Datasheet

DA14535

Ultra-Low Power Bluetooth 5.3 SoC

The DA14535 is an ultra-low power SoC integrating a 2.4 GHz transceiver and an Arm® Cortex®-M0+ microcontroller with a RAM of 64 kB and a One-Time Programmable (OTP) memory of 12 kB. It can be used as a standalone application processor or as a data pump in hosted systems. Ultra-low power can be achieved using the integrated Low I_Q Buck/Boost DCDC which is on during SLEEP in Buck mode.

The radio transceiver, the baseband processor, and the qualified Bluetooth® low energy stack is fully compliant with the Bluetooth® Low Energy 5.3 standard.

The DA14535 has dedicated hardware for the Link Layer implementation of Bluetooth® LE and interface controllers for enhanced connectivity capabilities.

The Bluetooth® LE firmware includes the L2CAP service layer protocols, Security Manager (SM), Attribute Protocol (ATT), the Generic Attribute Profile (GATT), and the Generic Access Profile (GAP). All profiles published by the Bluetooth® SIG as well as custom profiles are supported.

The device is suitable for disposables, wireless sensor nodes, beacons, proximity tags and trackers, smart HID devices (stylus, keyboards, mice, and trackpads), toys, and medical and industrial applications.

Key Features

- Compatible with Bluetooth® v5.3, ETSI EN 300 328 and EN 300 440 Class 2 (Europe), FCC CFR47 Part 15 (US) and ARIB STD-T66 (Japan)
- Supports up to three Bluetooth® LE connections
- Typical cold boot to radioactive 35 ms
- Processing power
  - 16 MHz 32-bit Arm Cortex-M0+ with SWD interface
  - Dedicated Link Layer and AES-128 Encryption Processor
  - Software-based True Random Number Generator (TRNG)
- Memories
  - 12 kB One-Time-Programmable (OTP)
  - 64 kB Retainable System RAM
  - 160 kB ROM
- Power management
  - Integrated Low I_Q Buck/Boost DCDC converter
  - Buck: 1.8 V ≤ V_{BAT_HIGH} ≤ 3.6 V
  - Boost: 1.2 V ≤ V_{BAT_LOW} ≤ 1.65 V
  - Clock-less hibernation mode (buck mode, no RAM retention): 500 nA
  - Built-in temperature sensor for die temperature monitoring
  - Deep Sleep (no RAM retention, running on RCX, DCDC on): 1 µA
- Clocks
  - 32 MHz crystal and 32 MHz RC osc.
  - 32 kHz crystal and 32/512 kHz RC osc.
  - 15 kHz RCX as crystal replacement
- 2× General purpose Timers with capture and PWM capabilities
- Digital interfaces
  - 12 x GPIOs
  - 2× UARTs (one with flow control)
  - SPI Master/Slave up to 32 MHz (Master)
  - I2C bus at 100 kHz and 400 kHz
  - Three-axis capable Quadrature Decoder
  - Keyboard controller
- Analog interfaces
  - Four-channel 10-bit ADC
- Radio transceiver
  - Fully integrated 2.4 GHz CMOS transceiver
  - Single wire antenna
  - TX: 3.5 mA, RX: 2.1 mA (system currents with DC-DC, V_{BAT_HIGH} = 3 V and 0 dBm)
  - Programmable transmit output power from -19 dBm to +4 dBm
  - -94.5 dBm receiver sensitivity
- Operating temperature: -40 °C to 105 °C
- Package: FCGQFN 24 pins, 2.2 × 3, 0.4 mm pitch
- Programmable Reset Circuitry

Applications

- Medical applications
- Disposables
- Beacons
- Proximity tags and trackers
- Wireless sensor nodes
  - Fitness trackers
  - Consumer health
- Smartwatches
- Human interface devices (HID)
  - Stylus pens
  - Keyboards
  - Mouse devices
  - Trackpads
- Toys
- Industrial appliances

Key Benefits

- Lowest power consumption
- Smallest system size
- Lowest system cost

System Diagram

Figure 1. System diagram

Contents

Figures

Tables

1. Block Diagram

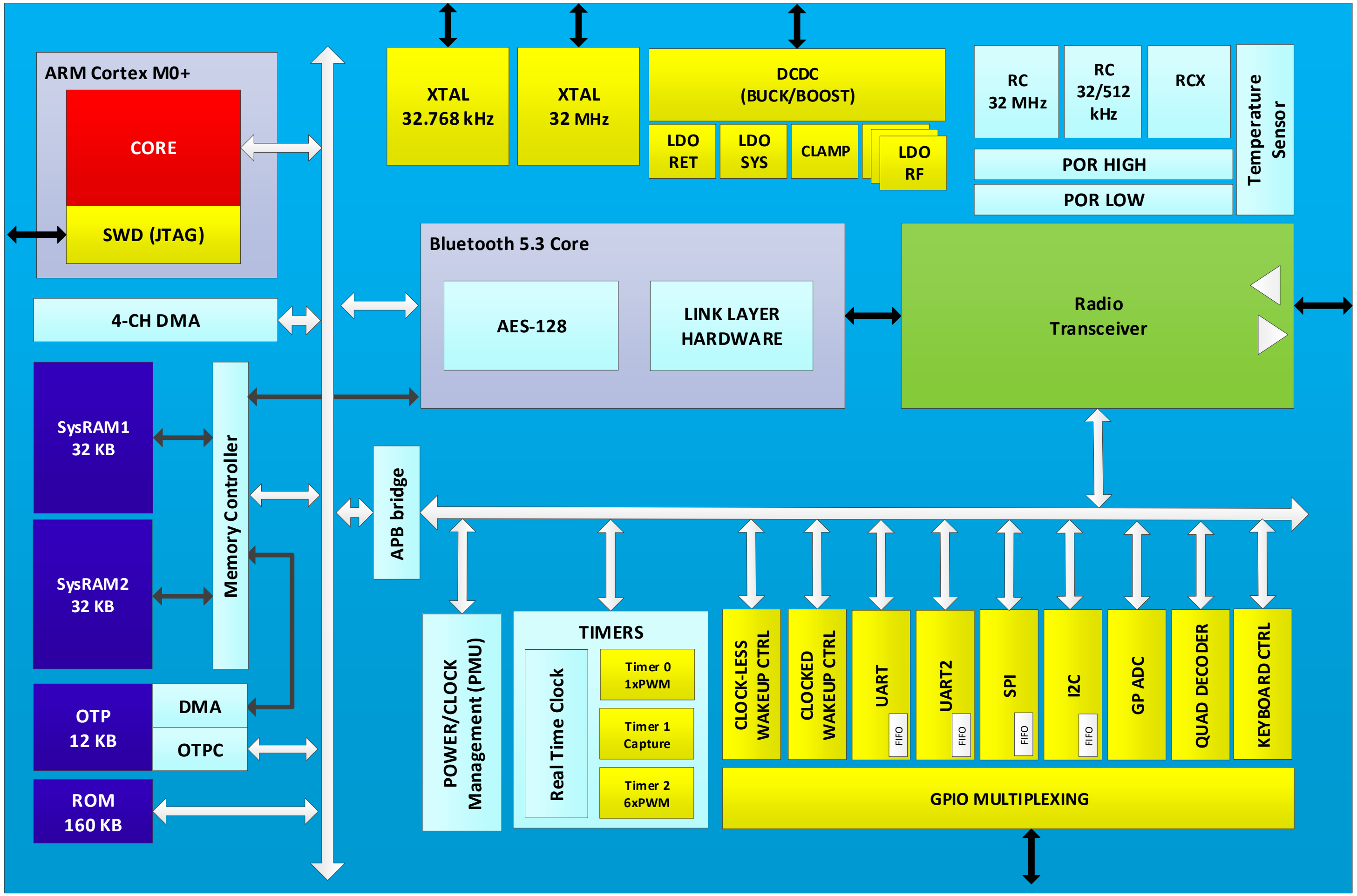

Figure 2. DA14535 block diagram

2. Package and Pinout

The DA14535 comes in one package:

- A Quad Flat No Leads Package (FCGQFN) with 24 pins

The actual pin/ball assignment is described in the following section.

2.1 FCGQFN24

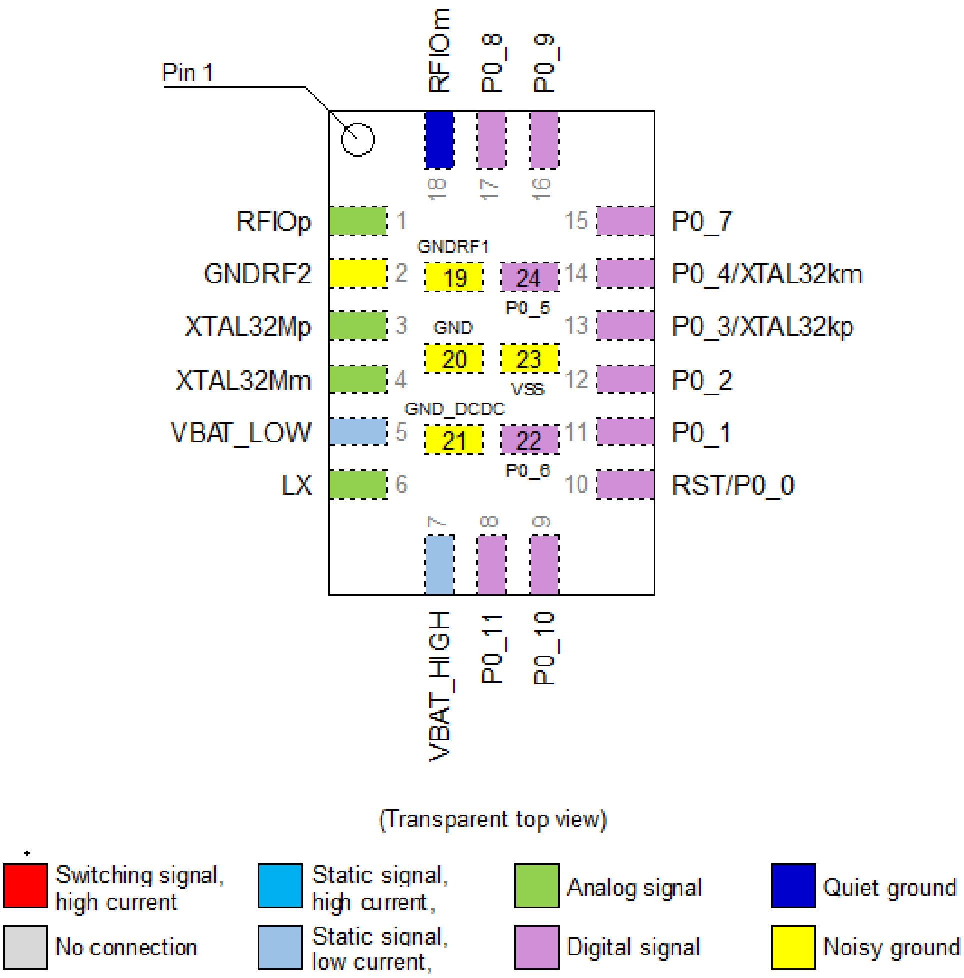

Figure 3. FCGQFN24 pin assignment (top view)

Table 1: DA14535 FCGQFN24 pin description

Pin no.	Pin name	Type	Reset state	Description
General purpose I/Os (Note 3)				
10	P0_0	DIO (Type B)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down.
	RST	DIO (Type B)		RST active high hardware reset (default).
11	P0_1	DIO (Type B)		INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or

Pin no.	Pin name	Type	Reset state	Description
			I-PD/ I-PU	alternate function nodes. Contains state retention mechanism during power down. Reset state: ▪ Buck mode: I-PD ▪ Boost mode: I-PU (Note 4)
	ADC0	AI		INPUT. Analog to Digital Converter input 0.
12	P0_2	DIO (Type B)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down.
	ADC1	AI		INPUT. Analog to Digital Converter input 1.
	SWCLK	DIO		INPUT JTAG clock signal (by default).
13	P0_3	DIO (Type B)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. Check GP_DATA_REG[P03_P04_FILT_DIS] for correct pad filter settings.
	XTAL32kp	AI		INPUT. Analog input of the XTAL32K crystal oscillator.
		DI		INPUT. Digital input for an external clock (square wave).
14	P0_4	DIO (Type B)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. Check GP_DATA_REG[P03_P04_FILT_DIS] for correct pad filter settings.
	XTAL32km	AO		OUTPUT. Analog output of the XTAL32K crystal oscillator.
24	P0_5	DIO (Type B)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down.
22	P0_6	DIO (Type A)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down.
	ADC2	AI		INPUT. Analog to Digital Converter input 2.
15	P0_7	DIO (Type A)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down.
	ADC3	AI		INPUT. Analog to Digital Converter input 3.
17	P0_8	DIO (Type A)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down.
16	P0_9	DIO (Type A)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or

Pin no.	Pin name	Type	Reset state	Description
				alternate function nodes. Contains state retention mechanism during power down.
9	P0_10	DIO (Type A)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during powerdown.
	SWDIO	DIO		INPUT/OUTPUT. JTAG Data input/output. Bidirectional data and control communication (by default).
8	P0_11	DIO (Type A)	I-PD	INPUT/OUTPUT with selectable pull-up/down resistors. Pull-down enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during powerdown.
Debug interface				
9	SWDIO	DIO	I-PD	INPUT/OUTPUT. JTAG Data input/output. Bidirectional data and control communication. Mapped on P0_10 (by default).
12	SWCLK	DIO	I-PD	INPUT JTAG clock signal. Mapped on P0_2 (by default).
Clocks				
3	XTAL32Mp	AI		INPUT. Crystal input for 32 MHz XTAL.
4	XTAL32Mm	AO		OUTPUT. Crystal output for 32 MHz XTAL.
13	XTAL32kp	AI		INPUT. Crystal input for 32.768 kHz XTAL. Mapped on P0_3.
14	XTAL32km	AO		OUTPUT. Crystal output for 32.768 kHz XTAL. Mapped on P0_4.
Quadrature decoder				
	QD_CHA_X	DI		INPUT. Channel A for the X axis. Mapped on Px ports.
	QD_CHB_X	DI		INPUT. Channel B for the X axis. Mapped on Px ports.
	QD_CHA_Y	DI		INPUT. Channel A for the Y axis. Mapped on Px ports.
	QD_CHB_Y	DI		INPUT. Channel B for the Y axis. Mapped on Px ports.
	QD_CHA_Z	DI		INPUT. Channel A for the Z axis. Mapped on Px ports.
	QD_CHB_Z	DI		INPUT. Channel B for the Z axis. Mapped on Px ports.
SPI Bus interface				
	SPI_CLK	DO		INPUT/OUTPUT. SPI Clock. Mapped on Px ports.
	SPI_DI	DI		INPUT. SPI Data input. Mapped on Px ports (Note 1).
	SPI_DO	DO		OUTPUT. SPI Data output. Mapped on Px ports (Note 2).
	SPI_EN	DI/DO		INPUT/OUTPUT. SPI Clock enable. Mapped on Px ports.
I2C Bus interface				
	SDA	DIO/DIOD		INPUT/OUTPUT. I2C bus Data with open drain port. Mapped on Px ports. The mapped Px pin is automatically configured with a pull-up resistor (25 kΩ) when pin x is mapped to the I2C_SDA PID function.
	SCL	DIO/DIOD		INPUT/OUTPUT. I2C bus Clock with open drain port. In open drain mode, SCL is monitored to support bit stretching by a slave. Mapped on Px ports. The mapped Px pin is automatically

Pin no.	Pin name	Type	Reset state	Description
				configured with a pull-up resistor (25 kΩ) when pin x is mapped to the I2C_SCL PID function.
UART Interface				
	UTX	DO		OUTPUT. UART transmit data. Mapped on Px ports.
	URX	DI		INPUT. UART receive data. Mapped on Px ports.
	URTS	DO		OUTPUT. UART Request to Send. Mapped on Px ports.
	UCTS	DI		INPUT. UART Clear to Send. Mapped on Px ports.
	UTX2	DO		OUTPUT. UART2 transmit data. Mapped on Px ports.
	URX2	DI		INPUT. UART2 receive data. Mapped on Px ports.
ADC IO channels				
11	ADC0	AI		INPUT. Analog to Digital Converter input 0. Mapped on P0_1.
12	ADC1	AI		INPUT. Analog to Digital Converter input 1. Mapped on P0_2.
22	ADC2	AI		INPUT. Analog to Digital Converter input 2. Mapped on P0_6.
15	ADC3	AI		INPUT. Analog to Digital Converter input 3. Mapped on P0_7.
Radio transceiver				
1	RFIOp	AIO		RF input/output. Impedance 50 Ω.
18	RFIOm	AIO		RF ground.
19	GND_RF1	AIO		RF ground.
2	GND_RF2	AIO		RF ground.
Miscellaneous				
10	RST	DIO		INPUT. Reset signal (active high). Mapped on P0_0 (by default).
6	LX	AIO		INPUT/OUTPUT. Connection for the external DC-DC converter inductor.
Power and ground				
23	VSS	AIO		Digital ground.
20	GND	AIO		Analog ground.
7	V_{BAT_HIGH}	AIO		INPUT/OUTPUT. Battery connection or DCDC output in BUCK/BOOST mode, respectively. IO supply.
5	V_{BAT_LOW}	AIO		INPUT/OUTPUT. Battery connection or DCDC output in BOOST/BUCK mode, respectively. System supply.
21	GND_DCDC	AIO		DCDC ground.

Note 1 Data input only. MOSI in SPI slave mode and MISO in SPI master mode.

Note 2 Data output only. MISO in SPI slave mode and MOSI in SPI master mode.

Note 3 The differences between Type A and Type B GPIO pads are presented in Types of GPIO Pads.

Note 4 P0_1 is recommended to be used as chip select for the external flash in boost configuration.

3. Specifications

All MIN/MAX specification limits are guaranteed by design, production testing, and/or statistical characterization. Typical values are based on characterization results at default measurement conditions and are informative only.

Default measurement conditions (unless otherwise specified): V_{BAT_HIGH} = 3.0 V (buck mode), V_{BAT_LOW} = 1.5 V (boost mode), T_A = 25 ºC. MIN and MAX values are based on characterization results over the temperature range, unless otherwise specified. All radio measurements are performed with standard RF measurement equipment providing a source/load impedance of 50 Ω. All listed currents involving any radio operation have been conducted without the external CLC filter.

Due to the voltage dependent capacitance of MLCC capacitors the specified capacitor values at V_{BAT_HIGH} and V_{BAT_LOW} are effective capacitances.

