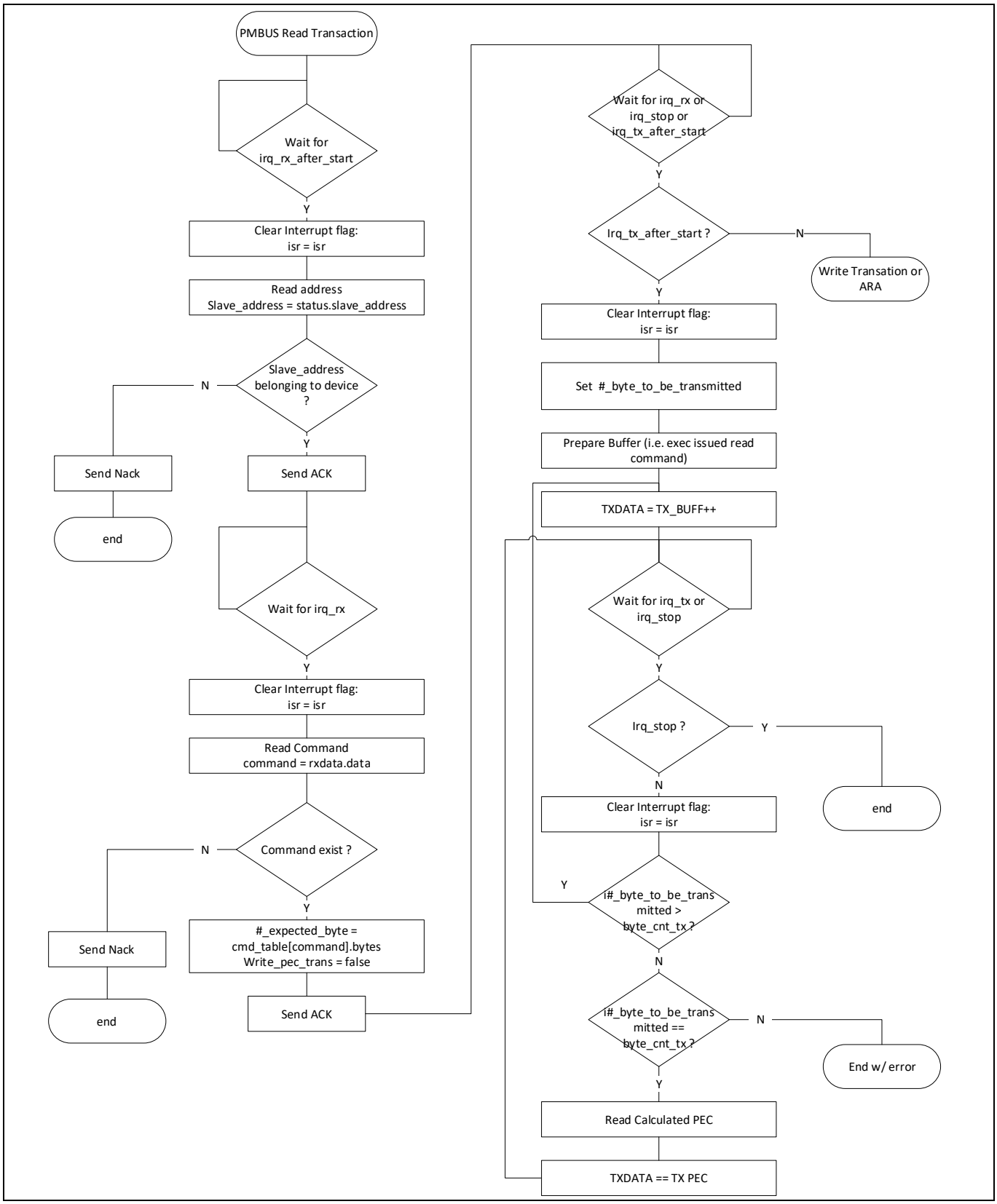

Figure 119 Read byte flowchart

15.10.3.4 PMBus ARA command

The ARA incoming transaction can be managed with the following sequence (Figure 120):

- Enable irq_rx_ara to detect ARA transaction
- Wait for irq_rx_ara interrupt assertion
- Clear the interrupts
- Check if at least one SMBALERT signal was generated by the device:
  - if true, set ACK, write TXDATA with device SLAVE_ADDRESS associated with the SMBALERT, clear the SMBALERT and trigger the FSM
  - if false, set NACK and trigger the FSM to move forward

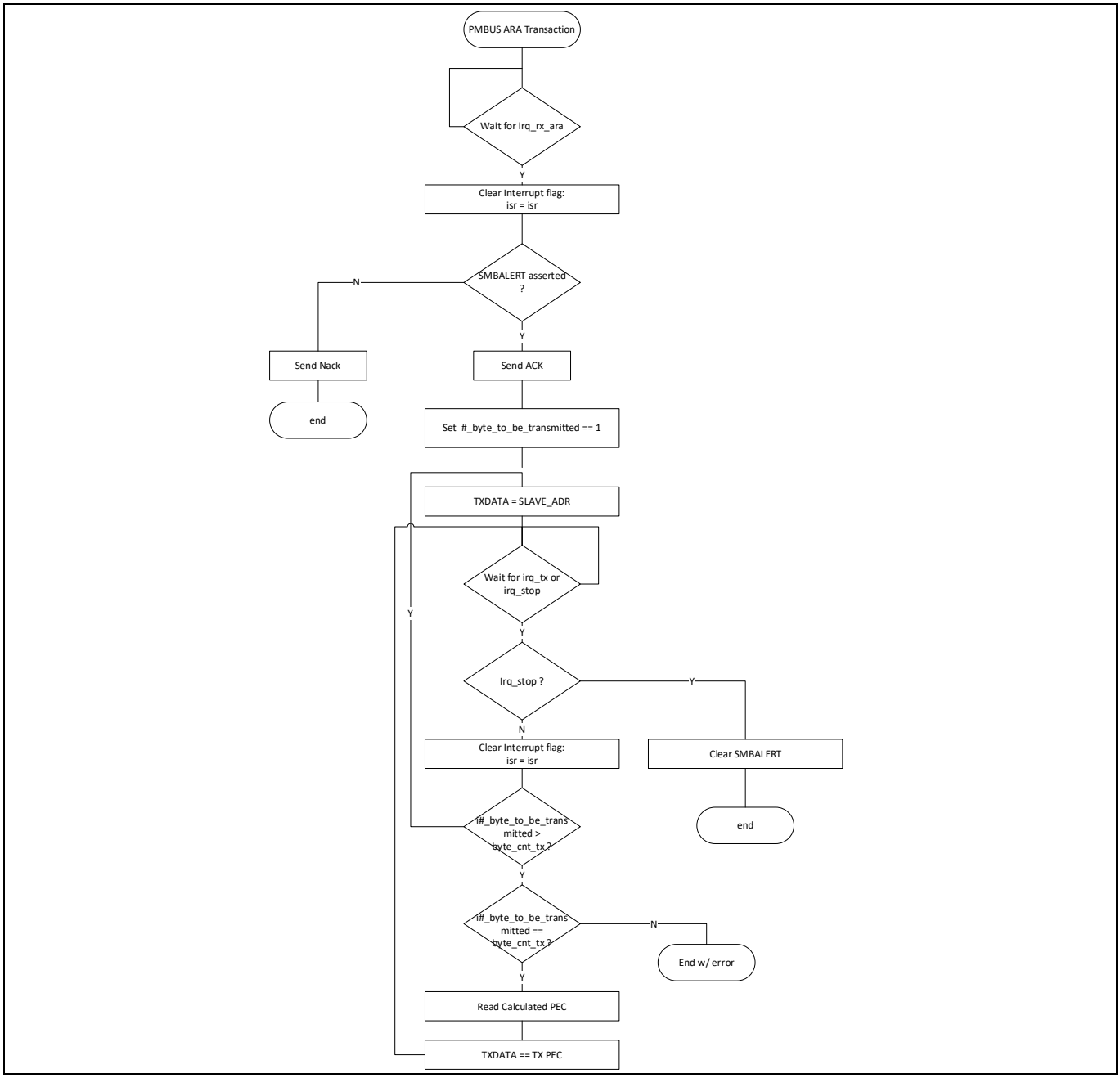

Figure 120 ARA transaction flow diagram

15.10.4 PMBus registers

The relevant PMBus-related registers and their descriptions are provided in Table 109.

Table 109 PMBus-related register descriptions

Register name	Field name	Access	Address	Bits	Description
STATUS	BUSY	R	7008_0000h	[0]	Reflects the status of the finite state machine (FSM). 0: FSM is in idle state 1: FSM is not in idle state
STATUS	DIRECTION	R	7008_0000h	[1]	Reflects the status of the transaction direction (this is correlated to the slave address if bit_cnt == 7). 0: Operation bit is set to write 1: Operation bit is set to read
STATUS	ACK_STATUS	R	7008_0000h	[2]	Reflects the status of ACK_NACK at the interrupt generation. 0: NACK 1: ACK
STATUS	TOO_FEW_BITS	R	7008_0000h	[3]	Reflects the flag status indicating a transaction termination not aligned to the byte. 0: All transactions have been aligned to bytes 1: Transaction not aligned to bytes
STATUS	BIT_CNT	R	7008_0000h	[7:4]	Current number of received or transmitted bits since last interrupt.
STATUS	BYTE_CNT_RX	R	7008_0000h	[15:8]	Current numbers of received bytes since start event.
STATUS	BYTE_CNT_TX	R	7008_0000h	[23:16]	Current numbers of transmitted bytes since start event.
STATUS	BUS_CONTENTION	R	7008_0000h	[24]	Reflects the bus contention status of the last TX transaction performed. 0: Bus contention did not occur 1: Bus contention occurred
STATUS	TOO_LONG_SCL_STRETCH	R	7008_0000h	[25]	SCL clock stretching timeout. 0: No timeout occurred 1: The SCL clock has been internally stretched more than the programmed timeout
STATUS	SCL_IN	R	7008_0000h	[26]	Status of the debounced SCL input signal.

Register name	Field name	Access	Address	Bits	Description
					0: SCL_IN is at low level 1: SCL_IN is at high level
STATUS	SDA_IN	R	7008_0000h	[27]	Status of the debounced SDA input signal. 0: SDA_IN is at low level 1: SDA_IN is at high level
STATUS	FISRT_TO_ASSERT_SMBALERT	R	7008_0000h	[28]	First to assert SMBALERT. 0: The SMBALERT was already asserted by another device 1: The SMBALERT has been asserted for the first time by this device
STATUS	TOO_LONG_SCL_HIGH	R	7008_0000h	[29]	This flag is indicating which kind of SCL timeout occurred after a watchdog timer interrupt event: 0: SCL was held low externally for too long 1: SCL was held high externally for too long
CNFG	EN_PHY	RW	7008_0004h	[0]	Enable/Disable PMBus PHY operation. When this bit is cleared, the clock of the I^2CF section is frozen, and the FSM is in the idle state. 0: PMBus PHY is not operating, FSM is forced into idle state. 1: PMBus PHY is operating.
CNFG	DS_FSM_FSM_CLK_G	RW	7008_0004h	[1]	Disable/Enable the FSM clk_fsm gating. 1: Clock is always running 0: Clock will freeze when no PMBus transaction is in progress
CNFG	DS_FSM_IRQ_CLK_G	RW	7008_0004h	[2]	Disable/Enable the FSM clk_irq gating. 1: Clock is always running 0: Clock will freeze if interrupt does not need to be managed
CNFG	DS_APB_IRQ_CLK_G	RW	7008_0004h	[3]	Disable/Enable the APB clk_irq gating. 1: Clock is always running 0: Clock will freeze if interrupt is not pending and does not need to be managed

Register name	Field name	Access	Address	Bits	Description
CNFG	PREEMPTIVE	RW	7008_0004h	[4]	Enable/Disable automatic handler of ACK/NACK. 0: CPU is in charge to generate ACK/NACK 1: HW takes care of generation of ACK/NACK. HW must be properly programmed
CNFG	EN_AUTO_TRANSMIT_TX_PEC	RW	7008_0004h	[5]	Enable/Disable automatic transfer of the TX PEC. 0: CPU is in charge of providing the TX PEC to be transmitted 1: HW takes care of transmission of the TX PEC as soon as the byte_cnt_tx is equal to the programmed byte_to_tx
CNFG	USE_START_IRQ_DELAY	RW	7008_0004h	[6]	Enable/Disable usage of delayed START_IRQ. Enable/Disable the FSM of PMBus to screen out the transactions not belonging to the device; the screen-out will take place only on first start after stop and not for restart 0: START_IRQ interrupt always occurs at the start phase, then the CPU is in charge of evaluating all of the PMBus transactions 1: START_IRQ interrupt is delayed at the end of reception of the PMBus address, before the direction bit, to allow PMBus transaction screening
CNFG	DEBOUNCE_LENGTH	RW	7008_0004h	[9:7]	Configure the number of consecutive bif_pmbus_kernel_clk cycles to accept a status change on scl_in and sda_in signals. 0: Signal must be stable for at least one clock signal 1: Signal must be stable for at least two clock signals 2: Signal must be stable for at least three clock signals 3: Signal must be stable for at least four clock signals 4: Signal must be stable for at least five clock signals

Register name	Field name	Access	Address	Bits	Description
					5: Signal must be stable for at least six clock signals 6: Signal must be stable for at least seven clock signals 7: Signal must be stable for at least eight clock signals
CNFG	WDT_SCL_STRETCH	RW	7008_0004h	[18:10]	Set the maximum number of clock cycles for which the SCL clock line can be kept low by XDPP1100. This number is internally multiplied by 2 * PCLK period (nominal 10 ns). XDPP1100 keeps the SCL low to stretch the PMBus transaction in order to have time to elaborate the request
CNFG	WDT_SCL_LOW_MAX	RW	7008_0004h	[31:19]	Set the SCL line watchdog timeout value. This number is multiplied by 256 * PMBUS_Kernel_clk period (nominal 40 ns). If the SCL clock line is continuously held low or high for longer than this time, a timeout event is generated. TOO_LONG_SCL_HIGH status bit signals if the line was kept low or high for too long.
ISR	I_FSM	RW	7008_0008h	[0]	PMBus IRQ status. 0: No interrupt 1: Interrupt triggered Write 1 to clear the interrupt.
ISR	IE_FSM	RW	7008_0008h	[1]	PMBus IRQ enable. 0: Interrupt disabled 1: Interrupt enabled
ISR	SEL_FLAG_SRC	RW	7008_0008h	[2]	Select the STATUS register source. 0: Current status (actual real-time status) 1: FIFO IRQ status When “1” is selected, status register is latched into a FIFO in the same way as interrupts, providing coherency between IRQ received and corresponded status.
ISR	I_CODE	R	7008_0008h	[6:3]	PMBus IRQ code. 0: No IRQ 1: Start IRQ 2: RX address write IRQ