3.1 Absolute Maximum Ratings

Table 2: Absolute maximum ratings

Parameter	Description	Conditions	Min	Max	Unit
$V_{BAT\_LIM\_LOW}$	Limiting supply voltage	Battery voltage in boost configuration, supply voltage in buck configuration	-0.2	3.6	V
$V_{BAT\_LIM\_HIGH}$	Limiting supply voltage	Battery voltage in buck configuration, supply voltage in boost configuration	-0.2	3.6	V
$V_{PIN\_LIM\_default}$	Limiting voltage on a pin		-0.2	3.6	V
V_{ESD_HBM}	Electrostatic discharge voltage (Human Body Model)			2.5	kV
V_{ESD_CDM}	Electrostatic discharge voltage (Charged Device Model)	RFIOp pin only up to 300 V		500	V
T_{STG}	Storage temperature		-50	150	°C

3.2 Recommended Operating Conditions

Table 3: Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
$I_{L\_VBAT\_HIGH\_BOOTING}$	Maximum DC load current on V_{BAT_HIGH} rail during booting in boost mode	Boost mode booting sequence active			0.85	mA
V_{BAT_HIGH}	Supply voltage. Buck/Bypass: battery voltage Boost: DCDC output		1.8	3	3.6	V
V_{BAT_LOW}	Supply voltage. Boost/Bypass: battery voltage Buck: DCDC or LDO_LOW output	Note: this is full range specification, for proper DCDC boost operation refer to V_{BAT_BOOST}	1.2	1.5	3.3	V
$V_{BAT\_HIGH\_OTP\_Program}$	Voltage range for OTP programming	Required temperature for programming is between -20 °C and 85 °C	2.25	2.5	3.6	V
$V_{BAT\_HIGH\_OTP\_Read}$	Voltage range for OTP reading		1.66	1.8	3.6	V
$V_{PIN\_default}$	Voltage on a pin		-0.2		VBAT_HIGH+0.2	V
T_A	Ambient temperature		-40	25	105	°C

3.3 DC Characteristics

Table 4: DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
$I_{BAT\_HIGH\_HIBERN\_0kB}$	Battery supply current	Hibernation mode, no RAM retained, no oscillator running; LDO_RET_TRIM = 0xE Note 1		0.5		µA
$I_{BAT\_HIGH\_HIBERN\_32kB}$	Battery supply current	Hibernation mode, 32 kB RAM retained, no oscillator running; LDO_RET_TRIM = 0xC		0.9		µA
$I_{BAT\_HIGH\_HIBERN\_64kB}$	Battery supply current	Hibernation mode, 64 kB RAM retained, no oscillator running; LDO_RET_TRIM = 0xC		1.5		µA
$I_{BAT\_LOW\_HIBERN\_0kB}$	Battery supply current	Hibernation mode, no RAM retained, no oscillator running; LDO_RET_TRIM = 0xE Note 2		0.5		µA
$I_{BAT\_LOW\_HIBERN\_32kB}$	Battery supply current	Hibernation mode, 32 kB RAM retained, no oscillator running; LDO_RET_TRIM = 0xC		0.9		µA
$I_{BAT\_LOW\_HIBERN\_64kB}$	Battery supply current	Hibernation mode, 64 kB RAM retained, no oscillator running; LDO_RET_TRIM = 0xC		1.4		µA
$I_{BAT\_LOW\_DP\_SLP\_0kB}$	Battery supply current	Deep-sleep with 0 kB RAM retained, running on RCX; LDO_RET_TRIM = 0xF		1.5		µA
$I_{BAT\_LOW\_EX\_SLP\_32kB}$	Battery supply current	Extended-sleep with 32 kB RAM retained, running on RCX; LDO_RET_TRIM = 0xF		2.1		µA
$I_{BAT\_LOW\_EX\_SLP\_64kB}$	Battery supply current	Extended-sleep with 64 kB RAM retained, running on RCX; LDO_RET_TRIM = 0xF		2.7		µA
$I_{BAT\_HIGH\_DP\_SLP\_0kB}$	Battery supply current	Deep-sleep with no RAM retained, running on RCX, DCDC On; LDO_RET_TRIM = 0xF		1		µA
$I_{BAT\_HIGH\_EX\_SLP\_32kB}$	Battery supply current	Extended-sleep mode with 32 kB RAM retained, running on RCX, DCDC On; LDO_RET_TRIM = 0xF		1.4		µA
$I_{BAT\_HIGH\_EX\_SLP\_64kB}$	Battery supply current	Extended-sleep mode with 64 kB RAM retained, running on RCX, DCDC On; LDO_RET_TRIM = 0xF		1.7		µA
$I_{BAT\_LOW\_ACT\_RX}$	Battery supply current	Application with Receiver Active, CPU idle at 16 MHz, DCDC off		5.6		mA

Parameter	Description	Conditions	Min	Typ	Max	Unit
$I_{BAT\_LOW\_ACT\_TX}$	Battery supply current	Application with Pout = 0 dBm, Transmit continuous unmodulated output power, CPU idle at 16 MHz, DCDC off		6.8		mA
$I_{BAT\_HIGH\_ACT\_RX}$	Battery supply current	Application with Receiver Active, CPU idle at 16 MHz, DCDC on		2.5		mA
$I_{BAT\_HIGH\_ACT\_TX}$	Battery supply current	Application with Pout = 0 dBm, Transmit continuous unmodulated output power, CPU idle at 16 MHz, DCDC on		3.1		mA
$I_{BAT\_LOW\_RUN\_16MHz}$	Battery supply current	CPU executing code from RAM running on XTAL32MHz oscillator at 16 MHz, DCDC off		940		µA
$I_{BAT\_HIGH\_RUN\_16MHz}$	Battery supply current	CPU executing code from RAM running on XTAL32MHz oscillator at 16 MHz		470		µA
$I_{BAT\_HIGH\_IDLE}$	Battery supply current	CPU in Wait-for-Interrupt (WFI) state running on XTAL32MHz oscillator at 16 MHz		220		µA
$I_{BAT\_LOW\_IDLE}$	Battery supply current	CPU in Wait-for-Interrupt (WFI) state running on XTAL32MHz oscillator at 16 MHz, DCDC off		400		µA

Note 1 I_{BAT_HIGH} is Buck configuration at VBAT = 3 V, at 25 °C.

Note 2 I_{BAT_LOW} is Boost configuration at VBAT = 1.5 V, at 25 °C.

3.4 Timing Characteristics

Table 5: Timing characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
t_{STA_HIBER}	Start-up time	Time from hibernation to the first executed code instruction. Applies to both buck or boost modes, excludes capacitors charging time. Assumes V_{BAT_LOW}=V_{BAT_HIGH} with use of the resistive switch so no charging of the respective rail is needed.		0.2	0.4	ms
$t_{STA\_SLP\_Buck}$	Start-up time	Time from GPIO toggle to the first executed code instruction. Sleep clock is RCX. Applicable for both Deep and Extended Sleep mode. Application in buck configuration at 3 V battery	0.86		1.2	ms

Parameter	Description	Conditions	Min	Typ	Max	Unit
		voltage, excluding capacitor charging.				
$t_{STA\_SLP\_Boost}$	Start-up time	Time from GPIO toggle to the first executed code instruction. Sleep clock is RCX. Applicable for both Deep and Etended Sleep mode. Application in boost configuration at 1.5 V battery voltage, excluding capacitor charging.	0.83		1.1	ms

3.5 DC-DC Converter

Table 6: DCDC - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
C_{OUT}	External capacitor	Effective value at DC output voltage	1			μF
L_{EXT}	External inductor		2	2.2	2.4	μH
V_{BAT_HIGH}	Operational supply voltage, applied to V_{BAT_HIGH}	POWER_LEVEL_REG[DCDC_LEVEL] = 0x0	1.8		3.6	V
V_{BAT_LOW}	Operational supply voltage, applied to V_{BAT_LOW}	POWER_LEVEL_REG[DCDC_LEVEL] = 0x1	1.2		1.65	V
$I_{L\_VBAT\_HIGH\_1V8}$	Load current on V_{BAT_HIGH} in BOOST mode	V_{OUT} = 1.8 V Note 1			20	mA
$I_{L\_VBAT\_HIGH\_2V5}$	Load current on V_{BAT_HIGH} in BOOST mode	V_{OUT} = 2.5 V Note 1			10	mA
$I_{L\_VBAT\_HIGH\_3V0}$	Load current on V_{BAT_HIGH} in BOOST mode	V_{OUT} = 3.0 V Note 1			10	mA

Note 1 $I_{L\_VBAT\_HIGH\_BOOTING}$ is applicable during booting.

Table 7: DCDC - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
$V_{O\_BUCK\_1V2}$	DC output voltage	DCDC_LEVEL = 0x0 DCDC_TRIM = 0x3 LEVEL1V1_BUMP = 0x1	1.175	1.2	1.225	V
$V_{O\_BOOST\_1V8}$	DC output voltage	DCDC_LEVEL = 0x1 DCDC_TRIM = 0x3	1.75	1.8	1.85	V
$V_{O\_BOOST\_2V5}$	DC output voltage	DCDC_LEVEL = 0x2 DCDC_TRIM = 0x3	2.425	2.5	2.575	V
$V_{O\_BOOST\_3V0}$	DC output voltage	DCDC_LEVEL = 0x3 DCDC_TRIM = 0x3	2.9	3	3.1	V

Parameter	Description	Conditions	Min	Typ	Max	Unit
η_{CONV_BUCK}	Conversion efficiency	V_{BAT} = 3.0 V V_{OUT} = 1.2 V I_{LOAD} = 10 mA	80	85	90	%
η_{CONV_BOOST}	Conversion efficiency	V_{BAT} = 1.5 V V_{OUT} = 1.8 V I_{LOAD} = 10 mA	85	90	95	%
V_{RPL_BUCK}	Ripple voltage	V_{BAT} = 3.0 V V_{OUT} = 1.2 V I_{LOAD} = 1 - 20 mA C_{EXT} = 1 µF	5	15	25	mV
V_{RPL_BOOST}	Ripple voltage	V_{BAT} = 1.5 V V_{OUT} = 1.8 V I_{LOAD} = 1 - 20 mA C_{EXT} = 1 µF	5	20	35	mV
$I_{Q\_VBAT\_HIGH}$	Quiescent current Buck mode	V_{BAT_HIGH} = 3.0 V No load on V_{BAT_LOW}			40	µA
$I_{Q\_VBAT\_LOW}$	Quiescent current BOOST mode	V_{BAT_LOW} = 1.5 V No load on V_{BAT_HIGH}			75	µA

3.6 Digital I/O Characteristics

Table 8: Digital Pad - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
I_{L_HIDRV}	Driving current - high mode			3.5		mA
I_{L_LODRV}	Driving current - low mode			0.35		mA
V_{IH}	HIGH level input voltage	V_{DD} = 0.9 V	0.7*VDD			V
V_{IL}	LOW level input voltage	V_{DD} = 0.9 V			0.3*VDD	V

Table 9: Digital Pad - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
I_{IH}	HIGH level input current	V_I = V_{BAT_HIGH} = 3.0 V	-10		10	µA
I_{IL}	LOW level input current	V_I = V_{SS} = 0 V	-10		10	µA
I_{IH_PD}	HIGH level input current	V_I = V_{BAT_HIGH} = 3.0 V	60		180	µA
I_{IL_PU}	LOW level input current	V_I = V_{SS} = 0 V, V_{BAT_HIGH} = 3.0 V	-180		-60	µA
V_{OH}	HIGH level output voltage	I_O = 3.5 mA, V_{BAT_HIGH} = 1.7 V	0.8*VBAT_HIGH			V

Parameter	Description	Conditions	Min	Typ	Max	Unit
V_{OL}	LOW level output voltage	I_O = 3.5 mA, V_{BAT_HIGH} = 1.7 V			0.2*VBAT_HIGH	V
V_{OH_LOWDRV}	HIGH level output voltage	I_O = 0.3 mA, V_{BAT_HIGH} = 1.7 V	0.8*VBAT_HIGH			V
V_{OL_LOWDRV}	LOW level output voltage	I_O = 0.3 mA, V_{BAT_HIGH} = 1.7 V			0.2*VBAT_HIGH	V
C_{IN}	Input capacitance			0.75		pF

Table 10: Digital Pad with LPF - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
I_{L_HIDRV}	Driving current - high mode			3.5		mA
I_{L_LODRV}	Driving current - low mode			0.35		mA

Table 11: Digital Pad with LPF - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
C_{IN}	Input capacitance	Note 1 Note 2		3.85		pF

Note 1 Digital pad characteristics are equal to the standard GPIO pads unless overruled or added in this table.

Note 2 P0_3 and P0_4 are type B pads with selectable filter via GP_DATA_REG[P03_P04_FILT_DIS], C_{IN} is equal to a Type A pad both with filter enabled or disabled.

3.7 GP ADC

Table 12: GPADC - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
N_{BIT_ADC}	Number of bits (resolution)			10		bit

Table 13: GPADC - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
E_G	ADC gain error without software correction (single-ended)	Trimmed bandgap	-3		5	%
E_{G_COR}	ADC gain error after software correction (single-ended)	Trimmed bandgap and Gain Error + Offset correction applied	-1		1.6	%
E_{OFS}	ADC offset error without software correction (single-ended)	Trimmed bandgap, no chopping	-40		40	LSB
E_{OFS_COR}	ADC offset error after software correction (single-ended)	Trimmed bandgap, chopping enabled and Gain Error + Offset correction applied	-4		7	LSB

Parameter	Description	Conditions	Min	Typ	Max	Unit
E_{G_DIF}	ADC gain error without software correction (differential)	Trimmed bandgap	-3		6.5	%
$E_{G\_DIF\_COR}$	ADC gain error after software correction (differential)	Trimmed bandgap and Gain Error + Offset correction applied	-1		1.6	%
E_{OFS_DIF}	ADC offset error without software correction (differential)	Trimmed bandgap, no chopping	-20		20	LSB
$E_{OFS\_DIF\_COR}$	ADC offset error after software correction (differential)	Trimmed bandgap, chopping enabled and Gain Error + Offset correction applied	-4		4	LSB
E_{G_ATTNX}	ADC gain error after software correction (including attenuator), excl ATTN2X SE	Trimmed bandgap and GPADC Gain Error + Offset correction applied	-4		4	%
E_{OFS_ATTNX}	ADC offset error after software correction (including attenuator), excl ATTN2X SE	Trimmed bandgap, chopping enabled and GPADC Gain Error + Offset correction applied	-16		16	LSB
E_{G_ATTN2x}	SE only, ADC gain error after software correction (including attenuator)	Trimmed bandgap and ATTN2X Gain Error + Offset correction applied	-1		1.5	%
E_{OFS_ATTN2x}	SE only, ADC offset error after software correction (including attenuator)	Trimmed bandgap and ATTN2X Gain Error + Offset correction applied	-4		4	LSB
INL	Integral non-linearity	Note 1	-2		2	LSB
DNL	Differential non-linearity		-2		2	LSB
ENOB	Effective Number of Bits	No averaging, no chopping, single-ended: $V_{IN,PP}$ = 800 mV		9		bit
$ENOB_{AVG128}$	Effective Number of Bits	128x averaging, single-ended: $V_{IN,PP}$ = 800 mV		10.3		bit

Note 1 INL is the deviation of a code from a straight line passing through the actual endpoints of the transfer curve.

Table 14: GPADC - Electrical performance

Parameter	Description	Conditions	Min	Typ	Max	Unit
t_{CONV_ADC}	Conversion time of the ADC			125	500	ns

3.8 LDO_LOW Characteristics

Table 15: LDO_LOW - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
C_{OUT}	Output capacitance		1		10	µF

Table 16: LDO_LOW - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
V_{O_ACTIVE}	Output voltage	Active mode (high current) POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x3 (default) I_{LOAD} = 1 mA	1.164	1.2	1.236	V
V_{O_SLEEP}	Output voltage	Sleep mode (low current) POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x3 (default) I_{LOAD} = 100 μA	1.152	1.2	1.248	V
$V_{O\_TRIM\_0}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x0		-75		mV
$V_{O\_TRIM\_1}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x1		-50		mV
$V_{O\_TRIM\_2}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x2		-25		mV
$V_{O\_TRIM\_3}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x3		0		mV
$V_{O\_TRIM\_4}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x4		25		mV
$V_{O\_TRIM\_5}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x5		50		mV
$V_{O\_TRIM\_6}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x6		75		mV
$V_{O\_TRIM\_7}$	Trim step, delta from 1.1 V nominal value	POWER_LEVEL_REG[LDO_LOW_TRIM] = 0x7		100		mV
I_{Q_ACTIVE}	Quiescent current	No load		12		μA
I_{Q_SLEEP}	Quiescent current	No load		40		nA
$R_{ON\_SW\_HIGH}$	On resistance of the high Ohmic switch between V_{BAT_HIGH} and V_{BAT_LOW} rail.	Buck mode, V_{BAT_HIGH} = 1.62 V, V_{BAT_LOW} = 1.2 V		250		Ω
$R_{ON\_SW\_LOW}$	On resistance of the low Ohmic switch between V_{BAT_HIGH} and V_{BAT_LOW} rail.	Buck mode, V_{BAT_HIGH} = 1.62 V, V_{BAT_LOW} = 1.2 V		10		Ω
$R_{ON\_SW\_LOW\_BOOST}$	On resistance of the low Ohmic switch between V_{BAT_HIGH} and V_{BAT_LOW} rail.	Boost mode, V_{BAT_LOW} = 1.2 V		10		Ω

3.9 Power on Reset

Table 17: POR VBAT_HIGH - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
V_{TH_H}	POR VBAT_HIGH reset release voltage			1.75	1.8	V

Parameter	Description	Conditions	Min	Typ	Max	Unit
V_{TH_L}	POR VBAT_HIGH reset activation voltage		1.57	1.66		V

Table 18: POR VBAT_LOW - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
V_{TH_H}	POR VBAT_LOW reset release voltage			1.15	1.2	V
V_{TH_L}	POR VBAT_LOW reset activation voltage		1.01	1.1		V

3.10 RC32M Oscillator

Table 19: RC32M - Timing characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
$f_{RC32M\_TRIMMED}$	RC oscillator frequency	At target trimming	28	30.5	32	MHz

3.11 RCX Oscillator

Table 20: RCX - Timing characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
Δf_{RC}	RCX oscillator frequency drift	100 ms time slot		100	500	ppm
$\Delta f_{RC}/\Delta V_{\_VBAT\_HIGH}$	Supply voltage dependency (V_{BAT_HIGH})		-500	80	1000	ppm/V
$\Delta f_{RC}/\Delta V_{\_VBAT\_LOW}$	Supply voltage dependency (V_{BAT_LOW})		-500	200	3000	ppm/V
f_{RCX}	RCX oscillator frequency	At target fixed trim setting	13	15	19	kHz
$\Delta f_{RC}/\Delta T_{\_1}$	Temperature dependency	Temperature range from -40 °C to 85 °C, RCX_BIAS at preferred value	-210		125	ppm/deg
$\Delta f_{RC}/\Delta T_{\_2}$	Temperature dependency	Temperature range from -40 °C to 105 °C, RCX_BIAS at preferred value	-210		200	ppm/deg

3.12 Temperature Sensor

Table 21: Temperature Sensor - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
T_{SENSE_RANGE}	Temperature sensor range		-40		105	°C
$T_{SENSE\_ACC\_OTP\_QFN}$	Applies to the QFN package. Absolute accuracy of temperature sensor using calibration value from	$T_{AMBIENT}$ = 25 °C, V_{BAT_LOW} = 1.1 V		±4		°C

Parameter	Description	Conditions	Min	Typ	Max	Unit
	OTP (single point calibration at 25 °C) . Formula: T_X = 25 °C + (ADC_X - $ADC_{OTP\_CAL\_25C}$) / (TC_{SENSE} * 64) (64 is used to correct 16b to 10b ADC values)					
TC_{SENSE}	Temperature coefficient of the internal temperature sensor	Reading through GP_ADC (16-bit result)		2.3		LSB/°C

3.13 XTAL32kHz Oscillator

Table 22: XTAL32K - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
f_{CLK_EXT}	External clock frequency	At pin 32KXTAL1/P0_3 in GPIO mode.	10		100	kHz
f_{XTAL}	Crystal oscillator frequency		30	32.768	35	kHz
ESR	Equivalent series resistance				100	kΩ
C_L	Load capacitance	No external capacitors are required for a 6 pF or 7 pF crystal.	6	7	9	pF
C_0	Shunt capacitance			1	2	pF
Δf_{XTAL}	Crystal frequency tolerance (including aging)	Timing accuracy is dominated by crystal accuracy. A much smaller value is preferred.	-250		250	ppm
P_{DRV_MAX}	Maximum drive power	Note 1	0.1			μW

Note 1 Select a crystal that can handle a drive level of at least this specification.

Table 23: XTAL32K - Timing characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
t_{STA_XTAL}	Crystal oscillator start-up time	Typical application, time until 1000 clocks are detected.		100	300	ms

3.14 Oscillator

Table 24: XTAL32M - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
P_{DRV_MAX}	Maximum drive power	Note 1			100	μW
V_{CLK_EXT}	External clock voltage	In case of external clock source on XTAL32Mp (XTAL32Mm floating or connected to mid-level 0.6 V)		0.9	1.2	V
$\varphi_{N\_EXT\_32M}$	Phase noise	f_C = 50 kHz; in case of external clock source			-130	dBc/Hz

Parameter	Description	Conditions	Min	Typ	Max	Unit
Δf_{XTAL_TRIM}	Crystal frequency trim			2		ppm
f_{XTAL_32M}	Crystal oscillator frequency			32		MHz
Δf_{XTAL}	Crystal frequency tolerance	After optional trimming; including aging and temperature drift Note 2	-20		20	ppm
Δf_{XTAL_UNT}	Crystal frequency tolerance	Untrimmed; including aging and temperature drift Note 3	-40		40	ppm
$ESR_{\_1pF}$	Equivalent series resistance	C_0 < 1 pF			200	Ω
$ESR_{\_3pF}$	Equivalent series resistance	C_0 < 3 pF			80	Ω
$ESR_{\_5pF}$	Equivalent series resistance	C_0 < 5 pF			50	Ω
C_L	Load capacitance	No external capacitors are required	4	6	8	pF

Note 1 Select a crystal which can handle a drive level of at least this specification.

Note 2 Using the internal varicaps a wide range of crystals can be trimmed to the required tolerance.

Note 3 Maximum allowed frequency tolerance for compensation by the internal varicap trimming mechanism.

3.15 Radio

Table 25: Radio1M - Recommended operating conditions

Parameter	Description	Conditions	Min	Typ	Max	Unit
f_{OPER}	Operating frequency		2400		2483.5	MHz
N_{CH}	Number of channels			40		1
f_{CH}	Channel frequency	K = 0 to 39		2402+K *2		MHz

Table 26: Radio1M - DC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
$I_{BAT\_RF\_RX}$	Current at V_{BAT_LOW} = 1.2 V	Radio receiver and synthesizer active; T_A = 25 °C		4.5		mA
$I_{BAT\_RF\_TX\_+4dBm}$	Current at V_{BAT_LOW} = 1.2 V	Radio receiver and synthesizer active, power setting = 12; T_A = 25 °C		9.5		mA
$I_{BAT\_RF\_TX\_0dBm}$	Current at V_{BAT_LOW} = 1.2 V	Radio receiver and synthesizer active, power setting = 8; T_A = 25 °C		7.5		mA
$I_{BAT\_RF\_TX\_-3dBm}$	Current at V_{BAT_LOW} = 1.2 V	Radio receiver and synthesizer active, power setting = 6; T_A = 25 °C		6.4		mA

Parameter	Description	Conditions	Min	Typ	Max	Unit
$I_{BAT\_RF\_TX\_-6dBm}$	Current at V_{BAT_LOW} = 1.2 V	Radio receiver and synthesizer active, power setting = 4; T_A = 25 °C		5.2		mA
$I_{BAT\_RF\_TX\_-12dBm}$	Current at V_{BAT_LOW} = 1.2 V	Radio receiver and synthesizer active, power setting = 2; T_A = 25 °C		3.7		mA
$I_{BAT\_RF\_TX\_-18dBm}$	Current at V_{BAT_LOW} = 1.2 V	Radio receiver and synthesizer active, power setting = 1; T_A = 25 °C		2.9		mA

Table 27: Radio1M - AC characteristics

Parameter	Description	Conditions	Min	Typ	Max	Unit
P_{SENS_CLEAN}	Sensitivity level	Dirty Transmitter disabled; DCDC converter disabled; PER = 30.8%. Note 1		-94.5		dBm
P_{SENS_EPKT}	Sensitivity level	Extended packet size (255 octets)		-91.5		dBm
P_{SENS}	Sensitivity level	Normal Operating Conditions; DCDC converter disabled; PER = 30.8%. Note 1		-94		dBm
P_{INT_IMD}	Intermodulation distortion interferer power level	Worst-case interferer level @ f_1, f_2 with $2*f_1 - f_2 = f_0$, $\|f_1 - f_2\|$ = n MHz and n = 3, 4, 5; P_{WANTED} = -64 dBm @ f_0; PER = 30.8%. Note 2		-23		dBm
CIR_0	Carrier to interferer ratio	n = 0; interferer @ $f_1 = f_0 + n*1$ MHz. Note 3		8		dB
CIR_1	Carrier to interferer ratio	n = ±1; interferer @ $f_1 = f_0 + n*1$ MHz. Note 3		-1		dB
CIR_{P2}	Carrier to interferer ratio	n = +2 (image frequency); interferer @ $f_1 = f_0 + n*1$ MHz. Note 3		-26.5		dB
CIR_{M2}	Carrier to interferer ratio	n = -2; interferer @ $f_1 = f_0 + n*1$ MHz. Note 3		-29		dB
CIR_{P3}	Carrier to interferer ratio	n = +3 (image frequency + 1 MHz); interferer @ $f_1 = f_0 + n*1$ MHz. Note 3		-37		dB

Parameter	Description	Conditions	Min	Typ	Max	Unit
CIR_{M3}	Carrier to interferer ratio	n = -3; interferer @ $f_1 = f_0$ + n*1 MHz. Note 3		-42		dB
CIR_4	Carrier to interferer ratio	|n| ≥ 4 (any other Bluetooth LE channel); interferer @ $f_1 = f_0$ + n*1 MHz. Note 3		-43		dB
P_{BL_I}	Blocker power level	30 MHz ≤ f_{BL} ≤ 2000 MHz; P_{WANTED} = -67 dBm. Note 2		5		dBm
P_{BL_II}	Blocker power level Note 4	2003 MHz ≤ f_{BL} ≤ 2399 MHz; P_{WANTED} = -67 dBm. Note 2		0		dBm
P_{BL_III}	Blocker power level	2484 MHz ≤ f_{BL} ≤ 2997 MHz; P_{WANTED} = -67 dBm. Note 2		0		dBm
P_{BL_IV}	Blocker power level	3000 MHz ≤ f_{BL} ≤ 12.75 GHz; P_{WANTED} = -67 dBm. Note 2		5		dBm
L_{ACC_RSSI}	Level accuracy	Tolerance at 5% to 95% confidence interval of P_{RF}: when RXRSSI[7:0] = *X*, 50 < *X* < 175; Burst mode, 1500 packets.		3		dB
L_{RES_RSSI}	Level resolution	Gradient of monotonous range for RXRSSI[7:0] = *X*, 50 < *X* < 175; Burst mode, 1500 packets.		0.5		dB/LSB
ACP_{2M}	Adjacent channel power level	f_{OFS} = 2 MHz. Note 5		-45		dBm
ACP_{3M}	Adjacent channel power level	f_{OFS} ≥ 3 MHz. Note 5		-52		dBm
P_{O_12}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 12		4		dBm
P_{O_11}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 11		3		dBm
P_{O_10}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 10		2		dBm
P_{O_09}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 9		1		dBm
P_{O_08}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 8		0		dBm
P_{O_07}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 7		-1		dBm

Parameter	Description	Conditions	Min	Typ	Max	Unit
P_{O_06}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 6		-2.5		dBm
P_{O_05}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 5		-4		dBm
P_{O_04}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 4		-6		dBm
P_{O_03}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 3		-9		dBm
P_{O_02}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 2		-12		dBm
P_{O_01}	Output power level	RF_ATTR_REG[PA_POWER_SETTING] = 1		-19		dBm

Note 1 Measured according to Bluetooth® Low Energy Test Specification RF-PHY.TS/4.0.1, section 6.4.1.

Note 2 Measured according to Bluetooth® Core Technical Specification document.

Note 3 Measured according to Bluetooth® Core Technical Specification document, version 4.0, volume 6, section 4.2.

Note 4 Frequencies close to the ISM band can show slightly worse performance.

Note 5 Measured according to Bluetooth® Low Energy Test Specification RF-PHY.TS/4.0.1, section 6.2.3.

4. System Overview

4.1 Internal Blocks

The DA14535 contains the following blocks:

Arm® Cortex®-M0+ CPU with Wake-up Interrupt Controller (WIC). This processor provides 0.9 dMIPS/MHz and is used for assisting the Bluetooth® LE protocol implementation, for providing processing power for calculations or data fetches required by applications, and for housekeeping, including controlling the power scheme of the system.

BLE core. This is the baseband hardware accelerator for the Bluetooth® LE protocol.

ROM. This 160-kB ROM contains the Bluetooth® LE protocol stack and the boot code sequence.

OTP. This 12-kB OTP memory array is used to store a secondary bootloader, application code, and Bluetooth® LE profiles. It also contains the system configuration and calibration data.

System SRAM (SysRAM). This 64-kB SysRAM is primarily used to mirror the program code from the OTP or external SPI Flash when the system wakes/powers up. It also serves as a Data RAM for intermediate variables and various data that the protocol requires. It can be used as an extra memory space for the Bluetooth® LE TX and RX data structures (Exchange RAM). The SysRAM cells can not only be retained during sleep modes but also be completely switched off during active mode if not needed.

UART and UART2. The serial interface of the UART implements hardware flow control while UART2 does not. Both UARTs feature FIFOs with depths of 16 bytes each.

SPI. This is the serial peripheral interface (SPI) with master/slave capability, and it has separate FIFOs for RX and TX of two 16-bit words each.

I2C. This Master/Slave I2C interface is used for sensors and/or host Micro-Controllers Units (MCUs) communication. It comprises a 32-place 9-bit deep FIFO.

General Purpose (GP) ADC. This 10-bit analog-to-digital converter (ADC) has four external input channels (GPIOs) and internal channels for reading die temperature, the battery voltage, and other internal analog nodes.

Radio Transceiver. This block implements the RF part of the Bluetooth® LE protocol.

Clock Generator. This block is responsible for clocking the system. It contains two XTAL oscillators, one running at 32 MHz (XTAL32M) and used for the active mode of the system and the other running at 32.768 kHz (XTAL32K) and used for the sleep modes of the system. There are also three RC oscillators available: a 32 MHz oscillator (RC32M) with low precision (> 500 ppm), a 32 kHz oscillator (RC32K) with low precision (> 500 ppm), and a ~15 kHz oscillator (RCX) with high precision (< 500 ppm). The RCX oscillator can be used as a sleep clock to replace the XTAL32K oscillator to further improve the power dissipation of the system while reducing the bill of materials. The RC32M oscillator is used to provide a clock to mirror the OTP code into the SysRAM while the XTAL32M oscillator is settling directly after power/wake up. This clock is also used to run the Booter at powerup. An external digital clock can be used as a sleep clock to replace the XTAL32K or the RCX oscillator.

Timers. This block contains three timers:

- A 16-bit general purpose timer (Timer0) with two pulse width modulation (PWM) signals (PWM1 is inverted to PWM0).
- A 11-bit timer (Timer1) with two capture channels.
- A 14-bit timer (Timer2) which controls six PWM signals that all have the same frequency, but each has a configurable duty cycle.

Real Time Clock (RTC). This hardware controller supports the complete time of day clock: 12/24 hours, minutes, seconds, milliseconds, and hundredths of a millisecond. It includes a configurable alarm function and can be programmed to generate an interrupt on any event, like a rollover of month, day, hour, minute, second, or hundredths of a millisecond.

Wake-Up Timer. This timer captures external events, and it can be used on any of the GPIO ports as a wake-up trigger based on a programmable number of external events.

Quadrature Decoder. This block decodes the pulse trains from a rotary encoder to provide the step and the direction of a movement of an external device. Three axes (X, Y, and Z) are supported. The block also supports an edge counting mode which enables counting positive or negative edges on the selected GPIOs.

Keyboard Controller. This circuit enables the reading and debouncing of a programmable number of GPIOs and generates an interrupt upon a configurable action.

AHB/APB Bus. This block implements the AMBA Lite version of the AHB and APB specifications.

Power Management. This sophisticated power management circuit is equipped with a Buck/Boost DCDC converter and several low-dropout regulators (LDOs) that can be turned on/off through software.

A more detailed description of each component of DA14535 is presented in the following sections.

4.2 Power Management Unit

4.2.1 Introduction

The DA14535 has an integrated power management unit (PMU) which comprises a VDD_Clamp, a POR circuitry, a DCDC convertor, and various LDOs. Figure 4 shows the system diagram of the integrated PMU.

Features

- Boost, Buck, and DCDC bypass configurations
- Single inductance DCDC converter configured for Boost or Buck configuration
- Programmable DCDC converter outputs
- Active and sleep mode LDOs
- Low BOM and use of small external components.

4.2.2 Architecture

The PMU integrates two externally decoupled power rails: V_{BAT_HIGH} and V_{BAT_LOW}, and one internal V_{DD} power rail. There are three main power configurations: Buck, Boost, and Bypass. The integrated PMU configures itself automatically to the appropriate mode depending on how the battery is initially connected in the application.

- V_{BAT_HIGH}: voltage range from 1.8 V to 3.6 V. This power rail is used for the blocks which require a higher supply voltage. The OTP and the GPIOs are connected to this power rail. V_{BAT_HIGH} is protected by the POR circuit POR_HIGH that generates a POR when the voltage drops below the threshold voltage.
- V_{BAT_LOW}: the main system supply and most internal blocks are powered from this rail. Its functional range is from 1.2 V to 3.3 V. When used in Boost configuration, its default voltage range is from 1.2 V to 1.65 V, and within this range the DCDC converter can provide a supply at V_{BAT_HIGH} in the range from 1.8 V to 3.0 V (see Section 4.2.3.2). V_{BAT_LOW} is protected with the POR circuit POR_LOW which generates a hardware reset when the voltage drops below the threshold voltage.
- The internal V_{DD} power rail supplies the digital power domains (see Section 4.2.2.1 for the details).

The VDD_Clamp and RC32k blocks are supplied by the highest of V_{BAT_HIGH} and V_{BAT_LOW}.

In Buck configuration (Figure 4), the battery is connected to V_{BAT_HIGH}. The voltage on V_{BAT_LOW} is generated from V_{BAT_HIGH}. In Boost configuration (Figure 5), the battery is connected to V_{BAT_LOW}. The voltage on V_{BAT_HIGH} is generated from V_{BAT_LOW}. The different power modes of the system are explained in Section 4.2.2.2.