Register name	Field name	Access	Address	Bits	Description
					3: RX IRQ 4: RX address read IRQ 5: TX IRQ 6: Stop IRQ 7: Watchdog timeout IRQ 8: Busy IRQ
ARA_CW	EN_ARA_CW	RW	7008_000Ch	[0]	Reserved
ARA_CW	DIRECTION	RW	7008_000Ch	[1]	Set the transaction direction. Configure the transaction direction to be considered in the ARA address matching. 0: Write transaction 1: Read transaction
ARA_CW	ARA_ADDR	RW	7008_000Ch	[8:2]	Set the address to identify the ARA transaction.
ARA_CW	SMBALERT	RW	7008_000Ch	[9]	Enable/Disable SMBALERT generation. 0: Release SMBALERT 1: Assert SMBALERT
ARA_CW	SLAVE_ADDR_GEN_SMBALERT	RW	7008_000Ch	[16:10]	Set the address generating the SMBALERT. This address will be automatically shifted out as soon as an ARA address is detected and the SMBALERT is asserted
CTRL_RX	RX_TRIGGER	W	7008_0010h	[0]	Trigger HW FSM to move on after a RX* interrupt has been received. Note: CPU has to trigger the FSM to move on within WDT_SCL_STRETCH clock cycles otherwise the transaction will be dropped, i.e., the FSM will move into the WAIT_FOR_START_OR_STOP state. On writing: 0: Do nothing. 1: (Return to 0), trigger HW FSM RX to move on
CTRL_RX	ACK_RECEPTION	RW	7008_0010h	[1]	Acknowledge the received byte. Set the status of the ACK/NACK bit to be transmitted after byte has been received when “PREEMPTIVE MODE” has not been selected. 0: Generate NACK 1: Generate ACK

Register name	Field name	Access	Address	Bits	Description
CTRL_RX	ACK_SRC_SEL	RW	7008_0010h	[4:2]	ACK/NACK source selection. Define which kind of comparison action is used to automatically generate the ACK/NACK event after a byte has been received. 0: Always NACK 1: Select ADR_LUT comparison table or ARA_CW. Basically (if ADDR_LUT(i) == RXDATA then ACK) or (if ARA_CW.ara_adr == RXDATA then ACK else NACK). 2: Select DATA_LUT comparison table, i.e., (if DATA_LUT(RXDATA) == 1 then ACK else NACK) 3: Select RXPEC comparison, i.e., (if RXPEC == RXDATA then ACK else NACK) 4: Always ACK 5: Always NACK 6: Always NACK 7: Always NACK
CTRL_RX	DATA_LUT_OP	RW	7008_0010h	[6:5]	DATA_LUT operation selection. By writing on this field it is possible to manage multiple DATA_LUTs with a single access. 0: No operation 1: Clear all the DATA_LUTs 2: Set all DATA_LUTs in the index interval [MIN_RANGE : MAX_RANGE] to 1 3: Initialize all the DATA_LUTs to 1
CTRL_RX	RX_REQ	R	7008_0010h	[7]	Acknowledge request. When high, PMBus FSM is requesting (to FW) to provide the confirmation to send the ACK on the PMBus line.
CTRL_RX	MIN_RANGE	RW	7008_0010h	[15:8]	Programmable LUT index for DATA_LUT_OP command (CTRL_RX and CTRL_TX)
CTRL_RX	MAX_RANGE	RW	7008_0010h	[23:16]	Programmable LUT index for DATA_LUT_OP command (CTRL_RX and CTRL_TX).
RXDATA	DATA	R	7008_0014h	[7:0]	Reports the last received data byte.
RXPEC	DATA	R	7008_0018h	[7:0]	The RXPEC register reflects the ongoing PEC CRC calculation for

Register name	Field name	Access	Address	Bits	Description
					the incoming bytes (address included). The RXPEC register is cleared at detection of a start event
CTRL_TX	TX_TRIGGER	W	7008_0020h	[0]	Trigger HW FSM to move on after a TX* interrupt has been received. Note: CPU has to trigger the FSM to move on within WDT_SCL_STRETCH clock cycles otherwise the transaction will be dropped (FSM will move into WAIT_FOR_START_OR_STOP state). On writing: 0: Do nothing. 1: (Return to 0), trigger HW FSM TX to move on
CTRL_TX	DROP_TRANSMISSION	RW	7008_0020h	[1]	Drop TX transmission. Force end of transmission in spite of received ACK in response of the transmitted byte or configured BYTE_TO_TX. 0: Don’t care 1: Drop byte transmission
CTRL_TX	BYTE_TO_TX	RW	7008_0020h	[10:2]	Number of bytes to be transmitted. Set the number of bytes to be automatically transmitted before output will be masked. Set BYTE_TO_TX to 0 force output to be masked to all 1. Maximum number of bytes to be transmitted: 256
CTRL_TX	BYTE_REMAINING	RW	7008_0020h	[13:11]	Number of remaining bytes to be transmitted before generating IRQ_TX. Set the level of the remaining number of bytes to be automatically transmitted before a new TX interrupt is propagated to CPU. Set BYTE_TO_TX to 0 force output to be masked to all 1. Maximum number of bytes depends on the byte size of the DATA_LUT
CTRL_TX	DATA_LUT_OP	RW	7008_0020h	[15:14]	DATA_LUT operation selection. By writing on this field it is possible to manage multiple DATA_LUTs with a single access. 0: No operation 1: Clear all the DATA_LUTs

Register name	Field name	Access	Address	Bits	Description
					2: Set all the DATA_LUTs in the index interval [MIN_RANGE : MAX_RANGE] to 1 3: Initialize all the DATA_LUTs to 1
CTRL_TX	TX_REQ	R	7008_0020h	[16]	TX status flag. When high, PMBus FSM is requesting (to FW) data in the transmit FIFO.
TXDATA	DATA	R	7008_0024h	[7:0]	The TXDATA register reports the last transmitted data byte
TXPEC	DATA	R	7008_0028h	[7:0]	The TXPEC register is 32-bit read-only register. It reflects the ongoing PEC CRC calculation for the transmitted bytes. The TXPEC is cleared at the start event
DMA_CTRL	EN_DMARX	RW	7008_002Ch	[0]	Enable/Disable DMARX request. The DMARX request is generated as soon as an irq_rx event is generated. This means the DMA can be used to move the received data (received address is not included) from PMBus to MEM or to move the data to be transmitted from MEM to PMBus. 0: DMARX request is masked 1: DMARX request is enabled
DMA_CTRL	EN_DMATX	RW	7008_002Ch	[1]	Enable/Disable DMATX request. The DMATX request, if enabled, is generated as soon as an irq_tx event is generated. 0: DMATX request is masked 1: DMATX request is enabled
ADDR_LUT0_ADDR_CW	EN_ADDR	RW	7008_0040h	[1:0]	Enable/Disable this ADDR_LUT0 configuration word. 0: ADDR configuration word is disabled 1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions
ADDR_LUT0_ADDR_CW	ADDR	RW	7008_0040h	[8:2]	Sets the slave address at which the PMBus interface will respond

Register name	Field name	Access	Address	Bits	Description
ADDR_LUT0_ADDR_CW	TYPE	RW	7008_0040h	[10:9]	Transaction type. Defines the type of transaction associated with the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved 3: Reserved
ADDR_LUT1_ADDR_CW	EN_ADDR	RW	7008_0044h	[1:0]	Enable/Disable this ADDR_LUT1 configuration word. 0: ADDR configuration word is disabled 1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions
ADDR_LUT1_ADDR_CW	ADDR	RW	7008_0044h	[8:2]	Sets the slave address at which the PMBus interface will respond
ADDR_LUT1_ADDR_CW	TYPE	RW	7008_0044h	[10:9]	Transaction type. Defines the type of transaction associated to the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved 3: Reserved
ADDR_LUT2_ADDR_CW	EN_ADDR	RW	7008_0048h	[1:0]	Enable/Disable this ADDR_LUT2 configuration word. 0: ADDR configuration word is disabled 1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions
ADDR_LUT2_ADDR_CW	ADDR	RW	7008_0048h	[8:2]	Sets the slave address at which the PMBus interface will respond
ADDR_LUT2_ADDR_CW	TYPE	RW	7008_0048h	[10:9]	Transaction type. Defines the type of transaction associated with the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved

Register name	Field name	Access	Address	Bits	Description
					3: Reserved
ADDR_LUT3_ADDR_CW	EN_ADDR	RW	7008_004Ch	[1:0]	Enable/Disable this ADDR_LUT3 configuration word. 0: ADDR configuration word is disabled 1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions
ADDR_LUT3_ADDR_CW	ADDR	RW	7008_004Ch	[8:2]	Sets the slave address at which the PMBus interface will respond
ADDR_LUT3_ADDR_CW	TYPE	RW	7008_004Ch	[10:9]	Transaction type. Defines the type of transaction associated with the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved 3: Reserved
ADDR_LUT4_ADDR_CW	EN_ADDR	RW	7008_0050h	[1:0]	Enable/Disable this ADDR_LUT4 configuration word. 0: ADDR configuration word is disabled 1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions
ADDR_LUT4_ADDR_CW	ADDR	RW	7008_0050h	[8:2]	Sets the slave address at which the PMBus interface will respond
ADDR_LUT4_ADDR_CW	TYPE	RW	7008_0050h	[10:9]	Transaction type. Defines the type of transaction associated with the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved 3: Reserved
ADDR_LUT5_ADDR_CW	EN_ADDR	RW	7008_0054h	[1:0]	Enable/Disable this ADDR_LUT5 configuration word 0: ADDR configuration word is disabled

Register name	Field name	Access	Address	Bits	Description
					1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions
ADDR_LUT5_ADDR_CW	ADDR	RW	7008_0054h	[8:2]	Sets the slave address at which the PMBus interface will respond
ADDR_LUT5_ADDR_CW	TYPE	RW	7008_0054h	[10:9]	Transaction type. Defines the type of transaction associated with the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved 3: Reserved
ADDR_LUT6_ADDR_CW	EN_ADDR	RW	7008_0058h	[1:0]	Enable/Disable this ADDR_LUT6 configuration word. 0: ADDR configuration word is disabled 1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions
ADDR_LUT6_ADDR_CW	ADDR	RW	7008_0058h	[8:2]	Sets the slave address at which the PMBus interface will respond
ADDR_LUT6_ADDR_CW	TYPE	RW	7008_0058h	[10:9]	Transaction type. Defines the type of transaction associated with the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved 3: Reserved
ADDR_LUT7_ADDR_CW	EN_ADDR	RW	7008_005Ch	[1:0]	Enable/Disable this ADDR_LUT7 configuration word. 0: ADDR configuration word is disabled 1: ADDR responds to write transactions only 2: ADDR responds to read transactions only 3: ADDR responds to both read and write transactions

Register name	Field name	Access	Address	Bits	Description
ADDR_LUT7_ADDR_CW	ADDR	RW	7008_005Ch	[8:2]	Sets the slave address at which the PMBus interface will respond
ADDR_LUT7_ADDR_CW	TYPE	RW	7008_005Ch	[10:9]	Transaction type. Defines the type of transaction associated with the defined slave address. 0: PMBus transaction 1: I^2C 2: Reserved 3: Reserved
DATA_LUT0_DATA_W	DATA	RW	7008_0080h	[31:0]	The 32-bit data word 0 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface. [31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0
DATA_LUT1_DATA_W	DATA	RW	7008_0084h	[31:0]	The 32-bit data word 1 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface. [31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0
DATA_LUT2_DATA_W	DATA	RW	7008_0088h	[31:0]	The 32-bit data word 2 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface. [31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0
DATA_LUT3_DATA_W	DATA	RW	7008_008Ch	[31:0]	The 32-bit data word 3 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface.

Register name	Field name	Access	Address	Bits	Description
					[31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0
DATA_LUT4_DATA_W	DATA	RW	7008_0090h	[31:0]	The 32-bit data word 4 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface. [31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0
DATA_LUT5_DATA_W	DATA	RW	7008_0094h	[31:0]	The 32-bit data word 5 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface. [31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0
DATA_LUT6_DATA_W	DATA	RW	7008_0098h	[31:0]	The 32-bit data word 6 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface. [31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0
DATA_LUT7_DATA_W	DATA	RW	7008_009Ch	[31:0]	The 32-bit data word 7 in 8x32-bit (or 8x4-byte) scratch table used in the prediction ACK/NACK approach or as buffer for the data to be transmitted over the PMBus interface. [31:24] = Byte 3 [23:16] = Byte 2 [15:8] = Byte 1 [7:0] = Byte 0

Register name	Field name	Access	Address	Bits	Description
DATA_LUT_BIT_SET	INDEX	W	7008_00C0h	[7:0]	The DATA_LUT_BIT_SET.INDEX sets one bit of the DATA_LUT 8x32-bit scratch table to “1”, where INDEX = 0 corresponds to DATA_LUT0_DATA_W.DATA[0] INDEX = 31 corresponds to DATA_LUT0_DATA_W.DATA[31] … INDEX = 224 corresponds to DATA_LUT7_DATA_W.DATA[0] INDEX = 255 corresponds to DATA_LUT7_DATA_W.DATA[31]
DATA_LUT_BIT_CLR	INDEX	W	7008_00C4h	[7:0]	The DATA_LUT_BIT_CLR.INDEX sets one bit of the DATA_LUT 8x32-bit scratch table to “0”, where INDEX = 0 corresponds to DATA_LUT0_DATA_W.DATA[0] INDEX = 31 corresponds to DATA_LUT0_DATA_W.DATA[31] … INDEX = 224 corresponds to DATA_LUT7_DATA_W.DATA[0] INDEX = 255 corresponds to DATA_LUT7_DATA_W.DATA[31]
STATUS_WORD[0]	STATUS_WORD	R	7008_0100h	[15:0]	Loop 0 STATUS_WORD command data. This is a read-only register with the data populated based on the other STATUS commands
STATUS_WORD[1]	STATUS_WORD	R	7008_0104h	[15:0]	Loop 1 STATUS_WORD command data. This is a read-only register with the data populated based on the other STATUS commands
STATUS_VOUT[0]	STATUS_VOUT	RW	7008_0108h	[7:0]	Loop 0 STATUS_VOUT command data for HW-based SMBALERT generation
STATUS_VOUT[1]	STATUS_VOUT	RW	7008_010Ch	[7:0]	Loop 1 STATUS_VOUT command data for HW-based SMBALERT generation
STATUS_IOUT[0]	STATUS_IOUT	RW	7008_0110h	[7:0]	Loop 0 STATUS_IOUT command data for HW-based SMBALERT generation
STATUS_IOUT[1]	STATUS_IOUT	RW	7008_0114h	[7:0]	Loop 1 STATUS_IOUT command data for HW-based SMBALERT generation

Register name	Field name	Access	Address	Bits	Description
STATUS_INPUT[0]	STATUS_INPUT	RW	7008_0118h	[7:0]	Loop 0 STATUS_INPUT command data for HW-based SMBALERT generation
STATUS_INPUT[1]	STATUS_INPUT	RW	7008_011Ch	[7:0]	Loop 1 STATUS_INPUT command data for HW-based SMBALERT generation
STATUS_TEMP[0]	STATUS_TEMP	RW	7008_0120h	[7:0]	Loop 0 STATUS_TEMP command data for HW-based SMBALERT generation
STATUS_TEMP[1]	STATUS_TEMP	RW	7008_0124h	[7:0]	Loop 1 STATUS_TEMP command data for HW-based SMBALERT generation
STATUS_CML[0]	STATUS_CML	RW	7008_0128h	[7:0]	Loop 0 STATUS_CML command data for HW-based SMBALERT generation
STATUS_CML[1]	STATUS_CML	RW	7008_012Ch	[7:0]	Loop 1 STATUS_CML command data for HW-based SMBALERT generation
STATUS_OTHER[0]	STATUS_OTHER	RW	7008_0130h	[7:0]	Loop 0 STATUS_OTHER command data for HW-based SMBALERT generation.
STATUS_OTHER[1]	STATUS_OTHER	RW	7008_0134h	[7:0]	Loop 1 STATUS_OTHER command data for HW-based SMBALERT generation
STATUS_MFR[0]	STATUS_MFR	RW	7008_0138h	[7:0]	Loop 0 STATUS_MFR command data for HW-based SMBALERT generation
STATUS_MFR[1]	STATUS_MFR	RW	7008_013Ch	[7:0]	Loop 1 STATUS_MFR command data for HW-based SMBAlert generation
STATUS_FAN12[0]	STATUS_FAN12	RW	7008_0140h	[7:0]	Loop 0 STATUS_FAN12 command data for HW-based SMBALERT generation.
STATUS_FAN12[1]	STATUS_FAN12	RW	7008_0144h	[7:0]	Loop 1 STATUS_FAN12 command data for HW-based SMBALERT generation
STATUS_PWRGOOD[0]	STATUS_PWRGOOD	RW	7008_0148h	[0]	Loop 0 STATUS_PWRGOOD command data for HW-based SMBALERT generation
STATUS_PWRGOOD[1]	STATUS_PWRGOOD	RW	7008_014Ch	[0]	Loop 1 STATUS_PWRGOOD command data for HW-based SMBALERT generation
STATUS_OFF[0]	STATUS_OFF	RW	7008_0150h	[0]	Loop 0 STATUS_OFF command data for HW-based SMBALERT generation

Register name	Field name	Access	Address	Bits	Description
STATUS_OFF[1]	STATUS_OFF	RW	7008_0154h	[0]	Loop 1 STATUS_OFF command data for HW-based SMBALERT generation
STATUS_BUSY[0]	STATUS_BUSY	RW	7008_0158h	[0]	Loop 0 STATUS_BUSY command data for HW-based SMBALERT generation
STATUS_BUSY[1]	STATUS_BUSY	RW	7008_015Ch	[0]	Loop 1 STATUS_BUSY command data for HW-based SMBALERT generation
STATUS_UNKNOWN[0]	STATUS_UNKNOWN	RW	7008_0160h	[0]	Loop 0 STATUS_UNKNOWN command data for HW-based SMBALERT generation
STATUS_UNKNOWN[1]	STATUS_UNKNOWN	RW	7008_0164h	[0]	Loop 1 STATUS_UNKNOWN command data for HW-based SMBALERT generation
STATUS_CLEAR_ALL[0]	STATUS_CLEAR_ALL	W	7008_0168h	[0]	Clears all loop 0 status registers, 0x0100 through 0x0160, on write
STATUS_CLEAR_ALL[1]	STATUS_CLEAR_ALL	W	7008_016Ch	[0]	Clears all loop 1 status registers, 0x0104 through 0x0164, on write
STATUS_MASK_LP0[0]	STATUS_VOUT_MASK	RW	7008_0170h	[7:0]	Loop 0 STATUS_VOUT mask to enable/disable bits for SMBALERT generation. For each bit [x], 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP0[1]	STATUS_IOUT_MASK	RW	7008_0174h	[7:0]	Loop 0 STATUS_IOUT mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP0[2]	STATUS_INPUT_MASK	RW	7008_0178h	[7:0]	Loop 0 STATUS_INPUT mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP0[3]	STATUS_TEMP_MASK	RW	7008_017Ch	[7:0]	Loop 0 STATUS_TEMP mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation

Register name	Field name	Access	Address	Bits	Description
					1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP0[4]	STATUS_CML_MASK	RW	7008_0180h	[7:0]	Loop 0 STATUS_CML mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP0[5]	STATUS_OTHER_MASK	RW	7008_0184h	[7:0]	Loop 0 STATUS_OTHER mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP0[6]	STATUS_MFR_MASK	RW	7008_0188h	[7:0]	Loop 0 STATUS_MFR mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP0[7]	STATUS_FAN12_MASK	RW	7008_018Ch	[7:0]	Loop 0 STATUS_FAN12 mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP1[0]	STATUS_VOUT_MASK	RW	7008_0190h	[7:0]	Loop 1 STATUS_VOUT mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP1[1]	STATUS_IOUT_MASK	RW	7008_0194h	[7:0]	Loop 1 STATUS_IOUT mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation

Register name	Field name	Access	Address	Bits	Description
STATUS_MASK_LP1[2]	STATUS_INPUT_MASK	RW	7008_0198h	[7:0]	Loop 1 STATUS_INPUT mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP1[3]	STATUS_TEMP_MASK	RW	7008_019Ch	[7:0]	Loop 1 STATUS_TEMP mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP1[4]	STATUS_CML_MASK	RW	7008_01A0h	[7:0]	Loop 1 STATUS_CML mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP1[5]	STATUS_OTHER_MASK	RW	7008_01A4h	[7:0]	Loop 1 STATUS_OTHER mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP1[6]	STATUS_MFR_MASK	RW	7008_01A8h	[7:0]	Loop 1 STATUS_MFR mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation
STATUS_MASK_LP1[7]	STATUS_FAN12_MASK	RW	7008_01ACh	[7:0]	Loop 1 STATUS_FAN12 mask to enable/disable bits for SMBALERT generation. For each bit [x]: 0: Bit [x] enabled for SMBALERT generation 1: Bit [x] enabled for SMBALERT generation

15.11 I²C module

XDPP1100 includes an Inventra I²C (MI2CV) module.

The Inventra MI2CV provides an interface between the microprocessor and an I²C bus that conforms to the Philips I²C Bus Protocol (April 1995 Update). It can operate in either master or slave mode. It performs arbitration in master mode to allow it to operate in multi-master systems. In slave mode, it can interrupt the processor when it recognizes its own 7-bit or 10-bit address or the general call address.

The Cortex®-M0 accesses I²C registers through the APB interface at base address 700B_0000h.

The I²C block diagram is shown in Figure 121.

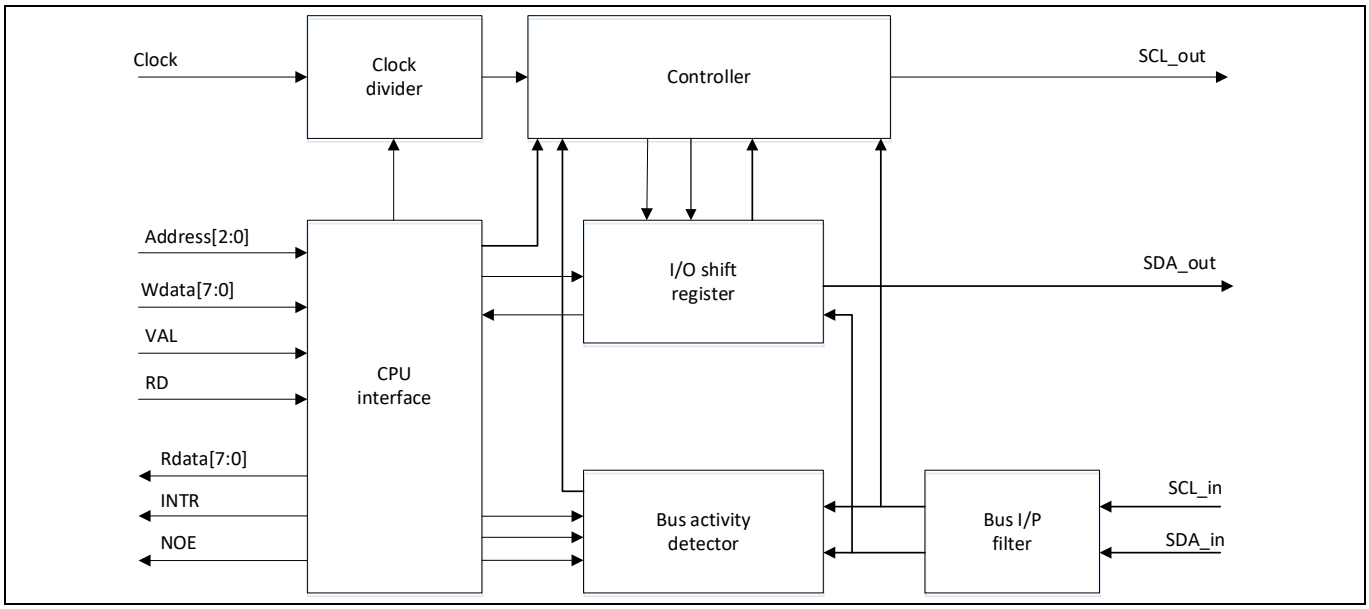

Figure 121 I²C block diagram

15.11.1 I²C operating modes

The I²C can operate in four modes: master transmit, master receive, slave transmit, and slave receive. The I²C will automatically enter slave transmit mode if it receives its own slave address and a read bit. It will similarly enter slave receive mode if it receives either its own slave address and a write bit, or the general call address.

15.11.1.1 Status information

The state of the interface at any time is indicated by the status code recorded in the STAT register.

There are 28 status codes corresponding to the different possible states of the I²C, plus a further code that indicates when no relevant status information is available.

The states reported cover all conditions from successful transmission to bus errors and loss of arbitration.

The appropriate microprocessor responses to the reported interface condition and the I²C actions these invoke are detailed in the I²C Product Specification.

5-bit status codes (STAT register) are described in Table 110.

Table 110 Status register codes

Code	Status
00h	Bus error
08h	START condition transmitted
10h	Repeated START condition transmitted
18h	Address + Write bit transmitted, ACK received
20h	Address + Write bit transmitted, ACK not received
28h	Data byte transmitted in master mode, ACK received
30h	Data byte transmitted in master mode, ACK not received
38h	Arbitration lost in address or data byte
40h	Address + Read bit transmitted, ACK received
48h	Address + Read bit transmitted, ACK not received
50h	Data byte received in master mode, ACK received
58h	Data byte received in master mode, ACK not received
60h	Slave address + Write bit received, ACK transmitted
68h	Arbitration lost in address as master, slave address + Write bit received, ACK transmitted
70h	General Call address received, ACK transmitted
78h	Arbitration lost in address as master, General Call address received, ACK transmitter
80h	Data byte received after slave address received, ACK transmitted
88h	Data byte received after slave address received, not ACK transmitted
90h	Data byte received after General Call received, ACK transmitted
98h	Data byte received after General Call received, not ACK transmitted
A0h	STOP or repeated START condition received in slave mode
A8h	Slave address + Read bit received, ACK transmitted
B0h	Arbitration lost in address as master, slave address + Read bit received, ACK transmitted
B8h	Data byte transmitted in slave mode, ACK received
C0h	Data byte transmitted in slave mode, ACK not received
C8h	Last byte transmitted in slave mode, ACK received
D0h	Second Address byte + Write bit transmitted, ACK received
D8h	Second Address byte + Write bit transmitted, ACK not received
E0h	Unused
E8h	Unused
F0h	Unused
F8h	No relevant status information, IFLG = 0

15.11.1.2 Master transmit

In master transmit mode, the I^2C will transmit a number of bytes to a slave receiver.

The master transmit mode is entered by setting the STA in the CNTR register bit to “1”. The I^2C will then test the I^2C bus and will transmit a START condition when the bus is free. When a START condition has been transmitted, the IFLG bit will be set and the status code in the STAT register will be 08h. Before this interrupt is serviced, the DATA register must be loaded with either a 7-bit slave address or the first part of a 10-bit slave address, with the LSB cleared to “0” (i.e., with an added write bit) to specify transmit mode. The IFLG bit should now be cleared to “0” to prompt the transfer to continue.

After the 7-bit slave address (or the first part of a 10-bit address) and the write bit have been transmitted, IFLG will be set again. The possible status register combinations are shown in Table 111.

Table 111 Master transmit status after transmitting the 7-bit address

Code	MI2CV state	Microprocessor response	Next MI2CV action
18h	Addr + Write bit transmitted, ACK received	**7-bit address:** Write byte to DATA, clear IFLG Or set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG **10-bit address:** Write extended address byte to DATA, clearIFLG	 Transmit data byte, receive ACK Transmit repeated START Transmit STOP Transmit STOP then START Transmit extended address byte
20h	Address + Write bit transmitted, ACK not received	Same as for code 18h	Same as for code 18h
38h	Arbitration lost	Clear IFLG Or set STA, clear IFLG	Return to idle Transmit START when bus free
68h	Arbitration lost, SLA + Write bit received, ACK transmitted	Clear IFLG, AAK=0 Or clean IFLG, AAK=1	Receive data byte, transmit not ACK Received data byte, transmit ACK
78h	Arbitration lost, general call addr received, ACK transmitted	Same as for code 68h	Same as for code 68h
B0h	Arbitration lost, SLA + Read bit received, ACK transmitted	Write byte to DATA, clear IFLG, AAK=0 Or Write byte to DATA, clear IFLG, AAK=1	Transmit last byte, receive ACK Transmit data byte, receive ACK

If 10-bit addressing is being used, then after the first part of a 10-bit address and the write bit have been successfully transmitted, the status code will be 18h or 20h.

After this interrupt has been serviced and the second part of the address transmitted, the STAT register will contain one code, as shown in Table 112.