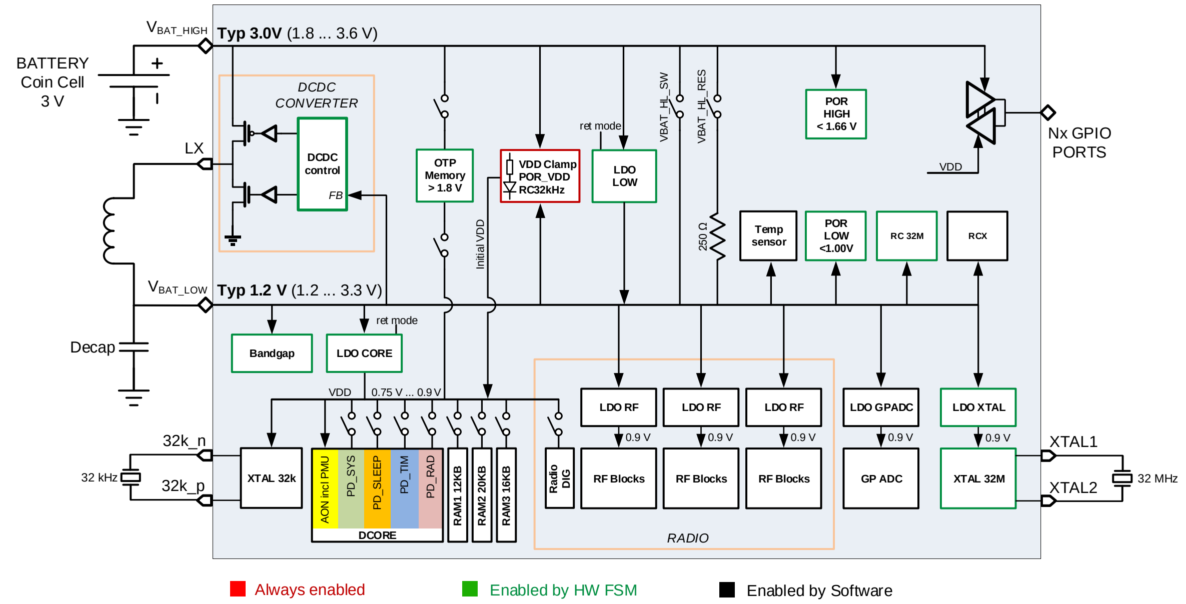

Figure 4. Power management unit: buck configuration

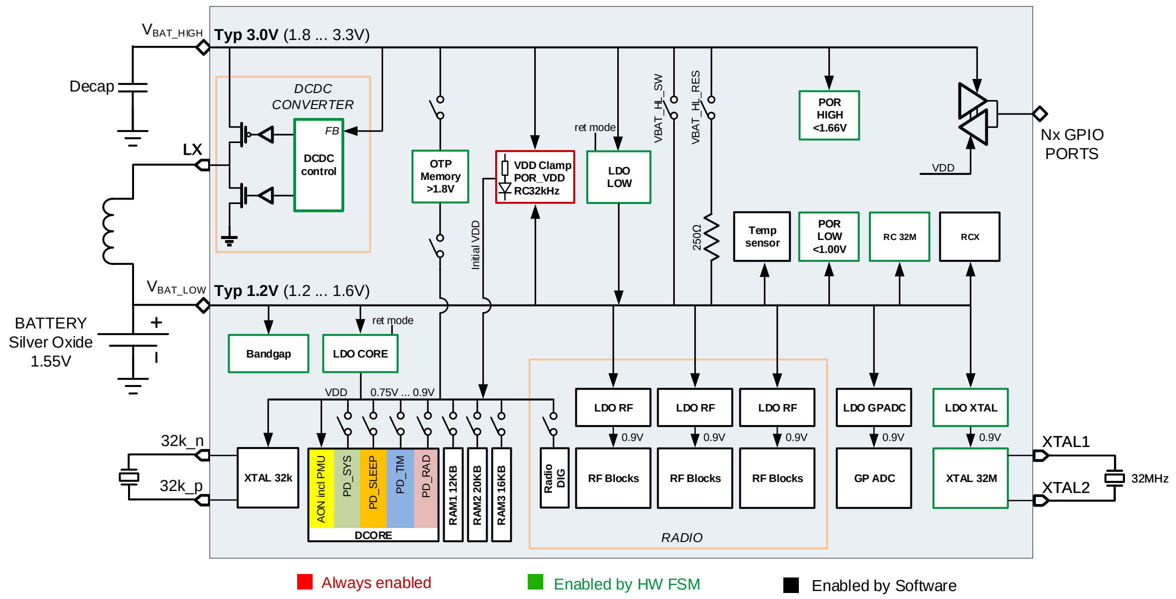

Figure 5. Power management unit: boost configuration

In bypass configuration (Figure 6), the DCDC is bypassed, and the battery is connected to both V_{BAT_LOW} and V_{BAT_HIGH}. In this mode an external inductor is omitted resulting in lower BOM. GPIOs supply follows the battery voltage in this configuration. The minimum cold boot voltage is 1.8 V. After cold boot and providing that no OTP or GPIO (at a specific voltage level) is needed, POR HIGH can be disabled and then V_{BAT_BYPASS} can go down to 1.2 V.

Note that because VBAT_LOW is limited to max 3.3 V, battery/supply voltage shall not exceed that level in the bypass configuration. If an application requires supply levels up to 3.6 V and the choice is made to omit the DCDC converter (reduced BOM at the cost of higher current consumption), it is recommended to use the "Buck Configuration" and enable LDO_LOW to power the VBAT_LOW rail (instead of via DCDC converter).

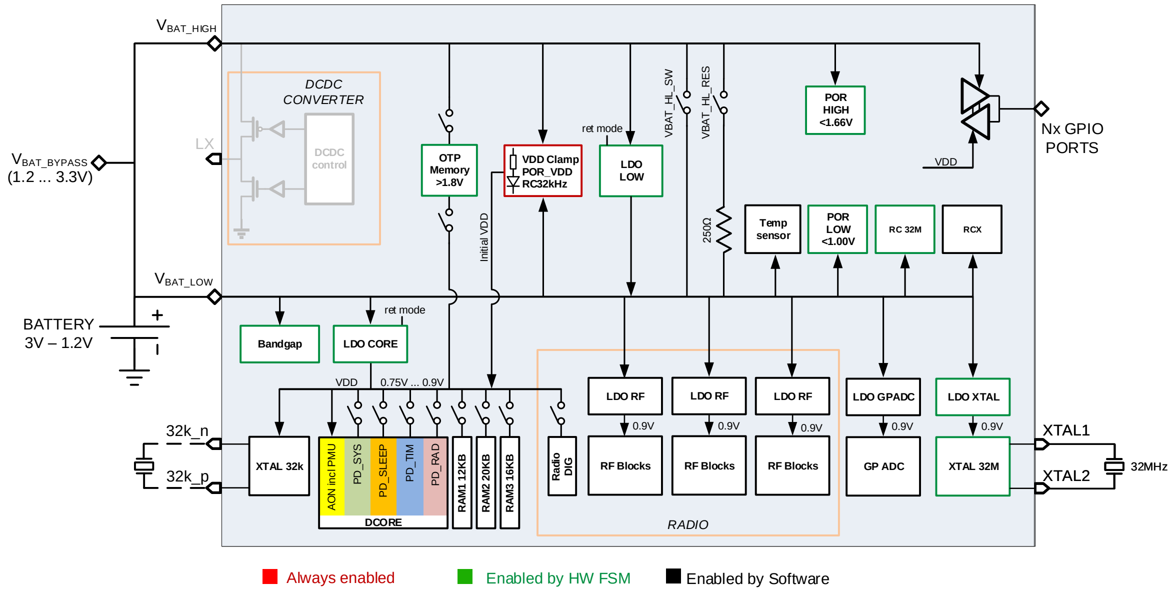

Figure 6. Power management unit: bypass configuration

4.2.2.1 Digital Power Domains

The DA14535 supports a number of digital power domains that can be turned on and off by software (Figure 7). Some of the blocks contain registers that can retain their values even if the digital power domain where they reside is powered off. RAM cells can retain their contents independently from the digital power domains state.

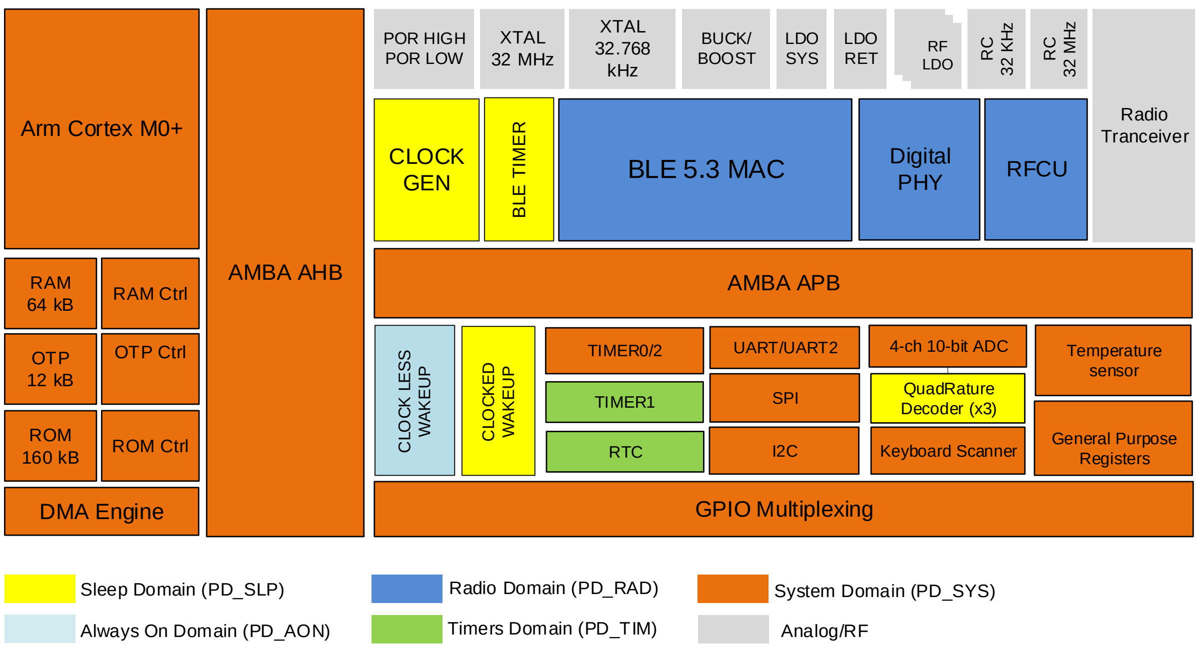

Figure 7. Digital power domains

Table 28 shows the list of blocks residing in each one of the digital power domains.

Table 28: Power domains description

Domain name	Description
PD_AON	Always powered domain. It contains a Clock-less Wake Up controller and the pad-ring.
PD_SLP	Sleep power domain. It comprises the Arm/WIC, the Bluetooth® LE Timer, the PMU/Clock Generation, the Clocked Wake-up Controller, the Quadrature Decoder, and various registers required for the Wake-up sequence.

Domain name	Description
PD_SYS	System Power Domain. It comprises the AHB bus, the OTP cell and controllers, the ROM, the System RAM, the Watchdog, the Software Timer, and the GPIO port multiplexing.
PD_TIM	Timer Power Domain. It comprises the RTC and the Timer1. These two blocks can be active during sleep modes.
PD_RAD	Radio Power Domain. It comprises the Bluetooth® LE Core and the digital PHY of the Radio.

4.2.2.2 Power Modes

There are five different power modes in DA14535:

- **Active mode:** System is active and operates at full speed.
- **Idle mode:** No power gating has been programmed. The Cortex CPU is idle, waiting for an interrupt. PD_SYS is on. PD_TIM and PD_RAD depend on the programmed enabled value.
- **Extended Sleep mode:** PD_AON, PD_SLP, and conditionally PD_TIM are active. RAM is expected to be retained for:
  - Keeping a Bluetooth® LE connection alive (stack variables or Bluetooth® LE data).
  - Potentially keep the application code and it can be omitted if the OTP or external Flash is instructed to automatically get mirrored into RAM upon every wake up.
- **Deep Sleep mode**: PD_AON and partially PD_SLP (CLOCKGEN and CLOCKED WAKEUP) are active. Device wakes up in reset state. RAM retainability and RTC operation are programmable.
- **Hibernation mode**: Shipping clock-less mode with all domains disabled. RAM retainability is programmable. No clock is running.

A summary of the power modes, the digital power domains, as well as the clocks and wake-up capabilities are explained in Table 29.

Table 29: Power modes, digital power domains, clocks, and wake-up triggers

Power mode	Digital power domains	LDOs, DCDC converter, and VDD level	Clock availability	RAM	Wake-up from
Active or Sleep (WFI)	PD_AON = ON PD_SLP = ON PD_SYS = ON PD_TIM = OPTIONAL PD_RAD = OPT	VDD = 0.9 V DCDC = ON (Buck or Boost) LDO_LOW = OFF LDO_CORE = ON, Active (0.9 V) VDD_Clamp = OFF LDO_RADIO = Programmable	All	SysRAM1 = ON (Application) SysRAM2 = optionally retained SysRAM3 = ON (Stack Data)	
Extended Sleep (with or without OTP)	PD_AON = ON PD_SLP = ON PD_SYS = OFF PD_TIM = OPTIONAL PD_RAD = OFF	VDD = 0.75 V DCDC = ON in Buck, OPTIONAL in Boost. LDO_LOW = OFF LDO_CORE = ON, in retain mode (0.75 V) VDD_Clamp = OFF LDO_RADIO = OFF	RCX or XTAL32K	SysRAMx = optionally retained (typically only SysRAM1 is retained)	▪ from any GPIOs ▪ RTC alarm ▪ Timer1 ▪ Bluetooth® LE sleep timer
Deep Sleep	PD_AON = ON PD_SLP = ON PD_SYS = OFF PD_TIM = OPTIONAL PD_RAD = OFF	VDD = 0.75 V DCDC = ON in Buck, OPTIONAL in Boost. LDO_LOW = OFF LDO_CORE = ON, in retain mode (0.75 V) VDD_Clamp = OFF LDO_RADIO = OFF	RCX or XTAL32K	SysRAMx = optionally retained (Typically OFF)	▪ from any GPIOs ▪ RTC alarm ▪ Timer1
Hibernation	PD_AON = ON PD_SLP = OFF PD_SYS = OFF PD_TIM = OFF PD_RAD = OFF	VDD = ~0.75 V DCDC = OFF LDO_LOW = OFF LDO_CORE = OFF VDD_Clamp = ~0.75 V LDO_RADIO = OFF	No Clocks	SysRAMx = optionally retained	▪ Wake up from P0_1, P0_2, P0_3, P0_4, P0_5

Table 30 shows the typical rail voltages and their drivers present during various PMU modes.

Table 30: Power rails drivers and voltages

<table>
<tr><th>Configurations</th><th>Mode</th><th>V_{BAT_HIGH}</th><th>V_{BAT_LOW}</th><th>V_{DD}</th></tr>
<tr><td rowspan="3">Buck</td><td>Active</td><td rowspan="3">Battery (3.6 V – 1.8 V)</td><td>DCDC out (1.2 V)</td><td>LDO_CORE (0.9 V)</td></tr>
<tr><td>Deep or Extended Sleep</td><td>LDO_LOW (1.2 V)</td><td>LDO_CORE in retain mode (0.75 V)</td></tr>
<tr><td>Hibernation</td><td>0 V (none)</td><td>VDD_Clamp (~0.75 V)</td></tr>
<tr><td rowspan="3">Boost</td><td>Active</td><td>DCDC out (3 V - 1.8 V)</td><td rowspan="3">Battery (1.5V – 1.2 V)</td><td>LDO_CORE 0.9 V</td></tr>
<tr><td>Deep or Extended Sleep</td><td>Parasitic diodes in DCDC or clamp (V_{BAT_LOW}) (Note 1)</td><td>LDO_CORE in retain mode (0.75 V)</td></tr>
<tr><td>Hibernation</td><td>Parasitic diodes in DCDC or clamp (V_{BAT_LOW}) (Note 1)</td><td>VDD_Clamp (~0.75 V)</td></tr>
<tr><td rowspan="3">Bypass</td><td>Active</td><td rowspan="3">Battery (3.3 V – 1.2 V)</td><td rowspan="3">Battery (3.3 V – 1.2 V)</td><td>LDO_CORE (0.9 V)</td></tr>
<tr><td>Deep or Extended Sleep</td><td>LDO_CORE in retain mode (0.75 V)</td></tr>
<tr><td>Hibernation</td><td>VDD_Clamp (~0.75 V)</td></tr>
</table>

Note 1 Parasitic diodes in DCDC and LDO_LOW circuits prevent V_{BAT_HIGH} from dropping to 0 V. V_{BAT_HIGH} can also be programmed to be clamped to V_{BAT_LOW}.

4.2.2.3 VDD Level in Hibernation

While in hibernation, the Always On domain (PD_AON) is supplied by a clamp. Because the reference is not enabled, the actual voltage supplied by the clamp depends on the load and temperature. To ensure proper operation of the PD_AON across the application operating temperature range and load, it is recommended to configure the voltage level of the VDD_Clamp using POWER_AON_CTRL_REG[LDO_RET_TRIM] according to Table 31.

Table 31: VDD_Clamp recommended settings over temperature and load

Temperature range	0 kB Retained RAM	64 kB Retained RAM
-40 ºC to +40 ºC	0xE	0xC
-40 ºC to +60 ºC	0xD	0xB
-40 ºC to +85 ºC	0xB	0xA
-40 ºC to +105 ºC	0xB	0x9

4.2.2.4 Retainable Registers

When the system enters one of the sleep modes, some registers need to retain their values even though their power domain might be shut down. These special retainable registers and their power domains are described in Table 32.

Table 32: Retainable registers

Power domains	Retainable registers
PD_SYS	**OTPC_MODE_REG**
	OTPC_TIM1_REG
	OTPC_TIM2_REG
	OTPC_AHBADR_REG
	OTPC_CELADR_REG
	OTPC_NWORDS_REG
	DEBUG_REG
PD_RAD	**BLE_CNTL2_REG**
	RF_ADCI_DC_OFFSET_REG
	RF_ADCQ_DC_OFFSET_REG
	RF_DC_OFFSET_RESULT_REG
	RF_DC_OFFSET_FULL_RES_REG
	RF_DC_OFFSET_MPAR_RES0/1/2/3_REG

4.2.3 Programming

4.2.3.1 Buck Configuration

In Buck configuration (Figure 4), the voltage on V_{BAT_LOW} and V_{DD} are generated from V_{BAT_HIGH} in the following ways:

- **Hibernation mode**

In Hibernation mode, the V_{BAT_LOW} rail is not powered and the digital core V_{DD} is supplied by the VDD clamp.

The VDD clamp is supplied automatically by selecting the highest of V_{BAT_HIGH} and V_{BAT_LOW}. Because in Buck configuration the V_{BAT_HIGH} rail is always the highest supply on the chip, it is safe to disable this automatic supply selection. Setting POWER_AON_CTRL_REG[CMP_VCONT_SLP_DISABLE] = 0x1 forces the clamp to use V_{BAT_HIGH} as supply and reduces the hibernation current by approximately 40 nA. VDD Clamp level (POWER_AON_CTRL_REG[LDO_RET_TRIM]) must follow the settings in Table 31.

- **Extended Sleep and Deep Sleep modes**

In the Extended Sleep mode or Deep Sleep mode, the V_{BAT_LOW} rail can be supplied either from the Low I_Q DCDC or the LDO_LOW which is in a retention low power mode. The digital core V_{DD} is powered from LDO_CORE, retention mode. To configure this mode, apply the following settings:

- POWER_CTRL_REG[LDO_CORE_RET_ENABLE] = 0x1
- Wait 300 µs for LDO_CORE_RET to settle
- POWER_LVL_REG[FORCE_RCX_LDO_CORE_RET] = 0x1
- POWER_AON_CTRL_REG[LDO_RET_TRIM] = 0xF
- Low I_Q DCDC:
  - DCDC_SLP_CTRL_REG[DCDC_SLP_ENABLE] = 0x1
  - LDO_LOW: POWER_CTRL_REG[LDO_LOW_CTRL_REG] = 0x1
- LDO_LOW:
  - DCDC_SLP_CTRL_REG[DCDC_SLP_ENABLE] = 0x0
  - POWER_CTRL_REG[LDO_LOW_CTRL_REG] = 0x3

- **Active and Idle modes**

The V_{BAT_LOW} rail is powered by LDO_LOW or by the DCDC converter. To enable the DCDC converter, apply the following settings:

- POWER_LEVEL_REG[DCDC_LEVEL] = 0x0
- DCDC_CTRL_REG[DCDC_ENABLE] = 0x1

4.2.3.2 Boost Configuration

In Boost configuration (Figure 5), voltages on V_{BAT_HIGH} and V_{DD} are generated from V_{BAT_LOW} as explained below:

- **Hibernation mode**

Hibernation mode is generally a state in which the device stays longer. It is recommended to force the VBAT_HL_RES switch on to reduce the power consumption from the comparator. When the VBAT_HL_RES switch is on, V_{BAT_HIGH} is connected to V_{BAT_LOW}. The register setting is:

- POWER_AON_CTRL_REG[VBAT_HL_CONNECT_RES_CTRL] = 0x1

In the hibernation mode of the Boost configuration, the digital core V_{DD} is powered from VDD Clamp. VDD Clamp level (POWER_AON_CTRL_REG[LDO_RET_TRIM]) must follow the settings in Table 31.