Table 112 Master transmit status after transmitting a 10-bit address

Code	MI2CV state	Microprocessor response	Next MI2CV action
38h	Arbitration lost	Clear IFLG Or set STA, clear IFLG	Return to idle Transmit START when bus free
68h	Arbitration lost, SLAX + Write bit received, ACK transmitted	Clear IFLG, AAK=0 Or clear IFLG, AAK=1	Receive data byte, transmit not ACK Receive data byte, transmit ACK
B0h	Arbitration lost, SLAX + Read bit received, ACK transmitted	Write byte to DATA, clear IFLG, AAK=0 Or write byte to DATA, clear IFLG, AAK=1	Transmit last byte, receive ACK Transmit data byte, receive ACK
D0h	Second Address byte + Write bit transmitted, ACK received	Write byte to DATA, clear IFLG Or set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG	Transmit data byte, receive ACK Transmit repeated START Transmit STOP Transmit STOP then START
D8h	Second Address byte + Write bit transmitted, ACK not received	Same as for code D0h	Same as for code D0h

If a repeated START condition has been transmitted, the status code will be 10h instead of 08h.

After each data byte has been transmitted, IFLG will be set and one of three status codes will be in the STAT register (Table 113).

Table 113 Master transmit status after a repeated START

Code	MI2CV state	Microprocessor response	Next MI2CV action
28h	Data byte transmitted, ACK received	Write byte to DATA, clear IFLG	Transmit data byte, receive ACK
		Or set STA, clear IFLG	Transmit repeated START
		Or set STP, clear IFLG	Transmit STOP
		Or set STA and STP, clear IFLG	Transmit START then STOP
30h	Data byte transmitted, ACK not received	Same as for code 28h	Same as for code 28h
38h	Arbitration lost	Clear IFLG	Return to idle
		Or set STA, clear IFLG	Transmit START when bus free

When all bytes have been transmitted, the STP bit should be set by writing a “1” to this bit in the CNTR register. The I^2C will then transmit a STOP condition, clear the STP bit and return to idle state (status code F8h).

15.11.1.3 Master receive

In master receive mode, the I^2C will receive a number of bytes from a slave transmitter.

After the START condition has been transmitted, the IFLG bit will be set and status code 08h will be in the STAT register. The DATA register should now be loaded with the slave address (or the first part of a 10-bit slave address), with the LSB set to “1” to signify a read. The IFLG bit should now be cleared to “0” to prompt the transfer to continue.

When the 7-bit slave address (or the first part of a 10-bit address) and the read bit have been transmitted, the IFLG bit will be set again. A number of status codes are possible in the STAT register (Table 114).

Table 114 Master receive status after address and read bit transmission

Code	MI2CV state	Microprocessor response	Next MI2CV action
38h	Arbitration lost	Clear IFLG	Return to idle
		Or set STA, clear IFLG	Transmit START when bus free
40h	Address + Read bit transmitted, ACK received	Clear IFLG, AAK=0	Receive data byte, transmit not ACK
		Or clear IFLG, AAK=1	Receive data byte, transmit ACK
48h	Address + Read bit transmitted, ACK not received	Set STA, clear IFLG	Transmit repeated START
		Or set STP, clear IFLG	Transmit STOP
		Or set STA and STP, clear IFLG	Transmit STOP then START

Code	MI2CV state	Microprocessor response	Next MI2CV action
68h	Arbitration lost, SLA + Write bit received, ACK transmitted	Clear IFLG, AAK=0 Or clean IFLG, AAK=1	Receive data byte, transmit not ACK Received data byte, transmit ACK
78h	Arbitration lost, general call addr received, ACK transmitted	Same as for code 68h	Same as for code 68h
B0h	Arbitration lost, SLA + Read bit received, ACK transmitted	Write byte to DATA, clear IFLG, AAK=0 Or Write byte to DATA, clear IFLG, AAK=1	Transmit last byte, receive ACK Transmit data byte, receive ACK

If 10-bit addressing is being used, the slave is first addressed using the full 10-bit address plus the write bit. The master then issues a restart followed by the first part of the 10-bit address again, plus the read bit – after which the status code will be 40h or 48h. It is the responsibility of the slave to remember that it was selected prior to the restart.

If a repeated START condition has been transmitted, the status code will be 10h instead of 08h.

After each data byte has been received, IFLG will be set and one of three status codes will be in the STAT register (Table 115).

Table 115 Master receive status after a byte has been received

Code	MI2CV state	Microprocessor response	Next MI2CV action
38h	Arbitration lost in not ACK bit	Clear IFLG Or set STA, clear IFLG	Return to idle Transmit START when bus free
50h	Data byte received, ACK transmitted	Read DATA, clear IFLG, AAK=0 Or read DATA, clear IFLG, AAK=1	Receive data byte, transmit not ACK Receive data byte, transmit ACK
58h	Data byte received, not ACK transmitted	Read DATA, set STA, clear IFLG Or read DATA, set STP, clear IFLG Or read DATA, set STA and STP, clear IFLG	Transmit repeated START Transmit STOP Transmit STOP then START

When all bytes have been received, a not ACK should be transmitted, then the STP bit should be set by writing a “1” to this bit in the CNTR register. The I^2C will transmit a STOP condition, clear the STP bit and return to idle state (status code F8h).

15.11.1.4 Slave transmit

In slave transmit mode, a number of bytes are transmitted to a master receiver. For the I²C to respond, the AAK bit in the CNTR register needs to be set.

The I²C will enter slave transmit mode when it receives its own slave address and a read bit after a START condition. The I²C will then transmit an acknowledge bit and set the IFLG bit in the CNTR register. The STAT register will contain the status code A8h.

Where the I²C has an extended slave address (signified by 11110b in ADDR[7:3]), it will first be selected, then there will be a restart followed by another address byte. If this address byte matches the value stored in ADDR, the I²C will transmit an acknowledge after this address byte is received. An interrupt will be generated, IFLG will be set and the status will be A8h. No second address byte will be sent by the master: it is up to the slave to remember that it was selected prior to the restart.

Slave transmit mode can also be entered directly from a master mode if arbitration is lost in master mode during the transmission of an address and the slave address and read bit are received. The status code in the STAT register will then be B0h.

The data byte to be transmitted should then be loaded into the DATA register and IFLG cleared. When the I²C has transmitted the byte and received an acknowledge, IFLG will be set and the STAT register will contain B8h. Once the last byte to be transmitted has been loaded into the DATA register, the AAK bit should be cleared when IFLG is cleared. After the last byte has been transmitted, IFLG will be set and the STAT register will contain C8h. The I²C will then return to idle state (status code F8h). The AAK bit must be set to “1” before slave mode can be entered again.

If no acknowledge is received after transmitting a byte, IFLG will be set and the STAT register will contain C0h. The MI2CV will then return to idle state.

If the STOP condition is detected after an acknowledge bit, the MI2CV will return to idle state.

15.11.1.5 Slave receive

In slave receive mode, a number of data bytes are received from a master transmitter.

The I²C will enter slave receive mode when it receives its own slave address and a write bit (LSB = 0) after a START condition. The I²C will then transmit an acknowledge bit and set the IFLG bit in the CNTR register: the STAT register will then contain status code 60h. The I²C will also enter slave receive mode when it receives the general call address 00h (if the GCE bit in the ADDR register is set). The status code will then be 70h.

Where the I²C has an extended slave address (signified by 11110b in ADDR[7:3]), it will transmit an acknowledge after the first address byte is received but no interrupt will be generated, IFLG will not be set, and the status will not change. Only after the second address byte has been received will the I²C generate an interrupt, set the IFLG bit and the status code as described above.

Slave receive mode can also be entered directly from a master mode if arbitration is lost in master mode during the transmission of an address and the slave address and write bit (or the general call address if bit GCE in the ADDR register is set to “1”) are received. The status code in the STAT register will then be 68h if the slave address was received or 78h if the general call address was received. The IFLG bit must be cleared to “0” to allow the data transfer to continue.

If the AAK bit in the CNTR register is set to “1”, then after each byte is received, an acknowledge bit (low level on SDA) is transmitted and the IFLG bit is set: the STAT register will then contain status code 80h (or 90h if slave receive mode was entered with the general call address). The received data byte can be read from the DATA register and the IFLG bit must be cleared to allow the transfer to continue. When the STOP condition or a repeated START condition is detected after the acknowledge bit, then the IFLG bit is set, and the STAT register will contain status code A0h.

If the AAK bit is cleared to “0” during a transfer, the MI2CV will transmit a not acknowledge bit (high level on SDA) after the next byte is received and set the IFLG bit. The STAT register will contain status code 88h (or 98h if slave receive mode was entered with the general call address). When the IFLG bit has been cleared to “0”, the MI2CV will return to idle state (status code F8h).

15.11.2 I²C registers

The relevant I²C-related registers and their descriptions are provided in Table 116.

Table 116 I²C register descriptions

Register name	Field name	Access	Address	Bits	Description
ADDR	GCE	RW	700B_0000h	[0]	General call address enable. If set to “1” the I²C peripheral will also respond to general call address 00h
ADDR	SLAX	RW	700B_0000h	[2:1]	**For 7-bit addressing:** Slave address bits [1:0] (SLA1..SLA0). When the I²C peripheral receives address SLA6..SLA0 after a START condition, it will generate an interrupt and enter slave mode. **For 10-bit addressing:** Extended slave address bits [9:8] (SLAX9..SLAX8). When ADDR.SLA[4:0]=11110b, the I²C peripheral recognizes this as the first part of a 10-bit address and if the next two bits match ADDR.SLAX[1:0] it sends an ACK. If the next byte of the address matches XADDR.SLAX[7:0] the I²C peripheral generates an interrupt and goes into slave mode.
ADDR	SLA	RW	700B_0000h	[7:3]	**For 7-bit addressing:** Slave address bits [6:2] (SLA6..SLA2). When the I²C peripheral receives address SLA6..SLA0 after a START condition, it will generate an interrupt and enter slave mode. **For 10-bit addressing:** When ADDR.SLA[4:0]=11110b, the I²C peripheral recognizes this as the first

Register name	Field name	Access	Address	Bits	Description
					part of a 10-bit address and if the next two bits match ADDR.SLAX[1:0] it sends an ACK. If the next byte of the address matches XADDR.SLAX[7:0] the I²C peripheral generates an interrupt and goes into slave mode.
DATA	DATA	RW	700B_0004h	[7:0]	This register contains the data byte/slave address to be transmitted or the data byte that has just been received. In transmit mode, the byte is sent to MSB first; in receive mode, the first bit received will be placed in the MSB of the register. After each byte is transmitted, the DATA register will contain the byte that was actually present on the bus, so in the event of lost arbitration, it will contain the received byte.
CNTR	AAK	RW	700B_0008h	[2]	Assert acknowledge. 0: A not acknowledge (high level on SDA) will be sent when a data byte is received in master or slave mode. If AAK is cleared to “0” in slave transmitter mode, the byte in the DATA register is assumed to be the “last byte”. After this byte has been transmitted, the peripheral will enter state C8h then return to the idle state (status code F8h) when IFLG is cleared. 1: An acknowledge (low level on SDA) will be sent during the acknowledge clock pulse on the I²C bus if: Either the whole of a matching 7-bit slave address or the first or the second byte of a matching 10-bit slave address has been received. The general call address has been received and the GCE bit in the ADDR register is set to “1”. A data byte has been received in master or slave mode.
CNTR	IFLG	W	700B_0008h	[3]	Interrupt flag. IFLG is automatically set to “1” when any of 28 (out of the possible 29) I²C peripheral states is entered (see STAT.CODE). The only

Register name	Field name	Access	Address	Bits	Description
					state that does not set IFLG is state F8h. If the IEN bit is set, the interrupt line goes high when IFLG is set to “1”. If the peripheral is operating in slave mode, data transfer is suspended when IFLG is set and the low period of the I^2C bus clock line (SCL) is stretched until “0” is written to IFLG. The I^2C clock line is then released and the interrupt line goes low.
CNTR	STP	W	700B_0008h	[4]	Master mode stop. 0: No effect 1: If set to “1” in master mode, a STOP condition is transmitted on the I^2C bus. If STP is set to “1” in slave mode, the peripheral will behave as if a STOP condition has been received but no STOP condition will be transmitted on the bus. If both STA and STP bits are set, the peripheral will first transmit the STOP condition (if in master mode) then transmit the START condition. The STP bit is cleared automatically.
CNTR	STA	W	700B_0008h	[5]	Master mode start. 0: No effect 1: The I^2C peripheral enters master mode and will transmit a START condition on the bus when the bus is free. If the STA bit is set to “1” when the peripheral is already in master mode and one or more bytes have been transmitted, then a repeated START condition will be sent. If the STA bit is set to “1” when the peripheral is being accessed in slave mode, the slave mode data transfer will complete and master mode will be entered when the bus is released. The STA bit is automatically cleared after a START condition has been sent.
CNTR	ENAB	RW	700B_0008h	[6]	Bus enable. 0: I^2C inputs are ignored and the I^2C peripheral will not respond to any address on the bus

Register name	Field name	Access	Address	Bits	Description
					1: The I^2C peripheral will respond to calls on its slave address and to the general call address if ADDR.GCE=1
CNTR	IEN	RW	700B_0008h	[7]	I^2C peripheral interrupt line (INTR) enable 0: INTR will always remain low 1: INTR will go high when the IFLG bit is set
STAT_CCR	N	W	700B_000Ch	[2:0]	Along with STAT_CCR.M, controls the frequency at which the I^2C bus is sampled and the frequency of the clock line (SCL) when in master mode. $F_{samp} = F_{clock} / 2^N$ $F_{SCL} = F_{clock} / (2^N * (M+1) * 10)$ where F_{clock} is the frequency of the I^2C peripheral clock input.
STAT_CCR	CODE	R	700B_000Ch	[7:3]	I^2C peripheral status code. 00h: Bus error 01h: START condition transmitted 02h: Repeated START condition transmitted 03h: Address + write bit transmitted, ACK received 04h: Address + write bit transmitted, ACK not received 05h: Data byte transmitted in master mode, ACK received 06h: Data byte transmitted in master mode, ACK not received 07h: Arbitration lost in address or data byte 08h: Address + read bit transmitted, ACK received 09h: Address + read bit transmitted, ACK not received 0Ah: Data byte received in master mode, ACK received 0Bh: Data byte received in master mode, not ACK received 0Ch: Slave address + write bit received, ACK transmitted

Register name	Field name	Access	Address	Bits	Description
					0Dh: Arbitration lost in address as master, slave address + write bit received, ACK transmitted 0Eh: General call address received, ACK transmitted 0Fh: Arbitration lost in address as master, general call address received, ACK transmitted 10h: Data byte received after slave address received, ACK transmitted 11h: Data byte received after slave address received, not ACK transmitted 12h: Data byte received after general call address received, ACK transmitted 13h: Data byte received after general call address received, not ACK transmitted 14h: STOP or repeated START condition received in slave mode 15h: Slave address + read bit received, ACK transmitted 16h: Arbitration lost in address as master, slave address + read bit received, ACK transmitted 17h: Data byte transmitted in slave mode, ACK received 18h: Data byte transmitted in slave mode, ACK not received 19h: Last byte transmitted in slave mode, ACK received 1Ah: Second address byte + write bit transmitted, ACK received 1Bh: Second address byte + write bit transmitted, ACK not received 1Ch: Unused 1Dh: Unused 1Eh: Unused 1Fh: No relevant status information, IFLG=0 If an illegal condition occurs on the I^2C bus, the bus error state is entered (status code 00h). To recover from this state, the CNTR.STP bit must be set and the CNTR.IFLG bit cleared.

Register name	Field name	Access	Address	Bits	Description
					The peripheral will then return to idle state (status code F8h) and no STOP condition will be transmitted on the I^2C bus. Note: To request resumption of transmission, set the STA bit to “1” at the same time as the STP bit is set. The peripheral will then send a START on recovery from the bus error.
STAT_CCR	M	W	700B_000Ch	[6:3]	Along with STAT_CCR.N, controls the frequency at which the I^2C bus is sampled and the frequency of the clock line (SCL) when in master mode. $F_{samp} = F_{clock} / 2^N$ $F_{SCL} = F_{clock} / (2^N * (M+1) * 10)$ where F_{clock} is the frequency of the I^2C peripheral clock input.
XADDR	SLAX	RW	700B_0010h	[7:0]	Extended slave address bits [7:0] (SLAX7..SLAX0). When ADDR.SLA[4:0]=11110b, the I^2C peripheral recognizes this as the first part of a 10-bit address and if the next two bits match ADDR.SLAX[1:0] it sends an ACK. If the next byte of the address matches XADDR.SLAX[7:0] the I^2C peripheral generates an interrupt and goes into slave mode.
SRST	RST	W	700B_001Ch	[6:0]	Software reset. A software reset of the I^2C peripheral is applied upon writing any value to this register. A software reset sets the peripheral back to IDLE (STAT.CODE=F8h) and sets the STP, STA and IFLG bits in the CNTR register to “0”

15.12 CRC module

The CRC module is a HW accelerator to support FW CRC calculations. The CRC module is implemented in a general-purpose way through configuration registers, to allow high configurability of the algorithm that needs to be realized.

The CRC module supports:

- Initial value definition
- Polynomial definition
- Final XOR configuration
- Data input reflection
- Data output reflection

CRC HW implementation is shown in Figure 122.

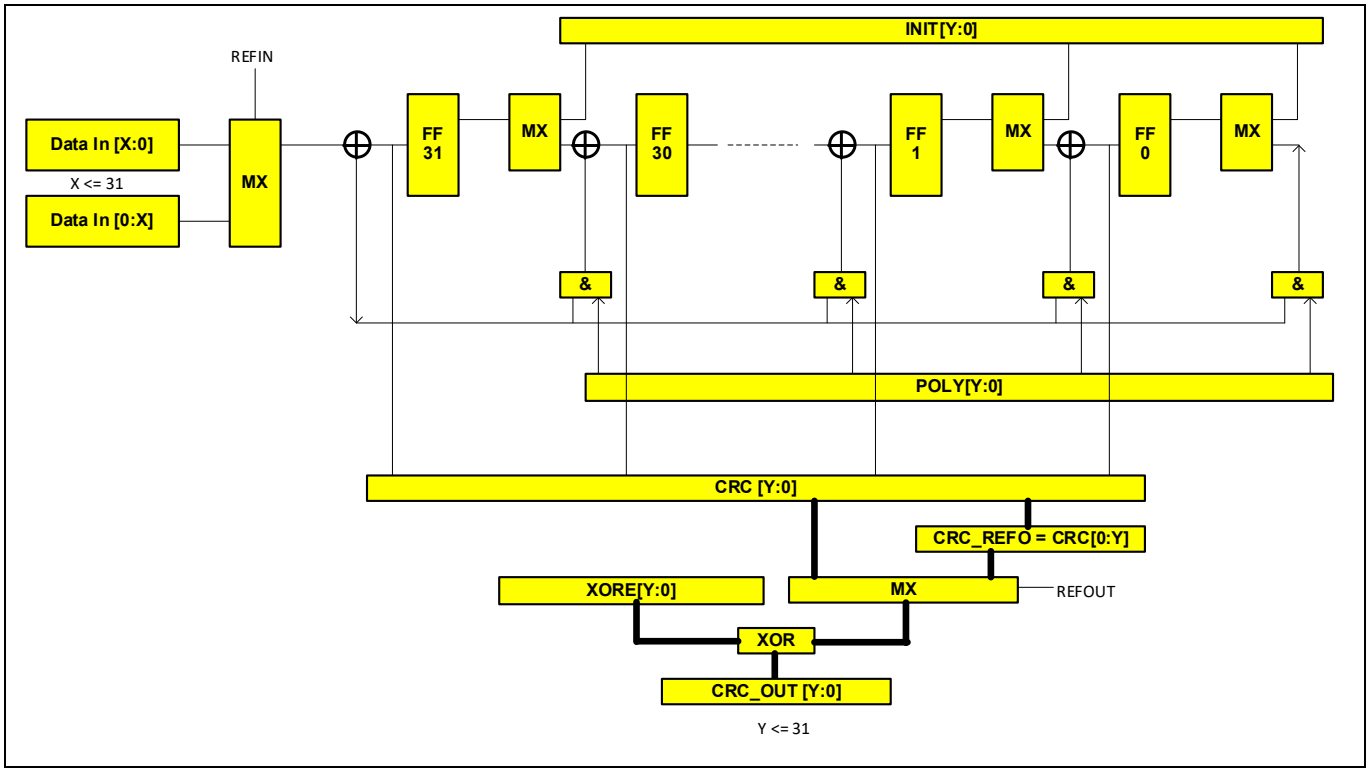

Figure 122 CRC HW implementation scheme

15.12.1 CRC registers

The relevant CRC-related registers and their descriptions are provided in Table 117.

Table 117 CRC register descriptions

Register name	Field name	Access	Address	Bits	Description
DATA	VAL	RW	7009_0000h	[31:0]	This register is used to feed the CRC module and to read the CRC result. On write, feed the CRC. On read, return the result of the CRC calculation
INIT	VAL	W	7009_0004h	[31:0]	This register is used to initialize the CRC polynomial. The INIT value has to be adapted for polynomial orders lower than 32 and must be left-aligned

Register name	Field name	Access	Address	Bits	Description
POLY	VAL	RW	7009_0008h	[31:0]	This register is a 32-bit read/write register, used to configure the polynomial of the CRC. The maximum order of the supported polynomial is 32. A typical polynomial will be: P(x)=x^32+x^26+x^23+x^22+x^16+x^12+x^11+x^10+x^8+x^7+x^5+x^4+x^2+x^1+x^0, i.e., VAL=0x04C11DB7. A lower order can be programmed by just left-shifting the polynomial. For example, to configure a P(x) = x^8+x^5+x^4+x^0, write a value of 0x31000000. In the same way, to configure a P(x) = x^16+x^12+x^5+x^0, write a value of 0x10210000
XORE	VAL	RW	7009_000Ch	[31:0]	This register is used to configure the final XOR of the CRC output. Typically set to 0xFFFFFFFF
CNFG	REFIN	RW	7009_0010h	[0]	Enable/Disable reflected CRC data input. 0: CRC data input is not reflected 1: CRC data input is reflected
CNFG	REFOUT	RW	7009_0010h	[1]	Enable/Disable reflected CRC data output. 0: CRC data output is not reflected 1: CRC data output is reflected

15.13 UART

The UART module is an Arm® PrimeCell IP (PL011); extensive documentation can be found in the “Arm® PrimeCell UART (PL011) Technical Reference Manual”.

The UART provides the following:

- Programmable use of UART or IrDA SIR input/output
- Separate 32×8 transmit and 32×12 receive FIFO memory buffers to reduce CPU interrupts
- Programmable FIFO disabling for 1-byte depth
- Programmable baud rate generator. This enables division of the reference clock by (1×16) to (65535 × 16) and generates an internal × 16 clock. The divisor can be a fractional number enabling use of any clock with a frequency greater than 3.6864 MHz as the reference clock
- Standard asynchronous communication bits (start, stop and parity). These are added prior to transmission and removed on reception
- Independent masking of transmit FIFO, receive FIFO, receive timeout, modem status, and error condition interrupts
- Support for direct memory access (DMA)
- False start bit detection
- Line-break generation and detection
- Support of the modem control functions CTS, DCD, DSR, RTS, DTR, and RI

- Programmable HW flow control
- Fully programmable serial interface characteristics:
  - data can be 5, 6, 7, or 8 bits
  - even, odd, stick, or no-parity bit generation and detection
  - 1 or 2 stop bit generation
  - baud rate generation, DC up to UARTCLK/16
- IrDA SIR ENDEC block providing:
  - programmable use of IrDA SIR or UART input/output
  - support of IrDA SIR ENDEC functions for data rates up to 115200 bps half-duplex
  - support of normal 3/16 and low-power (1.41 to 2.23 µs) bit durations

15.13.1 UART block diagram

Figure 123 shows a block diagram of the UART module.

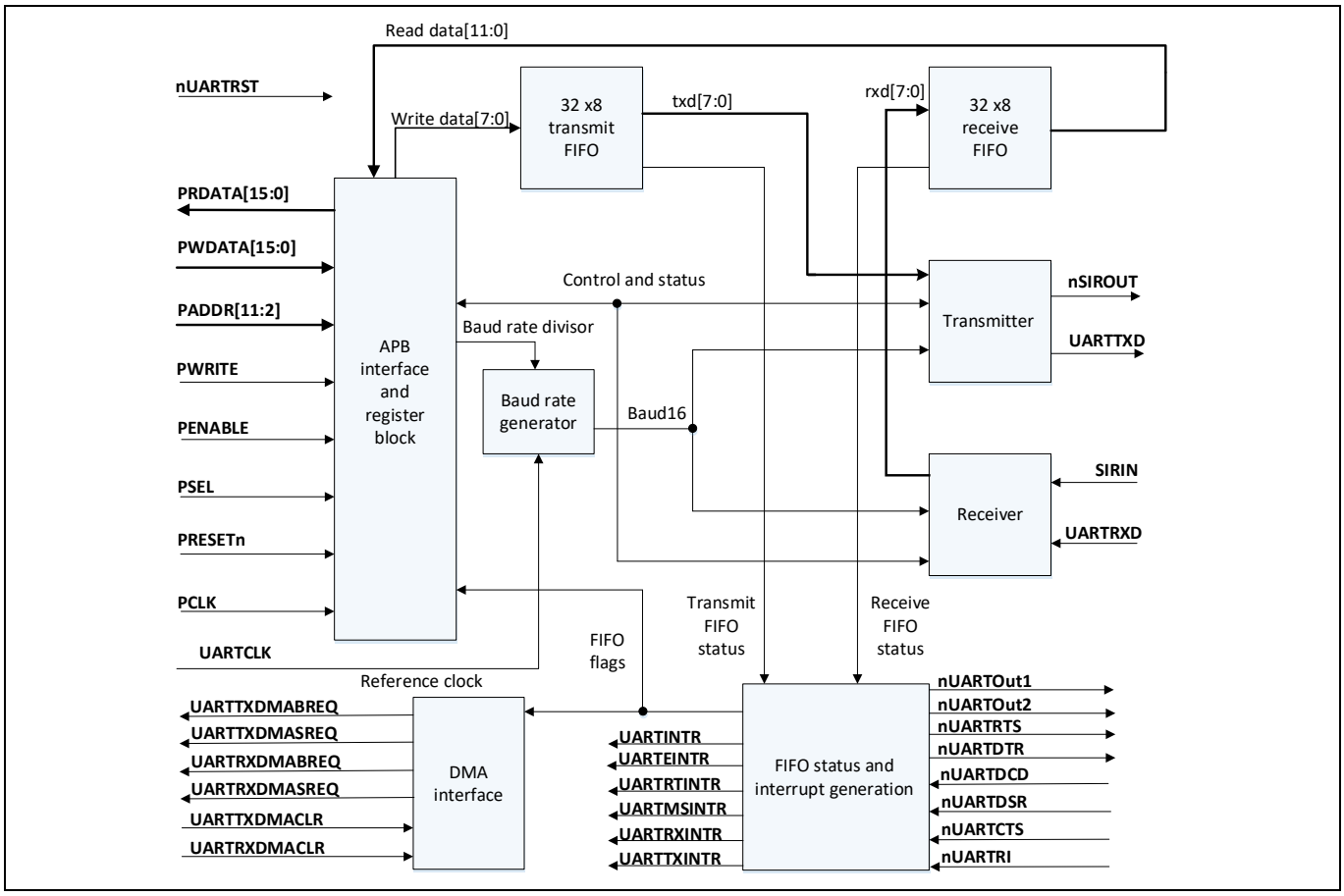

Figure 123 UART block diagram

15.13.2 UART registers

The relevant UART-related registers and their descriptions are provided in Table 118.