- **Extended Sleep and Deep Sleep modes**

The digital core V_{DD} is supplied by LDO_CORE switched to retention mode. The reference voltage for this LDO is generated by periodically enabling and sampling the bandgap (see Section 4.3.3 for details). V_{BAT_HIGH} rail, can be supplied by the following options:

- *Diodes controlled:* V_{BAT_HIGH} is connected to V_{BAT_LOW} through diodes. This option is useful for short sleep durations, typically below a second. When no significant load is present on V_{BAT_HIGH}, it will not drop below V_{BAT_LOW} during the sleep interval. This is the reset setting.
- *Software controlled:* V_{BAT_HIGH} and V_{BAT_LOW} are connected through VBAT_HL_RES. In typical use cases, this setting is energy efficient for sleep durations longer than one second. This mode requires the following register settings:
  - POWER_CTRL_REG[LDO_CORE_RET_ENABLE] = 0x1
  - Wait 300 µs for LDO_CORE_RET to settle
  - POWER_LVL_REG[FORCE_RCX_LDO_CORE_RET] = 0x1
  - POWER_AON_CTRL_REG[LDO_RET_TRIM] = 0xF
  - POWER_AON_CTRL_REG[VBAT_HL_CONNECT_RES_CTRL] = 0x2
- *Hardware controlled:* For short sleep durations, typically under one second, in combination with increased load conditions that can discharge V_{BAT_HIGH} to the level of V_{BAT_LOW} during the sleep interval, it is recommended to use the hardware-controlled V_{BAT_HIGH} connection, in which the two rails are connected as soon as V_{BAT_HIGH} drops to the same level of V_{BAT_LOW} (+/- 50 mV). This mode requires the following register settings:
  - POWER_CTRL_REG[LDO_CORE_RET_ENABLE] = 0x1
  - Wait 300 µs for LDO_CORE_RET to settle

  - POWER_LVL_REG[FORCE_RCX_LDO_CORE_RET] = 0x1
  - POWER_AON_CTRL_REG[LDO_RET_TRIM] = 0xF
  - POWER_CTRL_REG[CMP_VBAT_HIGH_OK_ENABLE] = 0x1
  - POWER_AON_CTRL_REG[VBAT_HL_CONNECT_RES_CTRL] = 0x3
- **Active and Idle modes**

In these modes, the V_{BAT_HIGH} rail is normally powered by the DCDC converter at 1.8 V. This level can be changed to 2.5 V or 3.0 V by configuring POWER_LEVEL_REG[DCDC_LEVEL].

Because the Boost converter is only needed to generate 1.8 V allowing for accessing to the OTP cell and supply the GPIOs, it is possible to disable it when access to the OTP is not required and no GPIOs are being driven. In this mode the $V_{BAT\_HL\_SW}$ should be activated to clamp V_{BAT_HIGH} at V_{BAT_LOW}, allowing some load on V_{BAT_HIGH} and GPIO operation at a reduced speed and voltage. Note that this mode rapidly discharges V_{BAT_HIGH} from the boosted voltage down to V_{BAT_LOW} level. Also, the POR_HIGH block has to be masked or disabled to avoid resets:

- If it is masked, the POR_HIGH status remains available, but it will not generate a reset.
- If it is disabled, the status becomes unavailable as well.

The following settings are required to enter this mode of operation:

- POWER_CTRL_REG[POR_VBAT_HIGH_DISABLE] = 0x1 (Disable) or
- POWER_AON_CTRL_REG[POR_VBAT_HIGH_RST_MASK] = 0x1 (Mask)
- DCDC_CTRL_REG[DCDC_ENABLE] = 0x0
- POWER_CTRL_REG[VBAT_HL_CONNECT_MODE] = 0x1

4.2.3.3 Bypass Configuration

In the bypass configuration, the V_{BAT_HIGH} and V_{BAT_LOW} rails are shorted on the PCB. This configuration is detected by the chip as a boost configuration, but because the boost converter is not able to generate a voltage on V_{BAT_HIGH}, the initial voltage must be above 1.75 V to allow the OTP to be read and mirrored.

Software can disable the DCDC converter and LDO_LOW to reduce quiescent current and avoid unnecessary switching of the DCDC converter. The following register settings are required to accomplish this:

- DCDC_CTRL_REG[DCDC_ENABLE] = 0x0
- POWER_CTRL_REG[LDO_LOW_CTRL_REG] = 0x1

If voltage drops below 1.75 V, POR_HIGH must be masked to prevent unnecessary resets. OTP reads cannot be performed after this point. Masking POR_HIGH is done by the following setting:

- POWER_AON_CTRL_REG[POR_VBAT_HIGH_RST_MASK] = 0x1

When POR_HIGH is masked, its status remains available, but it does not generate a reset. POR_HIGH can be also disabled by the following setting:

- POWER_CTRL_REG[POR_VBAT_HIGH_DISABLE] = 0x1

When POR_ HIGH is disabled, its status becomes unavailable.

4.3 Hardware FSM (Powerup, Wake-up, and Go-to-sleep)

Figure 8 shows the hardware Finite State Machine (FSM) responsible for the powerup, wake-up, and go-to-sleep processes of the system.

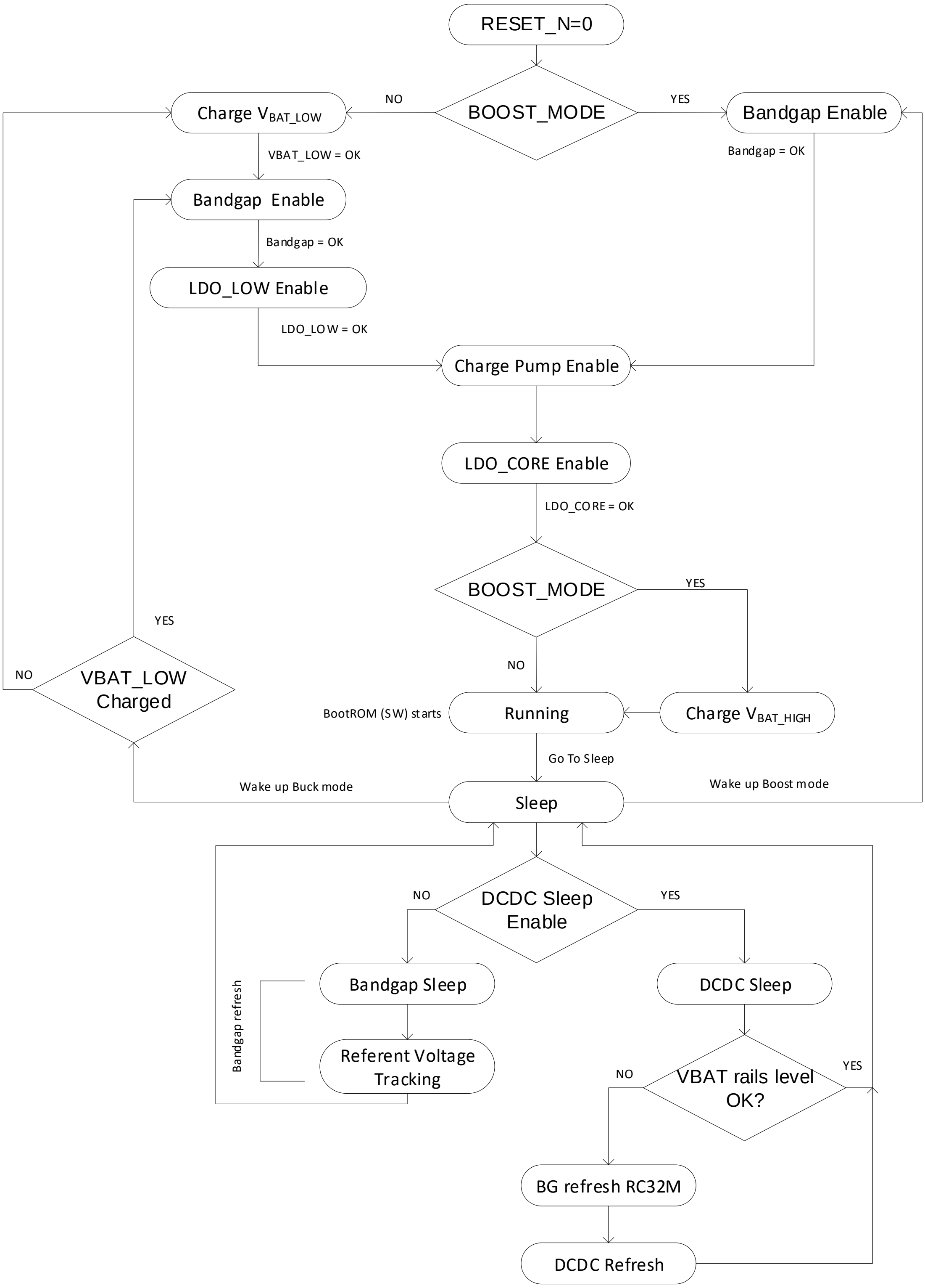

Figure 8. Powerup/wake-up/sleep FSM diagram

The details of the powerup, wake-up, and sleep sequences of the FSM for the different modes are described in the following sections.

In Boost configuration, the DCDC is enabled by the Booter during powerup. After waking up, if the DCDC was left enabled, it is started by the hardware FSM.

In Buck configuration, the DCDC is enabled by programming DCDC_CTRL_REG[DCDC_ENABLE] bit. When the system is allowed to go to sleep, the DCDC converter can remain on if POWER_CTRL_REG[DCDC_ENABLE_SLEEP] is set.

If DCDC converter is not used during sleep (LDO_LOW_RET case), the DCDC is started by the hardware FSM upon waking-up given that the DCDC_ENABLE is kept asserted before the system goes to sleep. If DCDC_ENABLE is cleared before the system goes to sleep, the DCDC can only be started by software asserting this bit after a wake-up.

4.3.1 Powerup/Wake-up in Buck Configuration

At the beginning of a powerup (cold boot), the PMU detects whether the system is in a Buck or a Boost configuration, and this decision is retained from that point on. Figure 9 shows the powerup (cold boot) sequence of the Buck configuration. When the system is at the start of the Buck configuration path, then V_{BAT_HIGH} rail has a stable supply already and "`boost_mode` = 0" (which means buck is identified) has been set.

To start the system, it is required that the V_{BAT_LOW} rail is also brought up to an acceptable level, which is done by hardware, enabling the resistive switch VBAT_HL_RES and monitoring POR_LOW. The rising voltage of the V_{BAT_LOW} rail eventually triggers POR_LOW to "ok" after that, the bandgap is enabled. The hardware FSM runs at 32 kHz from the RC32K oscillator at this time, and it dynamically changes to 512 kHz in one clock cycle after the switch is enabled. The hardware FSM continues working at 512 kHz. When the reference voltage from the bandgap is stable, LDO_LOW is enabled. The DCDC is not started in cold boot by hardware but needs to be activated by software for the first time. After a stable 1.2 V is generated, LDO_CORE is enabled to generate a stable V_{DD} of 0.9 V. After this, the main system clock, RC32MHz, is enabled. All conditions are now in place to release the system reset so that the Booter can start running. An indicative time, that is needed to power up the system in Buck configuration up until the application software starts running, is around 2.5 ms. The time required to charge V_{BAT_LOW} and V_{BAT_HIGH} depends on the external capacitor values on these rails.

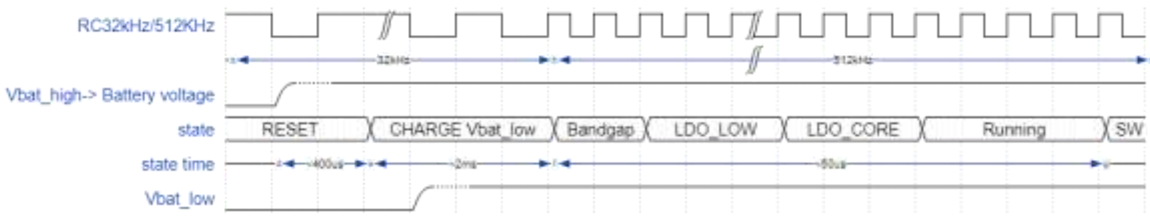

Figure 9. Power-up (buck)

In hibernation, the resistive switch VBAT_HL_RES can be closed to pre-charge V_{BAT_LOW} to the level of V_{BAT_HIGH} by programming POWER_AON_CTRL_REG[VBAT_HL_CONNECT_RES_CTRL]. Therefore, during the wake-up sequence from the hibernation mode, the "Charge V_{BAT_LOW}" step can be omitted (Figure 10) through POWER_AON_CTRL_REG[CHARGE_VBAT_DISABLE]. All other steps are the same as in the powerup cold-boot sequence. The total time needed to wake up the system from hibernation until booter software starts running is around 185 µs.

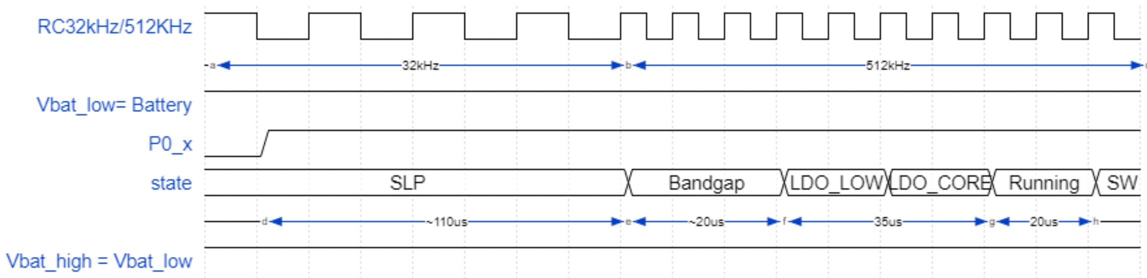

Figure 10. Wake-up from hibernation (buck)

The wake-up sequence from the Clocked, Extended Sleep, or Deep Sleep mode using an external GPIO toggle is shown in Figure 11. The difference between the powerup and wake-up sequence in Buck configuration is that the V_{BAT_LOW} rail is already charged through the switch VBAT_HL_RES in the wake-up sequence as shown in Figure 8. Therefore, when the system wakes up from a Deep Sleep or an Extended Deep Sleep, the bandgap is enabled within one 32 kHz clock cycle after the wake-up signal is triggered. After the bandgap is enabled, LDO_LOW and LDO_CORE is enabled one after the other. If DCDC is kept alive during sleep, the DCDC also switches from Sleep to Active mode after LDOs are enabled. In this case, LDO_LOW needs to be disabled by software. The total time which is needed to wake up the system until software starts running is around 800 µs. If RAM has been retained, running software means application, if not, then the Booter is started.

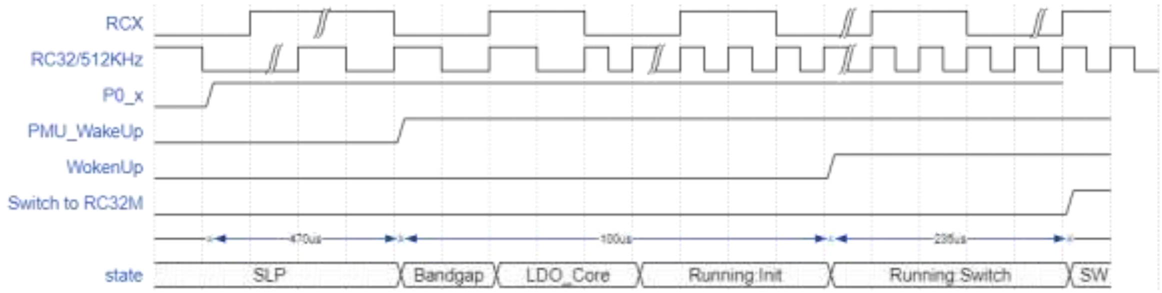

Figure 11. Wake-up (buck)

A GPIO trigger must go through the wake-up controller first, which requires seven RCX clock cycles before it is allowed to trigger the PMU and have the state machine running. Even after all power rails are done, switching the system clock from RCX into RC32M (so software stars running) takes 3.5 RCX clock cycles (indicated in state = RUNNING:SWITCH) in Figure 11Figure 11.

If the system wakes up from an internal timer running at the RCX clock, then the WokenUp signal is asserted six RCX clock cycles after the timer generates the interrupt (for example, ~400 µs).

4.3.2 Power-up/Wake-up in Boost Configuration

At the beginning of a power-up (cold boot), the PMU detects whether the system is in a Buck or a Boost configuration, and this decision is retained in the system. The power-up (cold boot) sequence of the Boost configuration is shown in Figure 12. When the system is at the start of the Boost configuration path, then V_{BAT_LOW} has a stable supply already and "`boost_mode` = 1" (which means boost is identified) has been set.

Because the supply on the V_{BAT_LOW} rail is already on an acceptable level, the bandgap will be enabled in the first step and LDO_CORE in the second step. After this, the resistive switch VBAT_HL_RES is closed to bring V_{BAT_HIGH} up to the same level as V_{BAT_LOW}. This is needed for the DCDC converter to start, because its drivers run on the V_{BAT_HIGH} rail. When the V_{BAT_HIGH} rail is stable, the system reset can be released for the Booter to start running. The DCDC will be enabled during the first actions in the Booter (Section 4.5). An indicative time which is needed to power up the system in Boost configuration up until software starts running is around 1.2 ms. The time required to charge V_{BAT_HIGH} depends on the external capacitor values on this rail.