Table 118 UART-related register description

Register name	Field name	Access	Address	Bits	Description
UARTDR	DATA	RW	700C_0000h	[7:0]	Data character. Receive (read) data character. Transmit (write) data character
UARTDR	FE	R	700C_0000h	[8]	Framing error. When set to 1, it indicates that the received character did not have a valid stop bit (a valid stop bit is 1). In FIFO mode, this error is associated with the character at the top of the FIFO
UARTDR	PE	R	700C_0000h	[9]	Parity error. When set to 1, it indicates that the parity of the received data character does not match the parity that the EPS and SPS bits in the line control register, UARTLCR_H on pages 3 to 12, select. In FIFO mode, this error is associated with the character at the top of the FIFO
UARTDR	BE	R	700C_0000h	[10]	Break error. This bit is set to 1 if a break condition was detected, indicating that the received data input was held low for longer than a full-word transmission time (defined as start, data, parity and stop bits). In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state), and the next valid start bit is received
UARTDR	OE	R	700C_0000h	[11]	Overrun error. This bit is set to 1 if data is received and the receive FIFO is already full. This is cleared to 0 once there is an empty space in the FIFO and a new character can be written to it
UARTRSR_UARTECR	FE	RW	700C_0004h	[0]	Framing error. When set to 1, it indicates that the received character did not have a valid stop bit (a valid stop bit is 1). This bit is cleared to 0 by a write to UARTECR. In FIFO mode, this error

Register name	Field name	Access	Address	Bits	Description
					is associated with the character at the top of the FIFO
UARTRSR_UARTECR	PE	RW	700C_0004h	[1]	Parity error. When set to 1, it indicates that the parity of the received data character does not match the parity that the EPS and SPS bits in the line control register, UARTLCR_H on page 31, select. This bit is cleared to 0 by a write to UARTECR. In FIFO mode, this error is associated with the character at the top of the FIFO
UARTRSR_UARTECR	BE	RW	700C_0004h	[2]	Break error. This bit is set to 1 if a break condition was detected, indicating that the received data input was held low for longer than a full-word transmission time (defined as start, data, parity, and stop bits). This bit is cleared to 0 after a write to UARTECR. In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state) and the next valid start bit is received
UARTRSR_UARTECR	OE	RW	700C_0004h	[3]	Overrun error. This bit is set to 1 if data is received and the FIFO is already full. This bit is cleared to 0 by a write to UARTECR. The FIFO contents remain valid because no more data is written when the FIFO is full; only the contents of the shift register are overwritten. The CPU must now read the data, to empty the FIFO
UARTFR	CTS	R	700C_0018h	[0]	Clear to send. This bit is the complement of the UART clear to send, nUARTCTS, modem status input. That is, the bit is 1 when nUARTCTS is low
UARTFR	DSR	R	700C_0018h	[1]	Data set ready. This bit is the complement of the UART data set ready, nUARTDSR, modem status

Register name	Field name	Access	Address	Bits	Description
					input. That is, the bit is 1 when nUARTDSR is low
UARTFR	DCD	R	700C_0018h	[2]	Data carrier detect. This bit is the complement of the UART data carrier detect, nUARTDCD, modem status input. That is, the bit is 1 when nUARTDCD is low
UARTFR	BUSY	R	700C_0018h	[3]	UART busy. If this bit is set to 1, the UART is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register. This bit is set as soon as the transmit FIFO becomes non-empty, regardless of whether the UART is enabled or not
UARTFR	RXFE	R	700C_0018h	[4]	Receive FIFO empty. The meaning of this bit depends on the state of the FEN bit in the line control register, UARTLCR_H. If the FIFO is disabled, this bit is set when the receive holding register is empty. If the FIFO is enabled, the RXFE bit is set when the receive FIFO is empty
UARTFR	TXFF	R	700C_0018h	[5]	Transmit FIFO full. The meaning of this bit depends on the state of the FEN bit in the line control register, UARTLCR_H. If the FIFO is disabled, this bit is set when the transmit holding register is full. If the FIFO is enabled, the TXFF bit is set when the transmit FIFO is full
UARTFR	RXFF	R	700C_0018h	[6]	Receive FIFO full. The meaning of this bit depends on the state of the FEN bit in the line control register, UARTLCR_H. If the FIFO is disabled, this bit is set when the receive holding register is full. If the FIFO is enabled, the RXFF bit is set when the receive FIFO is full
UARTFR	TXFE	R	700C_0018h	[7]	Transmit FIFO empty. The meaning of this bit depends on the state of the FEN bit in the line control register, UARTLCR_H on page 31. If the FIFO is disabled,

Register name	Field name	Access	Address	Bits	Description
					this bit is set when the transmit holding register is empty. If the FIFO is enabled, the TXFE bit is set when the transmit FIFO is empty. This bit does not indicate if there is data in the transmit shift register
UARTFR	RI	R	700C_0018h	[8]	Ring indicator. This bit is the complement of the UART ring indicator, nUARTRI, modem status input. That is, the bit is 1 when nUARTRI is low
UARTILPR	ILPDVSR	RW	700C_0020h	[7:0]	The UARTILPR register is the IrDA low-power counter register. This is an 8-bit read/write register that stores the low-power counter divisor value used to generate the IrLPBaud16 signal by dividing down UARTCLK. The IrLPBaud16 signal is generated by dividing down the UARTCLK signal according to the low-power divisor value written to the UARTILPR register. The low-power divisor value is calculated as follows: Low-power divisor (ILPDVSR) = (FUARTCLK / FIrLPBaud16) where FIrLPBaud16 is nominally 1.8432 MHz. You must select the divisor so that 1.42 MHz less than FIrLPBaud16 less than 2.12 MHz results in a low-power pulse duration of 1.41 to 2.11 µs (three times the period of IrLPBaud16).
UARTIBRD	BAUD_DIVINT	RW	700C_0024h	[15:0]	The UARTIBRD register is the integer part of the baud rate divisor value
UARTFBRD	BAUD_DIVFRAC	RW	700C_0028h	[5:0]	The UARTFBRD register is the fractional part of the baud rate divisor value. The baud rate divisor is calculated as follows: Baud rate divisor BAUDDIV = (FUARTCLK/(16 × baud rate))

Register name	Field name	Access	Address	Bits	Description
					where FUARTCLK is the UART reference clock frequency. The BAUDDIV is comprised of the integer value (BAUD_DIVINT) and the fractional value (BAUD _DIVFRAC). Example: If the required baud rate is 230400 and UARTCLK = 4 MHz then: Baud rate divisor = (4 × 106)/(16 × 230400) = 1.085 This means BRDI = 1 and BRDF = 0.085. Therefore, fractional part, m = integer((0.085 × 64) + 0.5) = 5 Generated baud rate divider = 1 + 5/64 = 1.078 Generated baud rate = (4 × 106)/(16 × 1.078) = 231911 Error = (231911 - 230400)/230400 × 100 = 0.656 percent The maximum error using a 6-bit UARTFBRD register = 1/64 × 100 = 1.56 percent. This occurs when m = 1, and the error is cumulative over 64 clock ticks.
UARTLCR_H	BRK	RW	700C_002Ch	[0]	Send break. If this bit is set to 1, a low level is continually output on the UARTTXD output, after completing transmission of the current character. For the proper execution of the break command, the software must set this bit for at least two complete frames. For normal use, this bit must be cleared to 0
UARTLCR_H	PEN	RW	700C_002Ch	[1]	Parity enable. 0: Parity is disabled and no parity bit added to the data frame 1: Parity checking and generation is enabled
UARTLCR_H	EPS	RW	700C_002Ch	[2]	Even parity select. Controls the type of parity the UART uses during transmission and reception. This bit has no effect when the PEN bit disables parity

Register name	Field name	Access	Address	Bits	Description
					checking and generation. See Figure 17-13 on page 31 for the parity truth table. 0: Odd parity. The UART generates or checks for an odd number of 1s in the data and parity bits. 1: Even parity. The UART generates or checks for an even number of 1s in the data and parity bits
UARTLCR_H	STP2	RW	700C_002Ch	[3]	Two stop bits select. If this bit is set to 1, two stop bits are transmitted at the end of the frame. The receive logic does not check for two stop bits being received
UARTLCR_H	FEN	RW	700C_002Ch	[4]	Enable FIFOs. 0: FIFOs are disabled (character mode); that is, the FIFOs become 1-byte-deep holding registers 1: Transmit and receive FIFO buffers are enabled (FIFO mode)
UARTLCR_H	WLEN	RW	700C_002Ch	[5]	Word length. These bits indicate the number of data bits transmitted or received in a frame as follows: 0: 5 bits 1: 6 bits 2: 7 bits 3: 8 bits
UARTLCR_H	SPS	RW	700C_002Ch	[7]	Stick parity select. This bit has no effect when the PEN bit disables parity checking and generation. 0: Low stick parity is disabled 1: High either: If the EPS bit is 0 then the parity bit is transmitted and checked as a 1 Or if the EPS bit is 1 then the parity bit is transmitted and checked as a 0
UARTCR	UARTEN	RW	700C_0030h	[0]	UART enable. 0: UART is disabled. If the UART is disabled in the middle of transmission or reception, it

Register name	Field name	Access	Address	Bits	Description
					completes the current character before stopping. 1: UART is enabled. Data transmission and reception occurs for either UART signals or SIR signals depending on the setting of the SIREN bit.
UARTCR	SIREN	RW	700C_0030h	[1]	SIR enable. This bit has no effect if the UARTEN bit disables the UART. 0: IrDA SIR ENDEC is disabled. nSIROUT remains low (no light pulse generated), and signal transitions on SIRIN have no effect. 1: IrDA SIR ENDEC is enabled. Data is transmitted and received on nSIROUT and SIRIN. UARTTXD remains high, in the marking state. Signal transitions on UARTRXD or modem status inputs have no effect.
UARTCR	SIRLP	RW	700C_0030h	[2]	SIR low-power IrDA mode. This bit selects the IrDA encoding mode. If this bit is cleared to 0, low-level bits are transmitted as an active high pulse with a width of 3/16 of the bit period. If this bit is set to 1, low-level bits are transmitted with a pulse width that is three times the period of the IrLPBaud16 input signal, regardless of the selected bit rate. Setting this bit uses less power, but might reduce transmission distances
UARTCR	LBE	RW	700C_0030h	[7]	Loopback enable. If this bit is set to 1 and the SIREN bit is set to 1 and the SIRTEST bit in the test control register, UARTTCR is set to 1, then the nSIROUT path is inverted, and fed through to the SIRIN path. The SIRTEST bit in the test register must be set to 1 to override the normal half-duplex SIR operation. This must be the requirement for accessing the test registers during normal operation, and SIRTEST must be cleared to 0

Register name	Field name	Access	Address	Bits	Description
					when loopback testing is finished. This feature reduces the amount of external coupling required during system test. If this bit is set to 1, and the SIRTEST bit is set to 0, the UARTTXD path is fed through to the UARTRXD path. In either SIR mode or UART mode, when this bit is set, the modem outputs are also fed through to the modem inputs. This bit is cleared to 0 on reset, to disable loopback.
UARTCR	TXE	RW	700C_0030h	[8]	Transmit enable. If this bit is set to 1, the transmit section of the UART is enabled. Data transmission occurs for either UART signals, or SIR signals depending on the setting of the SIREN bit. When the UART is disabled in the middle of transmission, it completes the current character before stopping
UARTCR	RXE	RW	700C_0030h	[9]	Receive enable. If this bit is set to 1, the receive section of the UART is enabled. Data reception occurs for either UART signals or SIR signals depending on the setting of the SIREN bit. When the UART is disabled in the middle of reception, it completes the current character before stopping
UARTCR	DTR	RW	700C_0030h	[10]	Data transmit ready. This bit is the complement of the UART data transmit ready, nUARTDTR, modem status output. That is, when the bit is programmed to 1 then nUARTDTR is LOW
UARTCR	RTS	RW	700C_0030h	[11]	Request to send. This bit is the complement of the UART request to send, nUARTRTS, modem status output. That is, when the bit is programmed to 1 then nUARTRTS is LOW
UARTCR	OUT1	RW	700C_0030h	[12]	Complement of the UART Out1 modem status output. This bit is

Register name	Field name	Access	Address	Bits	Description
					the complement of the UART Out1 (nUARTOut1) modem status output. That is, when the bit is programmed to 1 the output is 0. For DTE this can be used as data carrier detect (DCD)
UARTCR	OUT2	RW	700C_0030h	[13]	Complement of the UART Out2 modem status output. This bit is the complement of the UART Out2 (nUARTOut2) modem status output. That is, when the bit is programmed to 1, the output is 0. For DTE this can be used as ring indicator (RI)
UARTCR	RTSEn	RW	700C_0030h	[14]	RTS HW flow control enable. If this bit is set to 1, RTS HW flow control is enabled. Data is only requested when there is space in the receive FIFO for it to be received. 0: RTS HW flow control is disabled 1: RTS HW flow control is enabled
UARTCR	CTSEn	RW	700C_0030h	[15]	CTS HW flow control enable. If this bit is set to 1, CTS HW flow control is enabled. Data is only transmitted when the nUARTCTS signal is asserted. 0: CTS HW flow control is disabled 1: CTS HW flow control is enabled
UARTIFLS	TXIFLSEL	RW	700C_0034h	[2:0]	Transmit interrupt FIFO level select. The trigger points for the transmit interrupt are as follows: 0: Transmit FIFO becomes less than 1/8 full 1: Transmit FIFO becomes less than 1/4 full 2: Transmit FIFO becomes less than 1/2 full 3: Transmit FIFO becomes less than 3/4 full 4: Transmit FIFO becomes less than 7/8 full 5 to 7: Reserved
UARTIFLS	RXIFLSEL	RW	700C_0034h	[5:3]	Receive interrupt FIFO level select. The trigger points for the receive interrupt are as follows:

Register name	Field name	Access	Address	Bits	Description
					0: Receive FIFO becomes more than 1/8 full 1: Receive FIFO becomes more than 1/4 full 2: Receive FIFO becomes more than 1/2 full 3: Receive FIFO becomes more than 3/4 full 4: Receive FIFO becomes more than 7/8 full 5 to 7: Reserved
UARTIMSC	RIMIM	RW	700C_0038h	[0]	nUARTRI modem interrupt mask. A read returns the current mask for the UARTRIINTR interrupt. On a write of 1, the mask of the UARTRIINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	CTSMIM	RW	700C_0038h	[1]	nUARTCTS modem interrupt mask. A read returns the current mask for the UARTCTSINTR interrupt. On a write of 1, the mask of the UARTCTSINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	DCDMIM	RW	700C_0038h	[2]	nUARTDCD modem interrupt mask. A read returns the current mask for the UARTDCDINTR interrupt. On a write of 1, the mask of the UARTDCDINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	DSRMIM	RW	700C_0038h	[3]	nUARTDSR modem interrupt mask. A read returns the current mask for the UARTDSRINTR interrupt. On a write of 1, the mask of the UARTDSRINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	RXIM	RW	700C_0038h	[4]	Receive interrupt mask. A read returns the current mask for the UARTRXINTR interrupt. On a write of 1, the mask of the UARTRXINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	TXIM	RW	700C_0038h	[5]	Transmit interrupt mask. A read returns the current mask for the

Register name	Field name	Access	Address	Bits	Description
					UARTTXINTR interrupt. On a write of 1, the mask of the UARTTXINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	RTIM	RW	700C_0038h	[6]	Receive timeout interrupt mask. A read returns the current mask for the UARTRTINTR interrupt. On a write of 1, the mask of the UARTRTINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	FEIM	RW	700C_0038h	[7]	Framing error interrupt mask. A read returns the current mask for the UARTFEINTR interrupt. On a write of 1, the mask of the UARTFEINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	PEIM	RW	700C_0038h	[8]	Parity error interrupt mask. A read returns the current mask for the UARTPEINTR interrupt. On a write of 1, the mask of the UARTPEINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	BEIM	RW	700C_0038h	[9]	Break error interrupt mask. A read returns the current mask for the UARTBEINTR interrupt. On a write of 1, the mask of the UARTBEINTR interrupt is set. A write of 0 clears the mask
UARTIMSC	OEIM	RW	700C_0038h	[10]	Overrun error interrupt mask. A read returns the current mask for the UARTOEINTR interrupt. On a write of 1, the mask of the UARTOEINTR interrupt is set. A write of 0 clears the mask
UARTRIS	RIRMIS	R	700C_003Ch	[0]	nUARTRI modem interrupt status. Returns the raw interrupt state of the UARTRIINTR interrupt
UARTRIS	CTSRMIS	R	700C_003Ch	[1]	nUARTCTS modem interrupt status. Returns the raw interrupt state of the UARTCTSINTR interrupt
UARTRIS	DCDRMIS	R	700C_003Ch	[2]	nUARTDCD modem interrupt status. Returns the raw interrupt state of the UARTDCDINTR interrupt