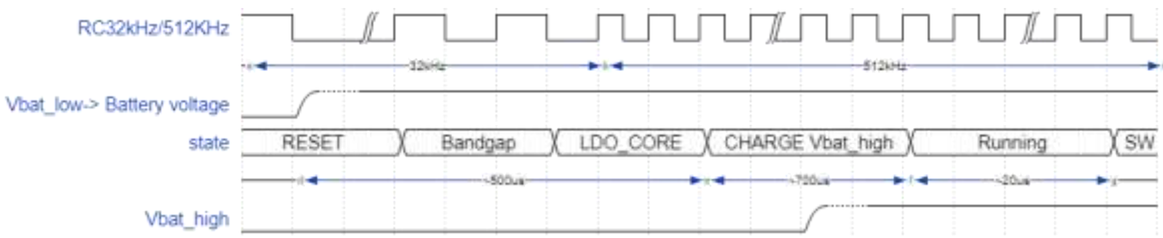

Figure 12. Power-up (boost)

In hibernation, the resistive switch VBAT_HL_RES can be closed to pre-charge V_{BAT_HIGH} to the level of V_{BAT_LOW} by programming POWER_AON_CTRL_REG[VBAT_HL_CONNECT_RES_CTRL]. Therefore, during the wake-up sequence from the hibernation mode, the "Charge V_{BAT_HIGH}" step can be omitted (Figure 13) through POWER_AON_CTRL_REG[CHARGE_VBAT_DISABLE]. All other steps are same as in the powerup cold-boot sequence. The total time needed to wake up the system from hibernation until booter software starts running is around 180 µs.

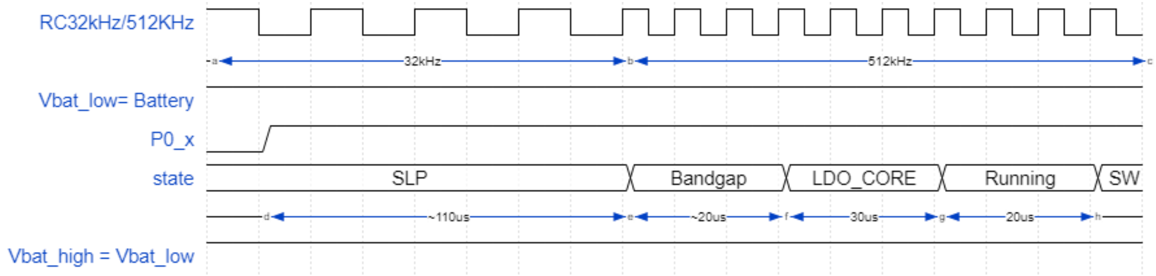

Figure 13. Wake-up from hibernation (boost)

In the wake-up sequence from Deep or Extended Sleep using a GPIO toggle, the "Charge V_{BAT_HIGH}" step lasts minimum four clock cycles (Figure 14). This time is sufficient to quickly re-charge V_{BAT_HIGH} under the condition that the leakage current and the sleep interval are sufficiently low and the V_{BAT_HIGH} voltage does not drop below V_{BAT_LOW}. The state Charge V_{BAT_HIGH} can be omitted by using the resistive paths. The total time which is needed to power up the system up until software starts running is around 865 µs. If RAM has been retained, running software means application, if not then the Booter is started.

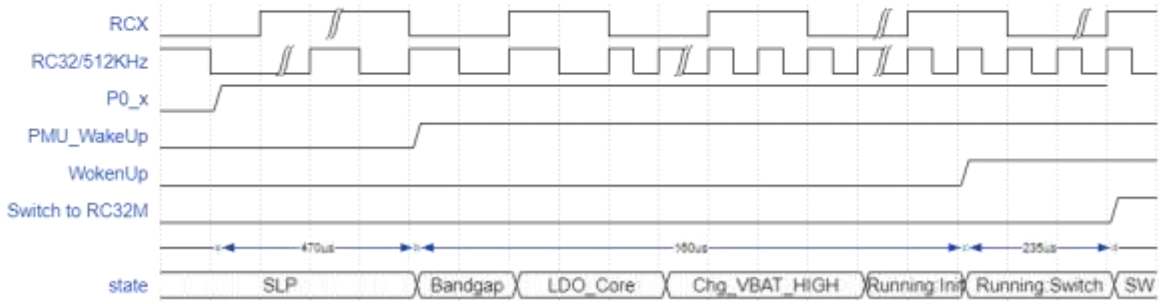

Figure 14. Wake-up from extended/deep sleep (boost)

A GPIO trigger must go through the wake-up controller first, which requires seven RCX clock cycles before it can trigger the PMU and have the state machine running. Even after all power rails are done, switching the system clock from RCX into RC32M (so software stars running) takes 3.5 RCX clock cycles (indicated in state = RUNNING:SWITCH) in Figure 14.

If the system wakes up from an internal timer running at the RCX clock, then the WokenUp signal is asserted six RCX clock cycles after the timer generates the interrupt (for example, ~400 µs).

4.3.3 Go-to-sleep and Refresh Bandgap

System can enter a sleep state, either sleep Buck or sleep Boost, depending on the status of the `boost_mode` bit. The sleep state disables the power-consuming blocks and triggers the "hold mode" for the bandgap referenced voltages. After a certain amount of time (sleep refresh counter), these "hold" voltages need to be refreshed. The lower loop of the FSM in Figure 8 enables the bandgap, refreshes the voltages, checks for BOD events through the POR circuits and if ok, resets the refresh timer and goes back to sleep. This is an autonomous cycle led by hardware until the system is woken up by the wake-up event. The refresh timer can be configured by setting the PMU_SLEEP_REG [BG_REFRESH_INTERVAL] bit field (1 LSB = 64 × 32 kHz clock cycles). The go-to-sleep and the bandgap refresh sequence is shown in Figure 15.

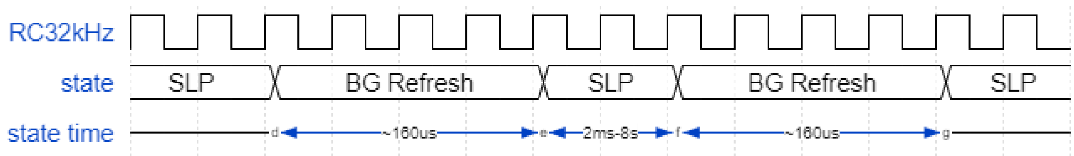

Figure 15. Go-to-sleep and bandgap refresh

4.4 OTP Memory Layout

The OTP memory has to be programmed according to a specific layout that structures information to be easily accessible from the BootROM code as well as the actual application. Figure 16 shows an overview of the layout scheme.

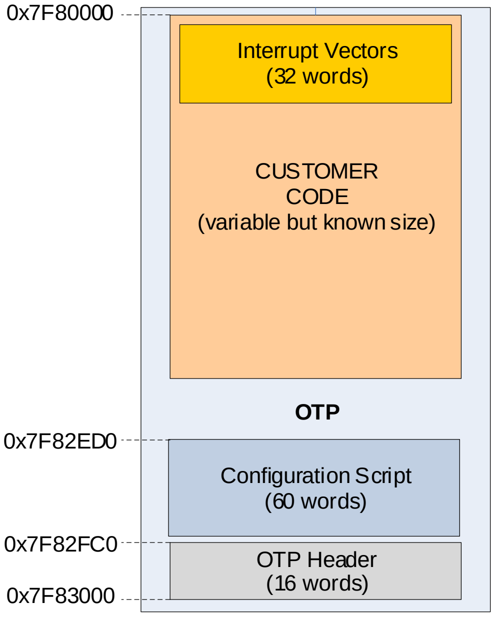

Figure 16. OTP layout scheme

The OTP memory is a matrix of 3Kx32-bit words. The contents are:

- **Interrupt Vectors:** they are the vectors of the interrupt service routines and always reside at the address 0x0. This is part of the application (customer) code. The size of this vector list is 32 words.
- **Customer Code:** it contains the applications and the profiles that a customer has developed. The size is known and fixed before the mass production and the programming of the OTP.
- **Configuration Script (Section 4.4.2):** it is used to program registers with values that are defined during production testing, to store a trim value for the application software, and to define the UART time-out timer during booting. It is executed by the Booter to prepare and initialize the system prior to that the CPU starts running the application code. Available size is 60 words.
- **OTP Header:** it contains various information about the configuration of the system and the Bluetooth LE-specific data. Size of the header is 16 words.

4.4.1 OTP Header

The OTP header breakdown is shown in Table 33.

Table 33: OTP header

<table>
<tr><th rowspan="2">Address</th><th rowspan="2">Words (32-bit)</th><th rowspan="2">Description</th><th colspan="2">Programmed during</th></tr>
<tr><th>Chip Test</th><th>Product Manufacturing</th></tr>
<tr><td>7F82FC0</td><td>1</td><td>Application Programmed Flag #1
0x1234A5A5 = Application is in OTP</td><td></td><td>Yes</td></tr>
<tr><td>7F82FC4</td><td>1</td><td>Application Programmed Flag #2
0xA5A51234 = Application is in OTP</td><td></td><td>Yes</td></tr>
<tr><td>7F82FC8</td><td>1</td><td>Boot specific configuration:
▪ Bits[7:0] :
• 0xAA = Boot from SPI port at a specific location
• 0xFF = Normal sequence
▪ Bits[15:8] = Wake-up Command opcode
▪ Bits[23:16] = SPI_DIV
▪ Bits[31:24]:
• 0x00 = Two-wire UART (P0_0/P0_1)
• 0x01 = One-wire UART (P0_3)
• 0x02 = One-wire UART (P0_5)
• 0x03 = Two-wire UART (P0_1/P0_3)
• Default (all other values) = Two-wire UART (P0_0/P0_1)</td><td></td><td>Yes</td></tr>
<tr><td>7F82FCC</td><td>1</td><td>Boot specific port mapping:
Bits[7:4] = SPI_CLK, Port number
Bits[3:0] = SPI_CLK, Pin number
Bits[15:12] = SPI_EN, Port number
Bits[11:8] = SPI_EN, Pin number
Bits[23:20] = SPI_DO, Port number
Bits[19:16] = SPI_DO, Pin number
Bits[31:28] = SPI_DI, Port number
Bits[27:24] = SPI_DI, Pin number</td><td></td><td>Yes</td></tr>
<tr><td>7F82FD0</td><td>1</td><td>Device and Package Flag:<table><tr><th>Package</th><th>Bits[7:0]</th><th>Bits[23:16]</th></tr><tr><td>FCQFN24</td><td>0xAA</td><td>0xFF</td></tr><tr><td>KGD</td><td>0xAA</td><td>0x11</td></tr></table>▪ Bits[15:8]:
• 0x35 = DA14535
• Others = Reserved
▪ Bits[31:24] = Reserved</td><td>Yes</td><td></td></tr>
<tr><td>7F82FD4</td><td>2</td><td>Bluetooth Device Address (64-bit word).
It is handled as a string of bytes.</td><td></td><td>Yes</td></tr>
<tr><td>7F82FDC</td><td>1</td><td>OTP DMA length (number of 32-bit words).</td><td></td><td>Yes</td></tr>
</table>

Address	Words (32-bit)	Description	Programmed during	
			Chip Test	Product Manufacturing
7F82FE0	1	Position: Bits[7:0] = X coordinate Bits[15:8] = Y coordinate Bits[23:16] = Wafer # Bits[31:24] = LOT #	Yes	
7F82FE4	1	Tester: Bits[7:0] = Tester_Site Bits[15:8] = Tester_ID (LSB) Bits[23:16] = Tester_ID (MSB) Bits[31:24] = Reserved	Yes	
7F82FE8	1	TimeStamp: Bits[7:0] = TS_Byte0 Bits[15:8] = TS_Byte1 Bits[23:16] = TS_Byte2 Bits[31:24] = TS_Byte3	Yes	
7F82FEC	5	Reserved for Future Needs		

The Device and Package Flag reflects what the current device (DA14535) is and which package is used. Default (unprogrammed) values are 0xFFFFFFFF.

Boot specific mapping value is used to define a specific configuration for the SPI interface when used for booting from an external device (either an MCU or a FLASH). Byte0 is the flag to instruct the BootROM to use the specific SPI pin mapping and skip the rest of the serial peripheral interfaces. The BootROM takes care of waking up an external flash when the flash memory is in deep power-down state.

Byte1 is used for the Wake-up Command opcode that the flash memory responds to. If Byte0 is left unprogrammed, the BootROM sends the "0xAB" opcode by default. Furthermore, the BootROM can wake up the external flash by toggling the CS pin.

Two more flags indicate whether the application code has indeed been programmed (burned) into the OTP. Both flags are read by the BootROM software designating that the system is in Normal mode and not in Development mode (Section 4.5).

4.4.2 Configuration script

The Configuration Script (CS) is a table of 32-bit entries and is 60 words deep, so in total the CS can utilize 240 bytes of space.

The CS is used to program registers with values that are defined during production testing, to store a trim value for the application software, and to define the UART time-out timer during booting. It is executed by the Booter to prepare and initialize the system prior to that the CPU starts running the application code.

The format of the commands in the CS is shown in Table 34.

Table 34: CS commands and description

#	Command type	Description
1	Start Command	One 32-bit word containing 0xA5A5A5A5 to signal a valid CS is in place.
2	Register Configuration	▪ One 32-bit word containing an address of an existing register ▪ One 32-bit word containing the data value of the register

#	Command type	Description
3	SDK Value	One 32-bit word which is equal to 0x9000YYXX indicating that the next word is a value stored during production testing. More specifically: ▪ 9: it indicates that the following word(s) are not to be stored to registers but will be used by the SDK software. ▪ YY: it indicates that YY amount of words follow. ▪ XX: it is an increasing value and can be used for indexing by the software application. If YY > 1, XX will not be increased for the words that belong to the same value. One or more 32-bit words can represent one value.
4	SWD mode	One 32-bit word which is equal to 0x70000000. It prevents the JTAG from being enabled at the end of the Booter and the Booter will not enter the endless while (1) loop. Instead, it will continue to rescan all peripherals in the development mode path.
5	UART STX timeout value	One 32-bit word which is equal to 0x80XXXXXX. The XXXXXX is used to program the selected STX timeout in multiples of 100 μs. So, for example, 0x80000028 is 40 × 100 μS = 4 ms.
6	SPI Clock value	0xA0000000 This value overwrites the default 2-MHz clock speed of the SPI boot path and sets it to 32 MHz.
7	SPI Flash wake up timeout value	0xBYYYXXXX YYY = This value is used to program the selected SPI CS delay timeout (before CS toggle) in multiples of 50 μs. So, for example, 0xB0040000 is 4x50 μs = 200 μs. XXXX = This value is used to program the selected SPI flash wakeup timeout (delay after wakeup command is sent) in multiples of 50 μs. So, for example, 0xB0000008 is 8x50 μs = 400 μs.

The Booter stops processing the CS when it encounters an empty OTP value (0xFFFFFFFF). This way, no more processing time is spent to check the rest and it is possible to add new entries later, for example, to patch/update previous entries.

An example describing the format of the configuration script is shown in Table 35.

Table 35: CS example

Words	Even words	Odd words	Description
0-1	0xA5A5A5A5	0x80000028	Start command of the CS Script, followed by STX timeout value of 4 ms (40 × 100 μs).
2-3	<Address>	<Value>	Booter automatically writes the <Value> to the <Address>.
4-5	0x90000301	<Value>	Three calibration values stored during production testing. SDK should know what this is for.
6-7	<Value>	<Value>	
8-9	<Address>	<Value>	Booter automatically write the <Value> to the <Address>.
10-11	<Address>	<Value>	Booter automatically write the <Value> to <Address>.
12-13	0x90000402	<Value1>	Four calibration values stored during production testing. SDK should know what this is for.
14-15	<Value2>	<Value3>	Calibration value stored during production testing. SDK should know what this is for.
16-17	<Value4>	0x70000000	Disable SWD.
18-19	0xFFFFFFFF	(don't care)	Booter stops running the CS after an empty entry, so anything after this is "don't care".

4.5 BootROM Sequence

The booting process of DA14535 is shown in Figure 17. The Booter is always executed when a POR or a Hardware Reset occurs, or the RESET_ON_WAKEUP feature is configured.

The booter starts executing with the RC32M clock to speed up its execution. Then the Booter checks whether the system is in the Boost configuration or not. The configuration (Buck or Boost) is already decided by the hardware and should be readable by software at the ANA_STATUS_REG[BOOST_SELECTED] bit field. To access the OTP, V_{BAT_HIGH} needs to be set at ≥ 1.8 V. In the Boost configuration, the DCDC is enabled to boost V_{BAT_HIGH} at 1.8 V and BOOST_VBAT_OK = 1 confirms that the voltage at the DCDC is stable. After the OTP is operational, the Booter initializes the UART baud rate at 115.2 kHz, and the CS (Section 4.4.2) is enabled to be executed.

After the Configuration Script has been executed, the Booter has to decide whether the device is in Development or Normal mode by reading the two words indicated as application flags in the OTP. The OTP image is copied into RAM starting at address 0x0 by the Booter.

In Development mode, the "Boot from Specific" flag is evaluated. If the flag is programmed, new pin locations for booting from an external SPI slave to make DA14535 an SPI Master is set. The "Boot from Specific" flag addresses mostly the QFN package allowing for booting from a different pin configuration than the default one, so that the system can boot from an external FLASH using Development mode. The details of the configuration are presented in Section 4.4.1. If this path is entered, the system always tries to boot from UART so that the SPI Flash can be updated if needed. Any of the three UART configurations specified in Table 36 can be selected by writing bits [31:24] at the "Boot specific config" field in the OTP header. If booting from SPI Flash fails, the Booter jumps back to the normal scan sequence of the peripheral devices.

If the "Boot from Specific" flag is not programmed, the system should continue with scanning the different serial interfaces to identify whether a device is connected to it. After OTP is disabled, seven steps as described in Table 36 are performed. Before using the UART, the XTAL32M clock needs to be enabled. All the boot steps are protected by a timeout.

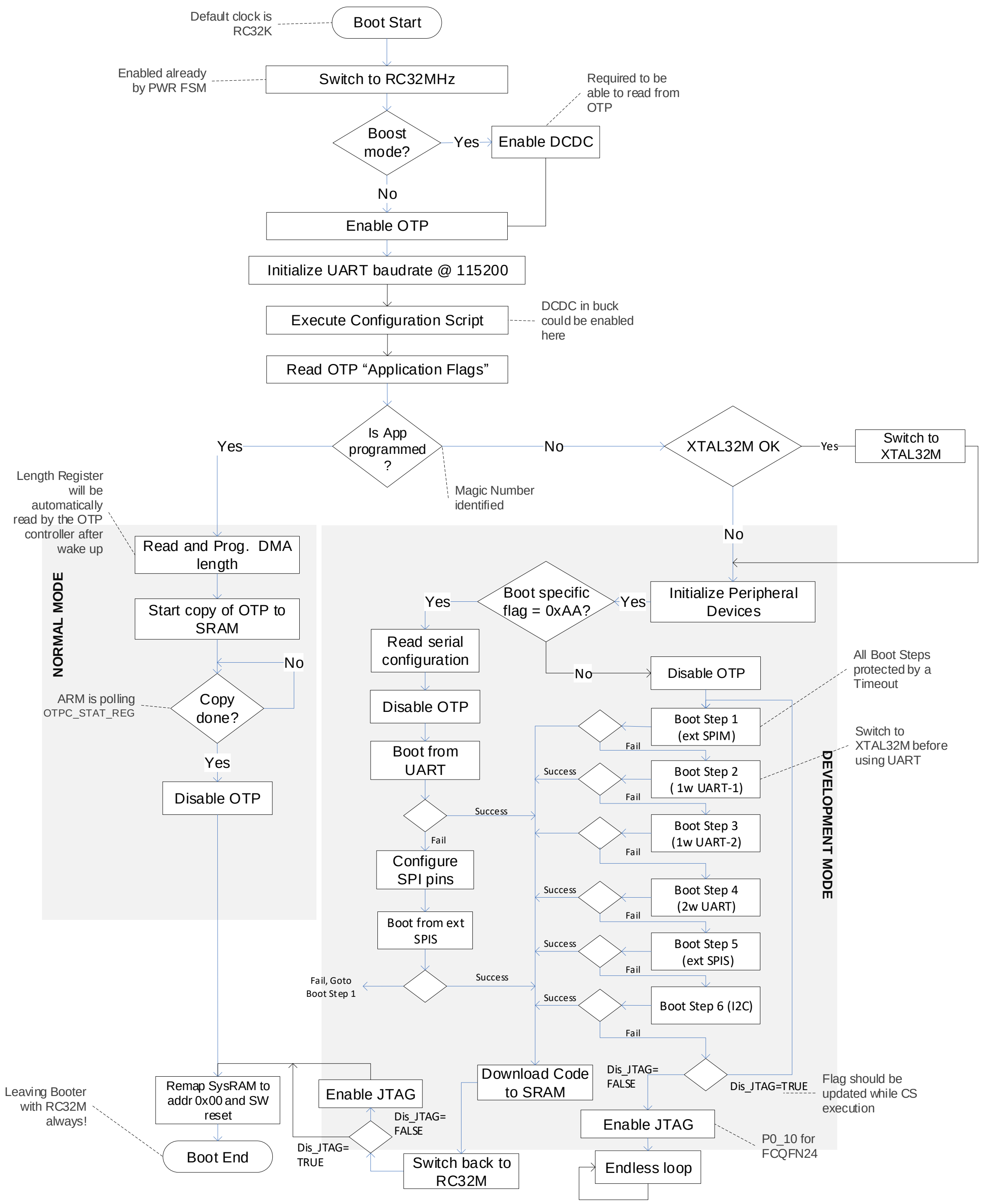

Figure 17. BootROM sequence

The one-wire UART boot capability is introduced due to the limited amount of the GPIOs. Because the booting from UART protocol is a half-duplex, a single GPIO is used in DA14535 for the external UART. The protocol is the same as for a two-wire UART booting except that the Booter software needs to change the pin direction before sending or receiving information.

Table 36: Booting sequence steps

	Step 1: Boot from External SPI Master	Step 2: Boot from 1-wire UART (First Option)	Step 3: Boot from 1-wire UART (Second Option)	Step 4: Boot from 2-wire UART	Step 5: Boot from External SPI Slave	Step 6: Boot from I2C
P0_0/RST	MISO			TX	MOSI	
P0_1	MOSI			RX	SCS	
P0_2						
P0_3	SCS		RXTX		MISO	SDA
P0_4	SCK				SCK	SCL
P0_5		RXTX (Default)				
P0_6						
P0_7						
P0_8						
P0_9						
P0_10						
P0_11						

If no bootable devices are found on any of the serial interfaces, the Booter can do two things, depending on what is stored in the CS. If the "Debugger disable" (0x70000000) command is stored there, the Booter starts scanning for peripherals again. Otherwise, it enters the endless loop with the debugger (JTAG) being enabled. The debugger is connected to P0_10 in the FCGQFN24 package.

After the BootROM sequence has completed, the default system clock is RC32M, regardless of which boot path has been chosen and all GPIOs are set back to their default reset values.

5. Reset

5.1 Introduction

The DA14535 comprises a reset (RST) pad which is active high. It contains an RC filter with a resistor of 465 kΩ and a capacitor of 3.5 pF to suppress spikes. It also contains a 25 kΩ pull-down resistor. This pad should be driven externally by a field-effect transistor (FET) or a single button connected to VBAT. The typical latency of the RST pad is in the range of 2 µs.

Features

- RC spike filter on RST to suppress external spikes (465 kΩ, 3.5 pF).
- Three different reset lines (software, hardware, and POR).
- Latching the cause of a reset operation (RESET_STAT_REG).
- Configurable POR circuitry.

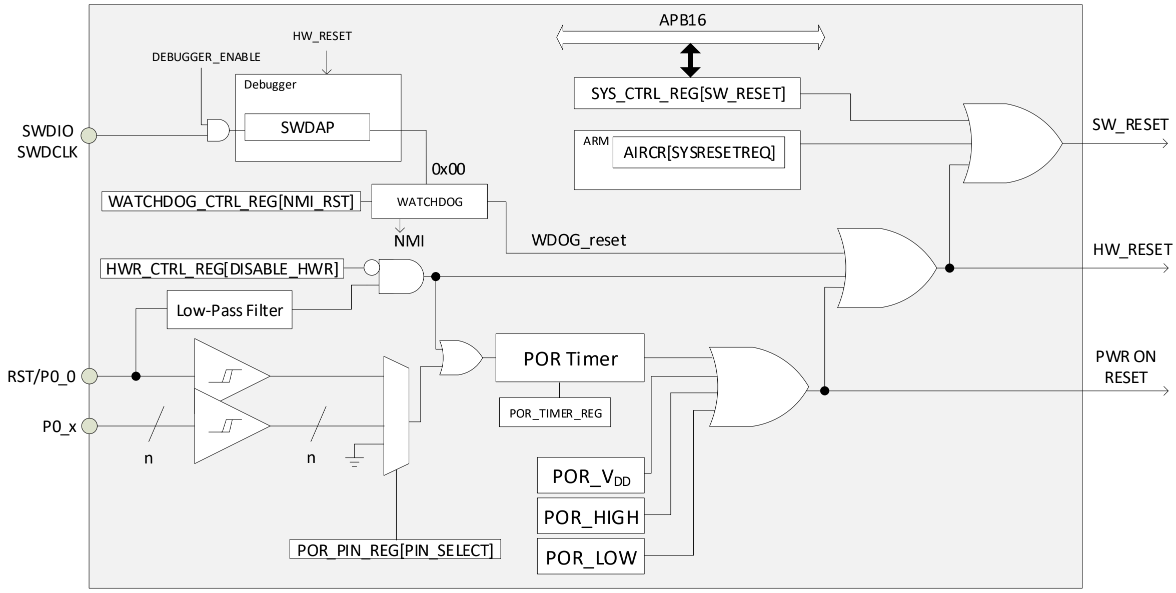

Figure 18. Reset block diagram

5.2 Architecture

5.2.1 POR, Hardware, and Software Reset

There are three main reset signals in DA14535:

- The Power-On Reset (POR): it is optional triggered by a GPIO set as the POR source with a selectable polarity and/or the RST pad (P0_0) after a programmable time delay.
- The hardware reset: it is optional triggered by the RST pad (P0_0) when it becomes active for a short period of time (less than the programmable delay for POR).
- The software reset: it is triggered by writing the SYS_CTRL_REG[SW_RESET] bit.

The POR signal is generated:

- Internally and releases the system's flip flops as soon as the VDD, V_{BAT_HIGH}, and V_{BAT_LOW} voltages crossed the specified thresholds.
- Externally by a POR source (RST pad multiplexed on a GPIO or P0_0 configured as RST pin).

The hardware reset can also be automatically activated when the system wakes up from the Extended or Deep Sleep mode by programming the bit PMU_CTRL_REG[RESET_ON_WAKEUP]. The POR and the hardware reset basically run the cold start-up sequence and the BootROM code is executed.

Note
If the device is in Boost configuration and P0_1 is used as chip select signal for an external SPI Flash, the PULL_HW_BYPASS_REG[PULL_HW_BYPASS] bit must be set before the PMU_CTRL_REG[RESET_ON_WAKEUP] is set to ensure a proper boot sequence after the system wakes up from sleep.

The software reset is the logical OR of a signal from the Cortex CPU (triggered by writing SCB->AIRCR = 0x05FA0004) and the SYS_CTRL_REG[SW_RESET] bit. It is mainly used to reboot the system after the base address has been remapped.

The block diagram of the reset block is shown in Figure 18.

Certain registers are reset by POR only, or by POR and the hardware reset signal but not by the software reset. These registers are listed in Table 37.

Table 37: Reset signals and registers

Reset by POR only	Reset by POR or hardware reset	Reset by POR, hardware reset, or software reset
BANDGAP_REG	BLE_CNTL2_REG	The rest of the Register File
POR_PIN_REG	CLK_AMBA_REG[OTP_ENABLE]	
POR_TIMER_REG	CLK_FREQ_TRIM_REG	
HWR_CTRL_REG	CLK_RADIO_REG	
RESET_STAT_REG[PORESET_STAT]	CLK_CTRL_REG	
PAD_LATCH_REG	PMU_CTRL_REG	
POWER_AON_CTARL_REG	SYS_CTRL_REG	
GP_DATA_REG	TRIM_CTRL_REG	
TEST_VDD_REG	RAM_PWR_CTRL_REG	
	CLK_RC32K_REG	
	CLK_XTAL32K_REG	
	CLK_RC32M_REG	
	CLK_RCX_REG	
	XTALRDY_CTRL_REG	
	XTAL32M_CTRL0_REG	
	PMU_SLEEP_REG	
	POWER_CTRL_REG	
	POWER_LEVEL_REG	
	DCDC_CTRL_REG	
	RAM_LPMX_REG	
	HIBERN_CTRL_REG	
	CLK_RTCDIV_REG	
	RTC_CONTROL_REG	
	RTC_KEEP_RTC_REG	
	OTPC_*_REG	
	QDEC_*_REG	
	PULL_HW_BYPASS_REG	

Reset by POR only	Reset by POR or hardware reset	Reset by POR, hardware reset, or software reset
	All RF calibration registers	

5.2.2 POR Functionality

The POR functionality is available by two sources:

- RST Pad: the RST pad is always capable of producing a POR.
- GPIO Pin: a GPIO can be selected by the user application to act as a POR source.

The time needed for a GPIO pin selected for the POR to be active is stored in the POR_TIMER_REG. The register field POR_TIME is a 7-bit field which holds the time factor by which the total time for POR is calculated. The maximum value of the field is 0x7F. The total time for POR is calculated by the following formula:

$$\text{Total time} = \text{POR\_TIME} \times 4096 \times \text{RC32k clock period} \tag{1}$$

where RC32k clock period = 31.25 μs at 25 ˚C.

The maximum time for which a POR can be performed is ~16.2 seconds at 25 ˚C.

The RC32k clock frequency depends on temperature, so based on the temperature span of -10 ˚C to 50 ˚C, the clock frequency range is calculated to be 25 kHz to 39 kHz. Then,

$T_{PORcold}$ = 13 s

T_{PORhot} = 20.8 s

5.2.2.1 POR Timer Clock

The POR timer is clocked by the RC32k clock. If a software application disables the RC32k, the hardware takes care of enabling the RC32k clock when a POR source (the RST pad or a selected GPIO pin) is asserted. It should be noted that if the POR is generated from the RST pad, the RC32k operates with the reset (default) trimming value. If a GPIO pin is used as the POR source, the RC32k clock is trimmed. The timing difference between both cases is expected to be minor.

5.2.2.2 RST Pad

The RST pad produces a hardware reset if the pin active time is less than the programmed value in the POR_TIMER_REG register or a POR if the pin active time is greater than or equal to that value. Reset pad is always Active High.

5.2.2.3 POR from GPIO

When a GPIO is used as a POR source, the selected pin retains its capability to act as GPIO. The POR_PIN_REG[PIN_SELECT] field holds the required GPIO pin number. If the value of the PIN_SELECT field equals to 0, the POR triggered by GPIO functionality is disabled. The polarity of the pin can be configured by the POR_PIN_REG [POR_POLARITY] bit, where 0 means Active Low and 1 means Active High.

5.2.3 POR Timing Diagram

The operation of the POR triggered by both the RST pad and a selected GPIO pin is shown in Figure 19.

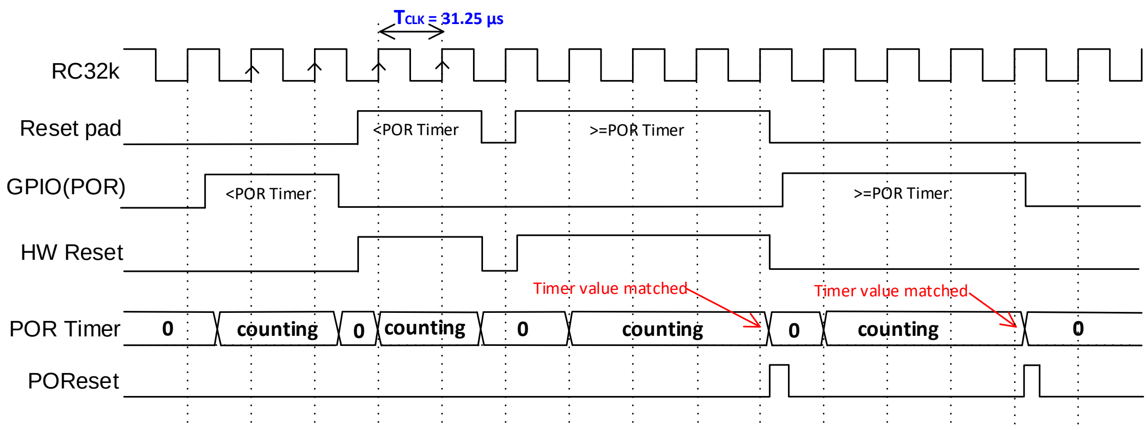

Figure 19. POR timing diagram

5.2.4 POR Considerations

When a POR source (the RST pad or a selected GPIO pin) is asserted, the POR timer starts to count. When the POR source is released before the timer has expired, the POR timer is reset to 0. If a POR source is asserted while there is already an asserted POR, and the first POR is released after the second POR is asserted, and the total time of the two asserted sources is larger than or equal to the POR_TIME, POR occurs.

It should also be noted that the POR timer triggered by the RST pad can only expire once. After the POR timer has expired, the RST pad has to be released so the timer can be reloaded. There is no such limitation when a GPIO is used as the POR source.

The POR_PIN_REG[PIN_SELECT] field cannot survive any reset (POR, hardware reset, or software reset), therefore, users must take special care on setting up the GPIO POR source right after a reset. This also applies to the POR_TIMER_REG[POR_TIME] field after a POR.

Be aware of that, if a GPIO is used as a POR source, the dynamic current of the system increases due to the dynamic current consumed by the RC32k oscillator. This increase is calculated to be from 100 nA to 120 nA and it is also present during sleep time period. POR from the RST pad does not add this dynamic current consumption.

5.3 Programming

To configure the functionality of triggering a POR by a GPIO pin:

1. Select a GPIO to be set as the POR source by programming POR_PIN_REG[POR_PIN_SELECT].
2. Set up the input polarity of the GPIO that causes POR by programming POR_PIN_REG[POR_PIN_POLARITY].
3. Configure the time for the POR to happen by programming POR_TIMER_REG[POR_TIME]. The default time is around three seconds.

NOTE
To set up the time when the RST pad produces a POR, just set the POR_TIMER_REG register.

6. Arm Cortex-M0+

6.1 Introduction

The Arm® 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 called Thumb, which was first supported in the ARM7TDMI processor, but it also uses several newer instructions from the Armv6 architecture and a few instructions from the Thumb-2 technology. Thumb-2 technology extends the previous Thumb instruction set to allow all operations to be carried out in one CPU state. The instruction set in Thumb-2 includes 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 only 56 base instructions, although some instructions can have more than one form. Although the instruction set is small, 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. It has a two-stage pipeline (fetch+predecode and decode+execute) as opposed to its predecessor (Cortex-M0) that has a three-stage pipeline (fetch, decode, and execute).

Figure 20 shows a simplified block diagram of the Cortex-M0+.

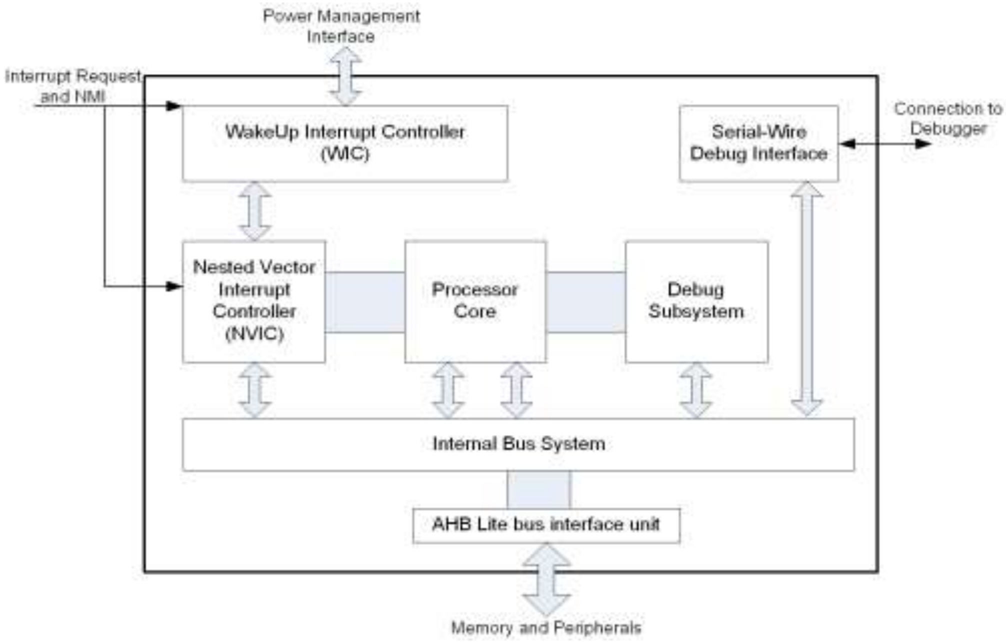

Figure 20. Arm Cortex-M0+ block diagram

Features

- Thumb instruction set: highly efficient, of high code density, and able to execute all Thumb and Thumb-2 instructions.
- High performance: up to 0.9 DMIPS/MHz (Dhrystone 2.1) with fast multiplier.
- Built-in Nested Vectored Interrupt Controller (NVIC): this makes interrupt configuration and coding of exception handlers easy. When an interrupt request is taken, the corresponding interrupt handler is executed automatically without the need to determine the exception vector in software.
- Interrupts can have four different programmable priority levels and the NVIC automatically handles nested interrupts.
- The design is configured to respond to exceptions (for example, interrupts) as soon as possible (minimum 15 clock cycles).
- Non maskable interrupt (NMI) input for safety critical systems.