Register name	Field name	Access	Address	Bits	Description
UARTRIS	DSRRMIS	R	700C_003Ch	[3]	nUARTDSR modem interrupt status. Returns the raw interrupt state of the UARTDSRINTR interrupt
UARTRIS	RXRIS	R	700C_003Ch	[4]	Receive interrupt status. Returns the raw interrupt state of the UARTRXINTR interrupt
UARTRIS	TXRIS	R	700C_003Ch	[5]	Transmit interrupt status. Returns the raw interrupt state of the UARTTXINTR interrupt
UARTRIS	RTRIS	R	700C_003Ch	[6]	Receive timeout interrupt status. Returns the raw interrupt state of the UARTRTINTR interrupt
UARTRIS	FERIS	R	700C_003Ch	[7]	Framing error interrupt status. Returns the raw interrupt state of the UARTFEINTR interrupt
UARTRIS	PERIS	R	700C_003Ch	[8]	Parity error interrupt status. Returns the raw interrupt state of the UARTPEINTR interrupt
UARTRIS	BERIS	R	700C_003Ch	[9]	Break error interrupt status. Returns the raw interrupt state of the UARTBEINTR interrupt
UARTRIS	OERIS	R	700C_003Ch	[10]	Overrun error interrupt status. Returns the raw interrupt state of the UARTOEINTR interrupt
UARTMIS	RIMMIS	R	700C_0040h	[0]	nUARTRI modem masked interrupt status. Returns the masked interrupt state of the UARTRIINTR interrupt
UARTMIS	CTSMMIS	R	700C_0040h	[1]	nUARTCTS modem masked interrupt status. Returns the masked interrupt state of the UARTCTSINTR interrupt
UARTMIS	DCDMMIS	R	700C_0040h	[2]	nUARTDCD modem masked interrupt status. Returns the masked interrupt state of the UARTDCDINTR interrupt
UARTMIS	DSRMMIS	R	700C_0040h	[3]	nUARTDSR modem masked interrupt status. Returns the masked interrupt state of the UARTDSRINTR interrupt
UARTMIS	RXMIS	R	700C_0040h	[4]	Receive masked interrupt status. Returns the masked interrupt state of the UARTRXINTR interrupt

Register name	Field name	Access	Address	Bits	Description
UARTMIS	TXMIS	R	700C_0040h	[5]	Transmit masked interrupt status. Returns the masked interrupt state of the UARTTXINTR interrupt
UARTMIS	RTMIS	R	700C_0040h	[6]	Receive timeout masked interrupt status. Returns the masked interrupt state of the UARTRTINTR interrupt
UARTMIS	FEMIS	R	700C_0040h	[7]	Framing error masked interrupt status. Returns the masked interrupt state of the UARTFEINTR interrupt
UARTMIS	PEMIS	R	700C_0040h	[8]	Parity error masked interrupt status. Returns the masked interrupt state of the UARTPEINTR interrupt
UARTMIS	BEMIS	R	700C_0040h	[9]	Break error masked interrupt status. Returns the masked interrupt state of the UARTBEINTR interrupt
UARTMIS	OEMIS	R	700C_0040h	[10]	Overrun error masked interrupt status. Returns the masked interrupt state of the UARTOEINTR interrupt
UARTICR	RIMIC	W	700C_0044h	[0]	nUARTRI modem interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	CTSMIC	W	700C_0044h	[1]	nUARTCTS modem interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	DCDMIC	W	700C_0044h	[2]	nUARTDCD modem interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	DSRMIC	W	700C_0044h	[3]	nUARTDSR modem interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	RXIC	W	700C_0044h	[4]	Receive interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	TXIC	W	700C_0044h	[5]	Transmit interrupt clear. 0: No effect 1: Clear interrupt

Register name	Field name	Access	Address	Bits	Description
UARTICR	RTIC	W	700C_0044h	[6]	Receive timeout interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	FEIC	W	700C_0044h	[7]	Framing error interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	PEIC	W	700C_0044h	[8]	Parity error interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	BEIC	W	700C_0044h	[9]	Break error interrupt clear. 0: No effect 1: Clear interrupt
UARTICR	OEIC	W	700C_0044h	[10]	Overrun error interrupt clear. 0: No effect 1: Clear interrupt
UARTDMACR	RXDMAE	RW	700C_0048h	[0]	Receive DMA enable. If this bit is set to 1, DMA for the receive FIFO is enabled
UARTDMACR	TXDMAE	RW	700C_0048h	[1]	Transmit DMA enable. If this bit is set to 1, DMA for the transmit FIFO is enabled
UARTDMACR	DMAONERR	RW	700C_0048h	[2]	DMA on error. If this bit is set to 1, the DMA receive request outputs, UARTRXDMASREQ or UARTRXDMABREQ, are disabled when the UART error interrupt is asserted
UARTPeriphID0	PARTNUMBER0	R	700C_0FE0h	[7:0]	PARTNUMBER[7:0]. Together with the upper bits from PARTNUMBER1, PARTNUMBER[11:0] identifies the peripheral. In this case the three-digit product code, 0x011, is returned
UARTPeriphID1	PARTNUMBER1	R	700C_0FE4h	[3:0]	PARTNUMBER[11:8]. Together with the lower bits from PARTNUMBER0, PARTNUMBER[11:0] identifies the peripheral. In this case the three-digit product code, 0x011, is returned
UARTPeriphID1	DESIGNER0	R	700C_0FE4h	[7:4]	DESIGNER[3:0]. Together with the upper bits from DESIGNER1, DESIGNER[7:0] identifies the

Register name	Field name	Access	Address	Bits	Description
					peripheral designer. In this case it returns 0x41, indicating Arm® Ltd
UARTPeriphID2	DESIGNER1	R	700C_0FE8h	[3:0]	DESIGNER[7:4]. Together with the lower bits from DESIGNER0, DESIGNER[7:0] identifies the peripheral designer. In this case it returns 0x41, indicating Arm® Ltd
UARTPeriphID2	REVISION	R	700C_0FE8h	[7:4]	Returns the peripheral revision number, with 0 indicating the initial revision
UARTPeriphID3	CONFIGURATION	R	700C_0FECh	[7:0]	Returns the configuration option of the peripheral
UARTPCellID0	UARTPCellID0	R	700C_0FF0h	[7:0]	UARTPCellID[7:0]. Together with the other cell ID registers, UARTPCellID[31:0] is used as a standard cross-peripheral ID system. In this case UARTPCellID[31:0] = 0xB105F00D
UARTPCellID1	UARTPCellID1	R	700C_0FF4h	[7:0]	UARTPCellID[15:8]. Together with the other cell ID registers, UARTPCellID[31:0] is used as a standard cross-peripheral ID system. In this case UARTPCellID[31:0] = 0xB105F00D
UARTPCellID2	UARTPCellID2	R	700C_0FF8h	[7:0]	UARTPCellID[23:16]. Together with the other cell ID registers, UARTPCellID[31:0] is used as a standard cross-peripheral ID system. In this case UARTPCellID[31:0] = 0xB105F00D
UARTPCellID3	UARTPCellID3	R	700C_0FFCh	[7:0]	UARTPCellID[31:24]. Together with the other cell ID registers, UARTPCellID[31:0] is used as a standard cross-peripheral ID system. In this case UARTPCellID[31:0] = 0xB105F00D

15.14 Debugger port

The debugger port on XDPP1100 is implemented through the Cortex®-M0 serial wire debugger (SWD) interface, a two-wire serial protocol that is used to access the Cortex® debug access point (DAP).

Cortex® DAP is a specific HW, integrated into the M0 microcontroller, that can take the control of the execution flow to allow an external debugger to access every register (on the Cortex®-M0 memory map), to halt/step/resume any code execution, to dump every memory (ROM, RAM, OTP), to write any writable memory (RAM, OTP) or to insert code breakpoints.

XDPP1100 registers can also be accessed using an I^2C interface. The main difference between these two methodologies is that while SWD is HW decoded by the DAP (no FW required), I^2C is interpreted by the FW, so in this case, the success of the access depends on the FW status (for example FW boot needs to be completed, internal tasks have to permit the I^2C driver to run, latency cannot be guaranteed).
Cortex® DAP structure, at high level, is shown in Figure 124.

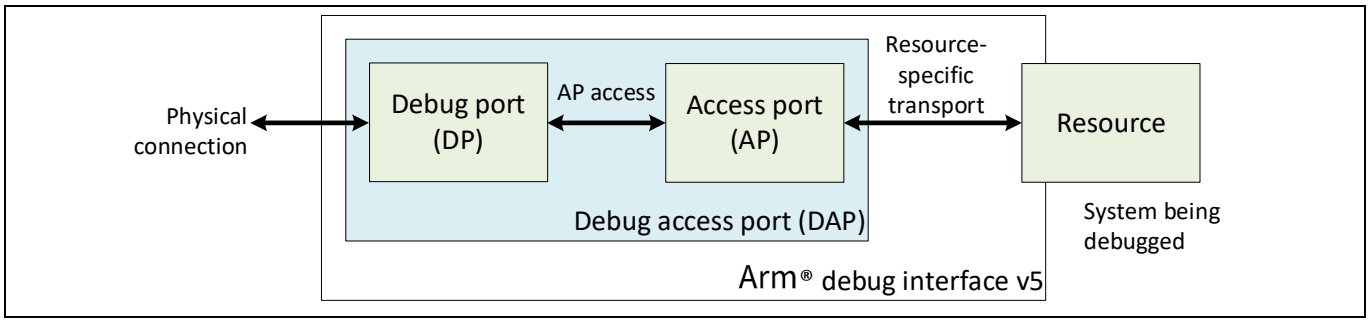

Figure 124 Arm® DAP structure

15.14.1 Serial wire debugger interface

SWD interface has two signals:

- SWCLK – input serial clock
- SWDIO – bidirectional serial data

The serial clock line is generally driven by the external host. Clock speed is fixed for all the transactions of a debug session and depends on target-specific implementation (it can normally be configured from 1 MHz to 20 MHz).

XDPP1100 can achieve up to 20 MHz clock speed for SWCLK, but because SWCLK is asynchronous with respect to any internal device clock and proper synchronizations are in place in the Cortex® DAP, it is important that a minimum division ratio of 4 is maintained between the internal core clock and the debugger clock itself. That means that 20 MHz can be achieved when the core clock is at its nominal 100 MHz frequency, but it should be reduced if the core clock is running at a lower speed.

The serial data line must be driven both by the debugger and XDPP1100 (bidirectional communication protocol), to avoid contention. The SWD protocol defines slots for data transmitting/receiving and turnaround periods.

The host driving the line low, when the clock is applied, is interpreted by the target as “idle cycle”.

On the XDPP1100, due to IO limitations, SWD pins are shared with other functionalities (as alternate functions), and because debug is not an application function, they are by default not available at power-up during a functional boot.

The relevant pins for the SWD interface are described in Table 119.

Table 119 Debugger interface pin mapping

Pin name	XDPP1100-Q040 Pin number	XDPP1100-Q024 Pin number	Debug function	Description
XADDR1	7	5	TEST_GATE	To detect and enter debug mode
SYNC	38	22	SWCLK	Serial clock
PWRGD	40	24	SWDIO	Serial bidirectional data

XADDR1's primary role, in functional mode, is to measure an external resistor to select the configuration address for the I^2C/PMBus interface.

But XADDR1, if kept at 3.3 V during power-up, enables the debugger interface to be available on SYNC and PWRGD IOs (configuring the multiplexer that selects SWCLK on SYNC and SWDIO on PWRGD).

The XADDR1 is latched to 250 µs after power-on-reset, so to have a stable value during the latching phase, it has to be kept at 3.3 V before the power-up of the XDPP1100 3.3 V supply.

After the latching phase, the value is maintained until a chip power-down or a chip reset occurs, so XADDR1 can be reprogrammed by FW for other functionalities.

The circuit to implement this feature is shown in Figure 125.

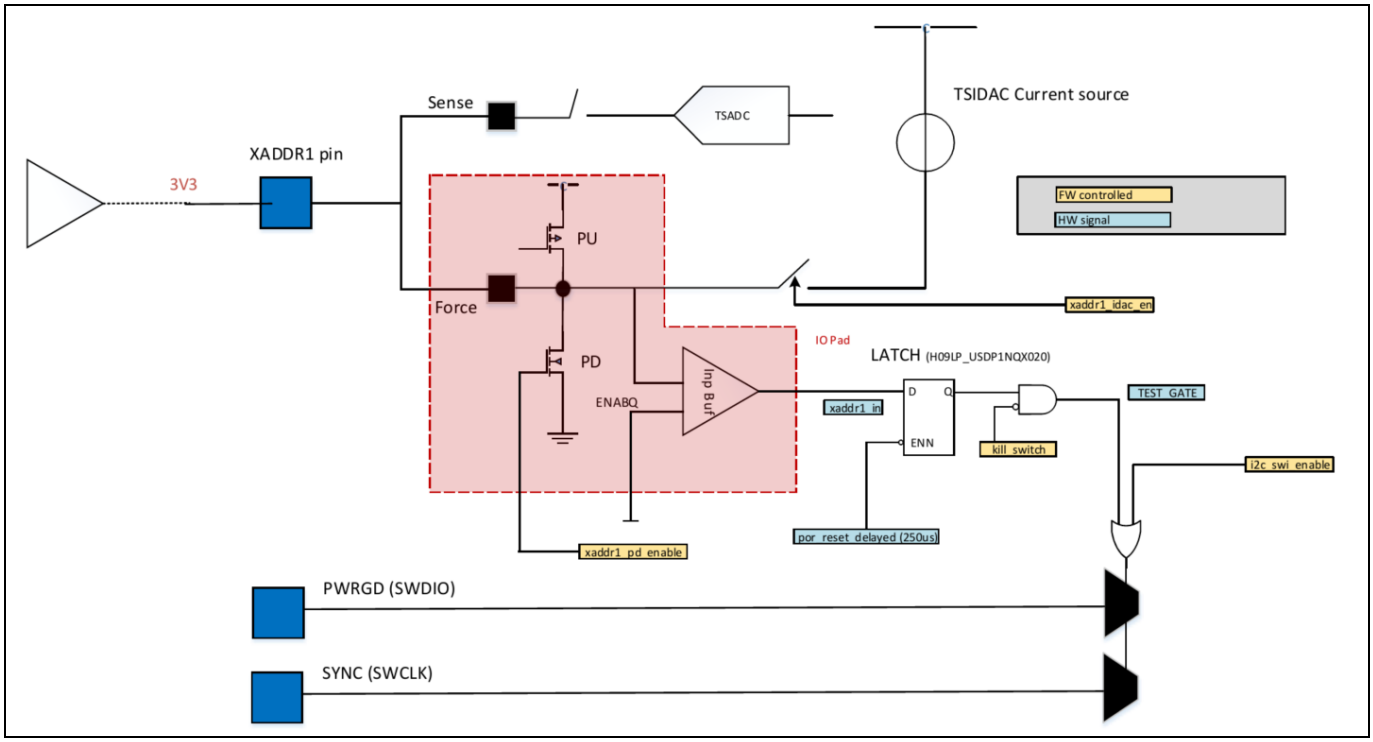

Figure 125 SWD enabling logic on XADDR1

In functional mode, TEST_GATE is latched to Logic 0 during power-up (XADDR1 is connected to an external resistor), so SWD IOs cannot be enabled by mistake.

The XADDR1 pin can be used anyway for multi-configuration purposes after the power-up sequence. The FW ensures proper configuration of the logic later in the boot sequence. The timing waveform of this process is shown in Figure 126, where XADDR1 is used during power-up to latch the debugger condition, and later on it is used by the FW for functional purposes.

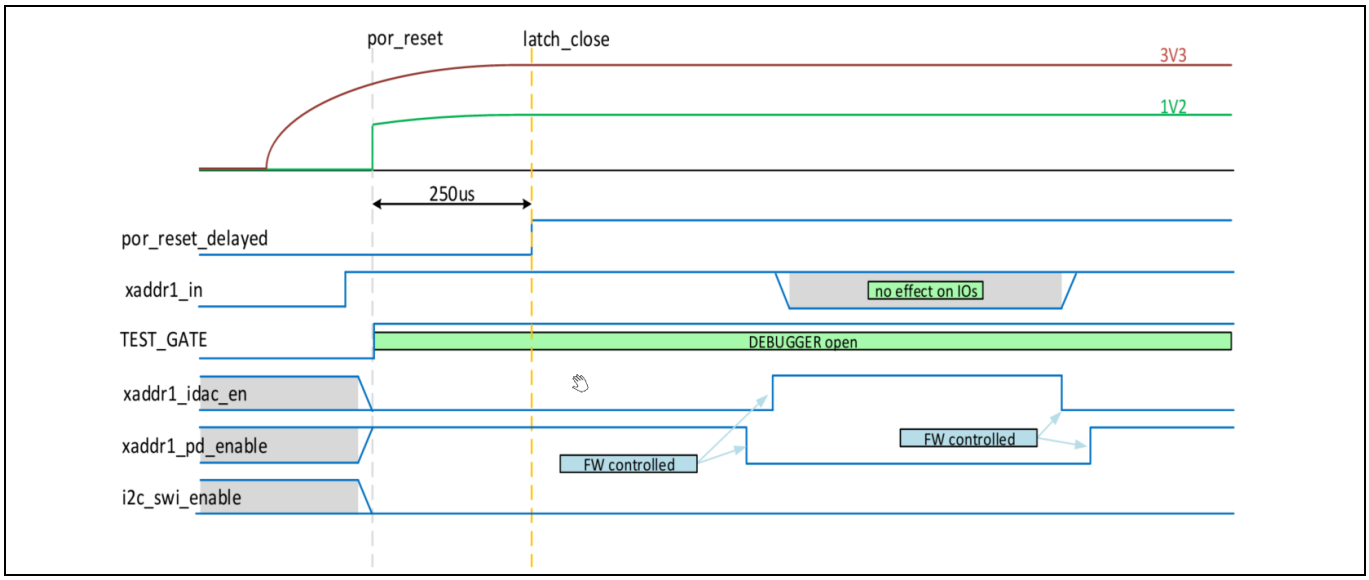

Figure 126 XADDR1 timing diagram

Additionally, the SWD interface can be enabled any time after the power-up, by writing in the CPUS_CFG register (DS_DBGPORT bit) using the I^2C interface (or FW).

16 Related resources

- Gate driver ICs
- Developer Community

Abbreviations or acronyms

Abbreviation	Expanded form
A	Ampere
ACF	Active clamp forward
ADC	Analog-to-digital converter
AFE	Analog front end
APC	Amps per code
BOM	Bill of materials
BW	Bandwidth
CDR	Current doubler topology on the secondary
CE	Current estimator
CPUS	Central processing unit subsystem
DAC	Digital-to-analog converter
dc	Delta code
DCM	Discontinuous conduction mode
DE	Dual edge
FB	Full-bridge
FBAL	Flux balance
FBFB	Full-bridge full-bridge rectifier
FEC	Front-end compensation
FF	Feed-forward
FW	Firmware
GPIO	General-purpose input output
GUI	Graphical user interface
HB	Half-bridge
HBCT	Half-bridge center tap
HIZ	High impedance
HPF	High-pass filter
HW	Hardware
IADC	Current sense analog-to-digital converter
IBAL	Current balance
IO	Input output
IPS	Integrated power stage
IS AFE	Current sense analog front end
ISHARE	Current sharing
ISP	Current sense processor
LE	Leading edge
LPF	Low-pass filter
LSB	Least significant bit

Abbreviations or acronyms

Abbreviation	Expanded form
LUT	Look-up table
NLC	Negative current limiting
NVM	Non-volatile memory
OC	Overcurrent
OSP	Open voltage sense fault protection
OT	Overtemperature
OTA	Operational transconductance amplifier
OTP	One-time programmable
OV	Overvoltage
PCB	Printed circuit board
PCL	Peak current limiting
PCMC	Peak current mode control
PI	Proportional-integral
PID	Proportional-integral-derivative
PPR	Pulses per revolution
PW	Pulse width
PWM	Pulse width modulation
RO	Read-only
ROM	Read-only memory
RPM	Revolutions per minute
RW	Read write
SEG	Segment
SCP	Short-circuit protection
SR	Synchronous rectification
SW	Software
TACH	Tachometer
TE	Trailing edge
TS	Telemetry sense
TSADC	Telemetry sense analog-to-digital converter
TS IDAC	Telemetry sense current digital-to-analog converter
TSP	Telemetry sense processor
UC	Undercurrent
UV	Undervoltage
VMC	Voltage mode control
VS	Voltage sense
VSADC	Voltage sense analog-to-digital converter
VSP	Voltage sense processor
XV	X-valent

Abbreviations or acronyms

Latin characters

Abbreviation	Expanded form
B	Magnetic flux density
BVREF_BVRREF	XDPP1100 input pin
BVSEN_BVRSEN	XDPP1100 input pin
BVOUT	Output voltage, Loop 1
BIREF	XDPP1100 input pin for current sensing
BISEN	XDPP1100 input pin for current sensing
C	Capacitor
COUT	Output filter capacitor
fz	Frequency of the compensator zero
fp	Frequency of the compensator pole
Fsample	Sampling frequency
Fswitch	Switching frequency
HPI	Filter magnitude response
IREF	XDPP1100 input pin for current sensing
IRQ	Interrupt
ISEN	XDPP1100 input pin for current sensing
IIN	Input current
IL	Inductor current
ILOAD	Load current
IOUT	Output current
KD	Derivative term of the compensator
KFP	Filter coefficient
KI	Integral term of the compensator
KP	Proportional term of the compensator
L	Inductor
Llk	Transformer leakage inductance
Lm	Transformer magnetizing inductance
LOUT	Output filter inductance
LPCB	PCB trace inductance
Loop 0	Control loop 0 associated with PMBus PAGE 0 and VSEN input
Loop 1	Control loop 1 associated with PMBus PAGE 1 and BVSEN input
Np	Number of primary turns
Ns	Number of secondary turns
Phase 1	First interleave phase associated with ISEN input
Phase 2	Second interleave phase associated with BISEN input
PID0	Compensator for Loop 0
PIN	Input power
POUT	Output power

Abbreviations or acronyms

Abbreviation	Expanded form
Q1…n	FET
RH	Upper sense resistor
Rishare	Current share summing resistor
RL	Lower sense resistor
RLoad	Load resistance
RPCB	PCB trace resistance
rW	Winding resistance
SR1…n	Synchronous rectification FET
Toff	Off-time
Ton	On-time
Tsw	Switching period
Vbody	Voltage across the body diode
Vcontrol	Internal reference voltage
Verrn	Error voltage
VIMON	Voltage at the IMON pin
VL	Inductor voltage
VIN	Input voltage
VOUT	Output voltage, Loop 0
VPRIMARY	Primary-side voltage
VRECT	Rectified voltage
VREF	XDPP1100 input pin for voltage sensing
VRREF	XDPP1100 input pin for voltage sensing
VRSEN	XDPP1100 input pin for voltage sensing
VSEN	XDPP1100 input pin for voltage sensing
VSEN	Sensed voltage

Revision history

Document revision	Date	Description of changes
V 1.0	2021-08-25	• Initial public release
V 1.1	2023-01-03	• Figure 47: updated signal resolutions • Figure 49: updated signal resolutions • Figure 50: updated signal resolutions, changed text "C" to "Clamp" • Equation 6.10: Changed exponent from -16 to -15 • Equation 6.14: Changed exponent from -26 to -25 • Equation 6.18: Changed exponent from -11 to -10 • Table 30: Update Kp_real entries to match Equation 6.10 exponent change • Table 31: Update Ki_real entries to match Equation 6.14 exponent change • Table 32: Update Kd_real entries to match Equation 6.18 exponent change • Table 61: updated tlm_vrect_rcorr description • Table 48: correct PWM_DEADTIME bit ranges in descriptions of fields pwm2_dr, pwm2_df, pwm12_dr and pwm12_df.
V 1.2	2024-02-22	• Page 1: Updated document title and “Scope and purpose” section to include XDPP1100A • Section 1: Updated introduction to include XDPP1100A • Figure 77: Added fault vdd_uvlo (XDPP1100A) to fault_loop_bus[24] • Section 9.1.10: Inserted description of VDD UVLO fault for XDPP1100A here, following 9.1.x section numbers incremented by 1 • Table 64: Added VDD_UVLO (XDPP1100A) fault • Table 66: Added VDD_UVLO (XDPP1100A) fault to fields fault_shut_mask_loop, fault_t2_shut_mask_loop, fault_enable_loop, fault_polarity_loop, fault_force_loop, fault_clear_loop, fault_status_loop, and fault_reg_loop • Table 67: Added VDD_UVLO (XDPP1100A) fault to commands STATUS_MFR_SPECIFIC, FW_CONFIG_FAULTS.Fault_pin_mask_hw, FW_CONFIG_FAULTS.Fault_enable_mask_loop_hw, and FW_CONFIG_FAULTS.Fault_t2_shut_mask_loop_hw • Section 10.1: Updated text concerning ishare_dead_zone and added equation (10.5) • Section 10.2: Renumbered previous equations (10.5) through (10.8) to (10.6) through (10.9) • Table 70: Updated ishare_dead_zone description
V 1.3	2026-06-12	• Section 1: removed the duplicated description • Section 1.3: moved the abbreviations to the end of the document • Section 2.3.1: updated Figure 9, corrected typos of the register format

Revision history

Document revision	Date	Description of changes
		• Table 2: vsp1_cnt_vrscomp_e, vsp1_cnt_vrscomp_o, vsp2_cnt_vrscomp_e, vsp2_cnt_vrscomp_o, register range is corrected to “0 to 10235 ns”. • Table 16: ce_pwmwin_dly bits column changed to [27:23] • Table 96: the description of RSTSR register SWRST and WDRST are swapped

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2026-06-24
Published by

Infineon Technologies AG
Am Campeon 1-15
85579 Neubiberg
Germany

© 2026 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?
Email: erratum@infineon.com

Document reference
UM193611

Important Notice
This evaluation board, demonstration board, reference board or kit and any related documentation (“Evaluation Board”) is sold or provided and delivered by Infineon Technologies AG and its affiliates (“Infineon”). Evaluation Boards are part of Infineon’s non-serial products.
This Evaluation Board is delivered to customer "AS IS", without any warranty (whether express or implied) or liability of Infineon of any kind, including but without limitation any warranty of merchantability, fitness for a particular purpose or non-infringement. These limitations and exclusions shall not apply in case of Infineon's willful misconduct or in any other case where liability is mandatory at law.
This Evaluation Board shall only be used by customer for test purposes and not for series production. However, the Evaluation Board shall not to be used for reliability testing. The Evaluation Board is exclusively intended for technically qualified and skilled customer representatives. The Evaluation Board shall not be used in any way where a failure of the Evaluation Board, or any consequences of the use thereof, can reasonably be expected to result in personal injury.
Evaluation Boards may not comply with CE or similar standards (including without limitation the EMC Directive 2004/EC/108 and the EMC Act) and may not fulfill other requirements of the country in which they are operated by the customer. The customer shall ensure that each Evaluation Board will be handled in a way which is compliant with all relevant requirements and standards in the country in which it is operated.
Software and associated documentation and materials included or referenced in the Evaluation Board (“Software”) is owned by Infineon and is protected by and subject to worldwide patent protection, worldwide copyright laws, and international treaty provisions. Therefore, you may use this Software only as provided in the license agreement accompanying the software package from which you obtained this Software.
If no license agreement applies, then any use, reproduction, modification, translation, or compilation of this Software is prohibited without the express written permission of Infineon. Unless otherwise expressly agreed with Infineon, this Software is provided to customer "AS IS", without any warranty (whether express or implied) or liability of Infineon of any kind, including but without limitation any warranty of merchantability, fitness for a particular purpose or non-infringement.
These limitations and exclusions shall not apply in case of Infineon's willful misconduct or in any other case where liability is mandatory at law. Infineon reserves the right to make changes to the Software without notice. You are responsible for properly designing, programming, and testing the functionality and safety of your intended application of the Software, as well as complying with any legal requirements related to its use. Infineon does not guarantee that the Software will be free from intrusion, data theft or loss, or other breaches (“Security Breaches”), and Infineon shall have no liability arising out of any Security Breaches. The Software shall not be used in any way where a failure of the Evaluation Board or any consequences of the use thereof, can reasonably be expected to result in personal injury.
Safety & Operating Instructions:
Customer shall check the Evaluation Board for any physical damage which may have occurred during transport. If customer detects any damages or defects in the Evaluation Board, customer shall not connect the Evaluation Board to a power source. Customer shall contact Infineon for further support. If customer observes unusual operating behavior during the evaluation process, customer shall immediately shut off the power supply to the Evaluation Board and consult Infineon for support.
Customer shall not touch the Evaluation Board during operation and keep a safe distance.
Customer shall not touch the Evaluation Board after disconnecting the power supply, several components may still store electrical voltage and can discharge through physical contact. Several parts, like heat sinks and transformers, may still be very hot. Allow the components to cool before touching or servicing.
The electrical installation must be completed in accordance with the appropriate safety requirements.