- Easy to use and C friendly. There are only two modes, Thread mode and Handler mode. The whole application, including exception handlers, can be written in C without any assemblers.
- Built-in System Tick timer for OS support. A 24-bit timer with a dedicated exception type is included in the architecture, which the OS can use as a tick timer or as a general timer in other applications without an OS.
- SuperVisor Call (SVC) instruction with a dedicated SVC exception and Pendable SuperVisor service (PendSV) to support various operations in an embedded OS.
- Architecturally defined sleep modes and instructions to enter sleep. The sleep features allow power consumption to be reduced dramatically. Defining sleep modes as an architectural feature makes porting of software easier because the sleep modes are entered by specific instructions rather than implementation defined control registers.
- Fault handling exception to catch various sources of errors in the system.
- Support for 21 interrupts.
- Little endian memory support.
- Wake-up Interrupt Controller (WIC) to allow the processor to be powered down during sleep, while interrupt sources are still allowed to wake up the system.
- Halt mode debug allows the processor activity to stop completely so that register values can be accessed and modified. No overhead in code size and stack memory size.
- CoreSight technology allows memories and peripherals to be accessed from the debugger without halting the processor.
- Supports Serial Wire Debug (SWD) connections. The SWD protocol can handle the same debug features as the JTAG, but it only requires two wires and is already supported by a number of debug solutions from various tools vendors.
- Four (4) hardware breakpoints and two (2) watch points.
- Breakpoint instruction support for an unlimited number of software breakpoints.
- The programmer's model is similar to the ARM7TDMI processor. Most existing Thumb code for the ARM7TDMI processor can be reused. This also makes it easy for ARM7TDMI users, as there is no need to learn a new instruction set.

6.2 Architecture

6.2.1 Interrupts

This section lists all 21 interrupt lines, except the NMI interrupt, and describes their sources and functionality. Table 38 shows the overview of the interrupts.

Table 38: Interrupt list

IRQ number (inherent priority)	IRQ name	Description
0	BLE_WAKEUP_LP_IRQn	Wake up the system from Low Power (Extended Sleep) interrupt from Bluetooth LE.
1	BLE_GEN_IRQn	Bluetooth LE Interrupt. Sources: ▪ BLE_FINETGTIM_IRQn: Fine Target Timer interrupt generated when Fine Target timer expires. The timer resolution is 625 μs base time reference. ▪ BLE_GROSSTGTIM_IRQn: Gross Target Timer interrupt generated when Gross Target timer expired. The timer resolution is 16 times 625 μs base time reference. ▪ BLE_CSCNT_IRQn: 625 μs base time reference interrupt, available in active modes. ▪ BLE_SLP_IRQn: End of Sleep mode interrupt. ▪ BLE_ERROR_IRQn: Error interrupt, generated when undesired behavior or bad programming occurs in the Bluetooth LE Core.

IRQ number (inherent priority)	IRQ name	Description
		▪ BLE_RX_IRQn: Receipt interrupt at the end of each received packets. ▪ BLE_EVENT_IRQn: End of Advertising/Scanning/Connection events interrupt. ▪ BLE_CRYPT_IRQn: Encryption/Decryption interrupt, generated when AES and/or CCM processing is finished. ▪ BLE_SW_IRQn: Software triggered interrupt, generated on software request.
2	UART_IRQn	UART interrupt.
3	UART2_IRQn	UART2 interrupt.
4	I2C_IRQn	I2C interrupt.
5	SPI_IRQn	SPI interrupt.
6	ADC_IRQn	Analog-Digital Converter interrupt.
7	KEYBRD_IRQn	Keyboard interrupt.
8	BLE_RF_DIAG_IRQn	Baseband or Radio Diagnostics Interrupt. Triggered by internal events of the Radio or Baseband selected by the BLE_RF_DIAGIRQ_REG. For Debug purposes only.
9	RF_CAL_IRQn	RF Calibration Interrupt.
10	GPIO0_IRQn	GPIO interrupt through debounce.
11	GPIO1_IRQn	GPIO interrupt through debounce.
12	GPIO2_IRQn	GPIO interrupt through debounce.
13	GPIO3_IRQn	GPIO interrupt through debounce.
14	GPIO4_IRQn	GPIO interrupt through debounce.
15	SWTIM_IRQn	Timer0/2 interrupt.
16	WKUP_QUADEC_IRQn	Combines the Wake-up Capture Timer interrupt, the GPIO interrupt, and the QuadDecoder interrupt.
17	TIM1_IRQn	Timer1 interrupt.
18	RTC_IRQn	Real Time Clock interrupt.
19	DMA_IRQn	DMA interrupt.
20	XTAL32RDY_IRQn	XTAL32M settling ready interrupt.

Interrupt priorities are programmable by the Arm Cortex-M0+. The lower the priority number, the higher the priority level. The priority level is stored in a byte-wide register, which is set to 0x0 at reset. Interrupts with the same priority level follow a fixed priority order using the interrupt number listed in Table 38 (a lower interrupt number has a higher priority level).

To access the Cortex-M0+ NVIC registers, the Cortex Microcontroller Software Interface Standard (CMSIS) functions can be used. The input parameter IRQn of the CMSIS NVIC access functions is the IRQ number. This can be the IRQ number or (more conveniently) the corresponding IRQ name listed in Table 38. For example, the corresponding interrupt handler name in the vector table for IRQ#15 is SPI_Handler. For more information on the Arm Cortex-M0+ interrupts and the corresponding CMSIS functions, see *Section 4.2 Nested Vectored Interrupt Controller* in the *Cortex-M0+ Devices Generic User Guide*.

The Watchdog interrupt is connected to the NMI input of the processor.

6.2.2 System Timer (SysTick)

The Cortex-M0+ System Timer (SysTick) can be configured for using two different clocks. The SysTick Control and Status (STCSR) register specifies which clock should be used by the counter.

- STCSR[CLKSOURCE] = 0: use the (fixed) external reference clock STCLKEN of 1 MHz.
- STCSR[CLKSOURCE] = 1: use the (HCLK_DIV dependent) processor clock SCLK (for example, 2, 4, 8, or 16 MHz).

The default SysTick Timer configuration uses the (fixed) external reference clock STCLKEN (STCSR[CLKSOURCE] = 0). When necessary, higher clock frequencies can be used with STCSR[CLKSOURCE] = 1, but the software should take the HCLK_DIV dependent core clock SCLK into account about the timing.

6.2.3 Wake-up Interrupt Controller

The Wake-up Interrupt Controller (WIC) is a peripheral that can detect an interrupt and wake the processor from Extended Sleep mode. The WIC is enabled only when the SLEEPDEEP bit in the system control register is set to 1 (see *System Control Register* in the *Cortex-M0+ Technical Reference Manual*).

The WIC is not programmable and does not have any registers or user interface. It operates entirely from hardware signals. When the WIC is enabled and the processor enters Extended Sleep mode, the power management unit in the system can power down most of the Cortex-M0+ processor. This has the side effect of stopping the SysTick timer. When the WIC receives an interrupt, it takes a number of clock cycles to wake up the processor and restore its state before it can process the interrupt. This means the interrupt latency is increased in Extended Sleep mode.

6.3 Programming

For more information on the Arm Cortex-M0+, see the documents in Table 39.

Table 39: Arm documents list

	Document title	Arm document number
1	Cortex-M0+ Devices Generic User Guide	Arm DUI 0662B (available on the website)
2	Cortex-M0+ Technical Reference Manual, r0p1	Arm DDI 0484C (available on the website)
3	Armv6-M Architecture Reference Manual	Arm DDI 0419C (can be downloaded by registered customers)

7. AMBA Bus

7.1 Introduction

The DA14535 is based on the AMBA 2.0 AHB and APB components. The AHB is an AMBA Lite version which requires a single master on the system, but there is arbitration between the Arm Cortex-M0+ CPU and the Direct Memory Access (DMA) engine. There are two APB bridges, one for APB16 and the other for APB32, implementing three different decoded slaves which are grouped according to the power domain structure of the chip.

Figure 21 shows the AMBA bus organization.

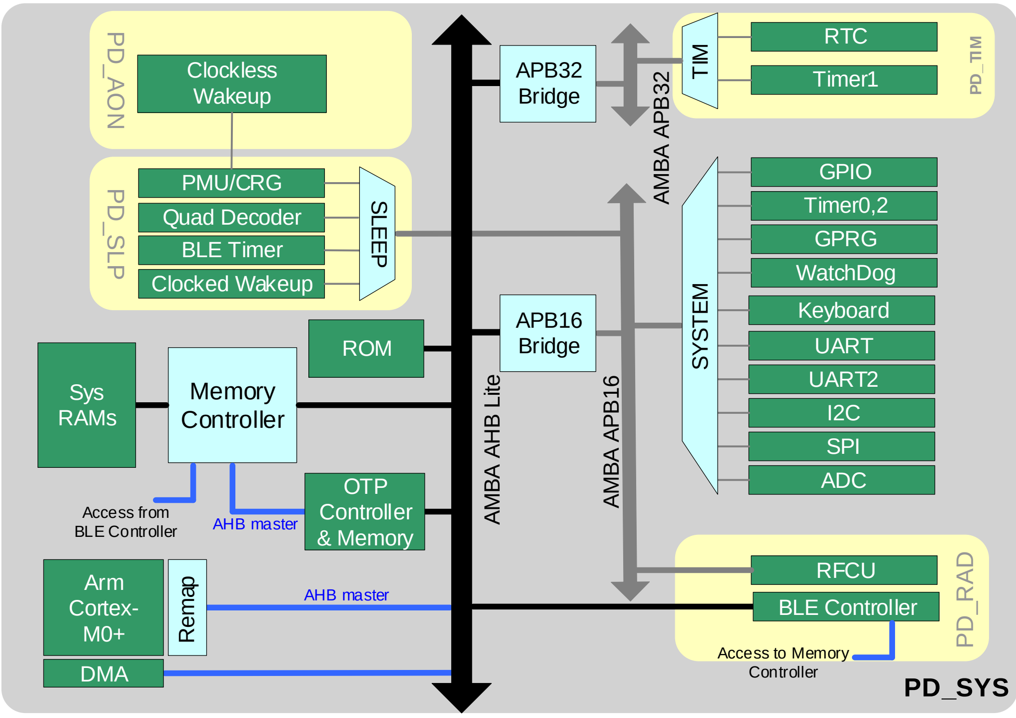

Figure 21. AMBA bus architecture and power domains

7.2 Architecture

Because DA14535 consists of several different power domains that are digitally controlled and can be shut down completely, various slave resources, especially on the APB bus, are grouped together to reduce signal isolation requirements. On the AHB Lite bus, the CPU or the DMA can be the master, while OTP, Bluetooth® LE Core, Memory and ROM controllers are slaves.

The Always On power domain (PD_AON) contains only the clock-less wake-up controller and the start-up hardware FSM responsible for the activation of the power devices within the system.

The sleep power domain (PD_SLP) contains the clock tree, the Bluetooth® LE Timer, the Clocked Wake-up Controller, and the Quadrature Decoder. These blocks are supposed to trigger or to capture wake-up events while the system is in any of the clocked sleep modes.

The timers power domain (PD_TIM) contains special purpose timers that might or might not be crucial for an application: a full featured Real Time Clock (RTC) engine and Timer1. The registers of these blocks are 32-bit wide, hence they are connected to the APB32 bus.

The APB16 bus connects to the radio power domain (PD_RAD), which consists of the Radio control unit and the Bluetooth® LE controller, and to the peripheral blocks which are all part of the same power domain as the CPU (PD_SYS).

7.3 Programming

Because the AMBA Bus only acknowledges a single master at a time, a programmable arbitration is implemented to decide whether the Arm Cortex-M0+ or the DMA is the master. The priority can be configured in the GP_CONTROL_REG[CPU_DMA_BUS_PRIO] with the CPU having the highest priority by default.

8. Memory Map

Table 40: Memory map

Address	Description	Power domain
0x00000000 0x04000000	**Boot/BLE ROM/OTP/RAM** Remapped address space based on SYS_CTRL_REG[REMAP_ADR0].	
0x04000000 0x07F00000	**RESERVED**	
0x07F00000 0x07F28000	**Boot/BLE ROM** Contains Boot ROM code and Bluetooth LE protocol related code.	
0x07F28000 0x07F40000	**RESERVED**	
0x07F40000 0x07F40100	**OTP-Regs** Contains the control registers of the OTP Subsystem.	PD_SYS
0x07F40100 0x07F80000	**RESERVED**	
0x07F80000 0x07F83000	**OTP** Contains the OTP cell actual memory space.	
0x07F83000 0x07FC0000	**RESERVED**	
0x07FC0000 0x07FD0000	**System RAM** 64 kB. Contains application code, data for the application, stack, and heap. SysRAM1 (32 kB): 0x07FC0000 to 0x07FC7FFF SysRAM2 (32 kB): 0x07FC8000 to 0x07FCFFFF	
0x07FD0000 0x40000000	**RESERVED**	
0x40000000 0x40001000	**AHB/BLE-Regs** Contains the control registers of the BLE Link Layer Processor.	PD_RAD
0x40001000 0x40004000	**AHB/Radio**	PD_RAD
0x40004000 0x50000000	**RESERVED**	
0x50000000 0x50000100	**APB16/PMU-CRG** Contains the control registers of the Power Management Unit and the Clock Generator.	PD_SLP
0x50000100 0x50000200	**APB16/Wake-up** Contains the registers of the clocked and clock-less wake up controllers.	PD_SLP
0x50000200 0x50000300	**APB16/Quadrature Decoder** Contains Logic that implements a step counter for X and Y axis from a rotary encoder.	PD_SLP
0x50000300 0x50001000	**RESERVED**	
0x50001000 0x50001100	**APB16/UART** Contains the control registers of the UART.	PD_SYS
0x50001100 0x50001200	**APB16/UART2** Contains the control registers of the UART2.	PD_SYS

Address	Description	Power domain
0x50001200 0x50001300	**APB16/SPI** Contains the control registers of the SPI interface.	PD_SYS
0x50001300 0x50001400	**APB16/I2C** Contains the control registers of the I2C interface.	PD_SYS
0x50001400 0x50001500	**APB16/Kbrd** Contains the registers of the Keyboard controller.	PD_SYS
0x50001500 0x50001600	**APB16/ADC** Contains the registers of the 4-channel ADC.	PD_SYS
0x50001600 0x50001700	**APB16/AnaMisc** Contains registers for various analog blocks.	PD_SYS
0x50001700 0x50003000	**RESERVED**	
0x50003000 0x50003100	**APB16/Ports** Contains the mode and direction registers of the GPIOs.	PD_SYS
0x50003100 0x50003200	**APB16/Watchdog** Contains the control registers of the Watchdog timer.	PD_SYS
0x50003200 0x50003300	**APB16/Version** Contains the version/revision of the chip.	PD_SYS
0x50003300 0x50003400	**APB16/Gen Purpose** Contains general purpose control registers.	PD_SYS
0x50003400 0x50003500	**APB16/Timer** Contains the control registers of Timer0 and Timer2.	PD_SYS
0x50003500 0x50003600	**APB16/RF Monitor** Contains the control registers of the RFMON.	PD_SYS
0x50003600 0x50003700	**APB16/DMA** Contains the control registers of the DMA.	PD_SYS
0x50003700 0x50004000	**RESERVED**	
0x50004000 0x50004100	**APB32/Timer1** Contains the control registers of Timer1.	PD_TIM
0x50004100 0x50004200	**APB32/RTC** Contains the control registers of the Real Time Clock.	PD_TIM
0x50004200 0xE0000000	**RESERVED**	
0xE0000000 0xE0100000	**Internal Private Bus** Contains various registers of the Arm Cortex-M0+.	PD_SYS

9. Memory Controller

9.1 Introduction

The Memory Controller of the DA14535 is responsible for the interface between the memory cells and the masters of the system that request access. It comprises two arbiters which use a fixed priority level scheme to allow parallelization among the three main masters of the RAM. The memory controller also provides the actual physical sequence of the RAM cells in a continuous memory space to enable the activation of the required amount of SysRAM only to save power.

Figure 22 shows the block diagram.

Features

- Two different RAM cells with retention capability (2 x 32 kB)
- Arbitration among the AHB masters (CPU or DMAs), OTP, and the Bluetooth LE core
- Transparently interfaces the AHB buses to memory signaling
- Fixed arbitration algorithm with time sharing.

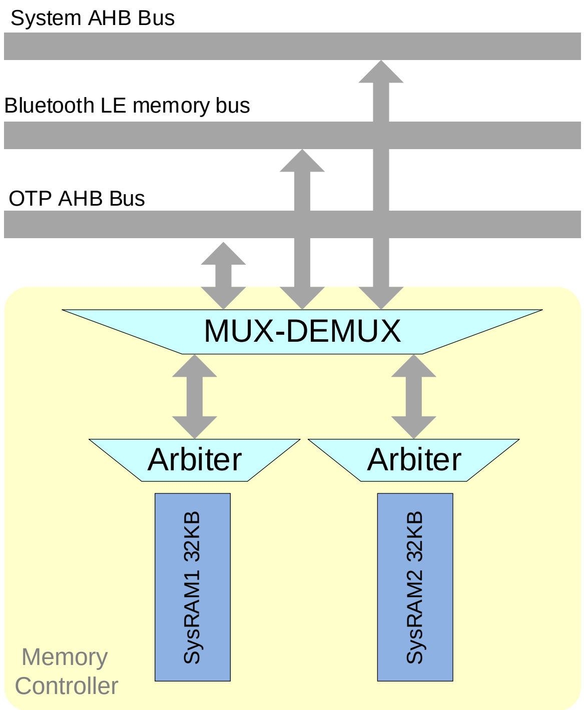

Figure 22. Memory controller block diagram

9.2 Architecture

The Memory Controller contains two arbiters that connect to the following buses via a Mux-Demux:

- Bluetooth® LE Mem I/F: this is a memory interface directly from the Bluetooth® 5.3 Core to the RAM used as an exchange memory (TX/RX descriptors and others). This interface always operates at 16 MHz.
- System Mem I/F: This is a memory interface directly from the OTP memory to the RAM used for copying data after powerup/wake-up.

9.2.1 Arbitration

Arbitration is a mixture of the highest priority and a fair use policy. If more than one master request access to cells which reside under the same arbiter, time division is employed. This is to make sure none of the buses can stall the others for a long period. The OTP and Bluetooth® LE accesses are handled as very critical and therefore they have the highest priority.

10. Clock Generation

10.1 Clock Tree

Figure 23 shows the generation of the system's clocks in detail.

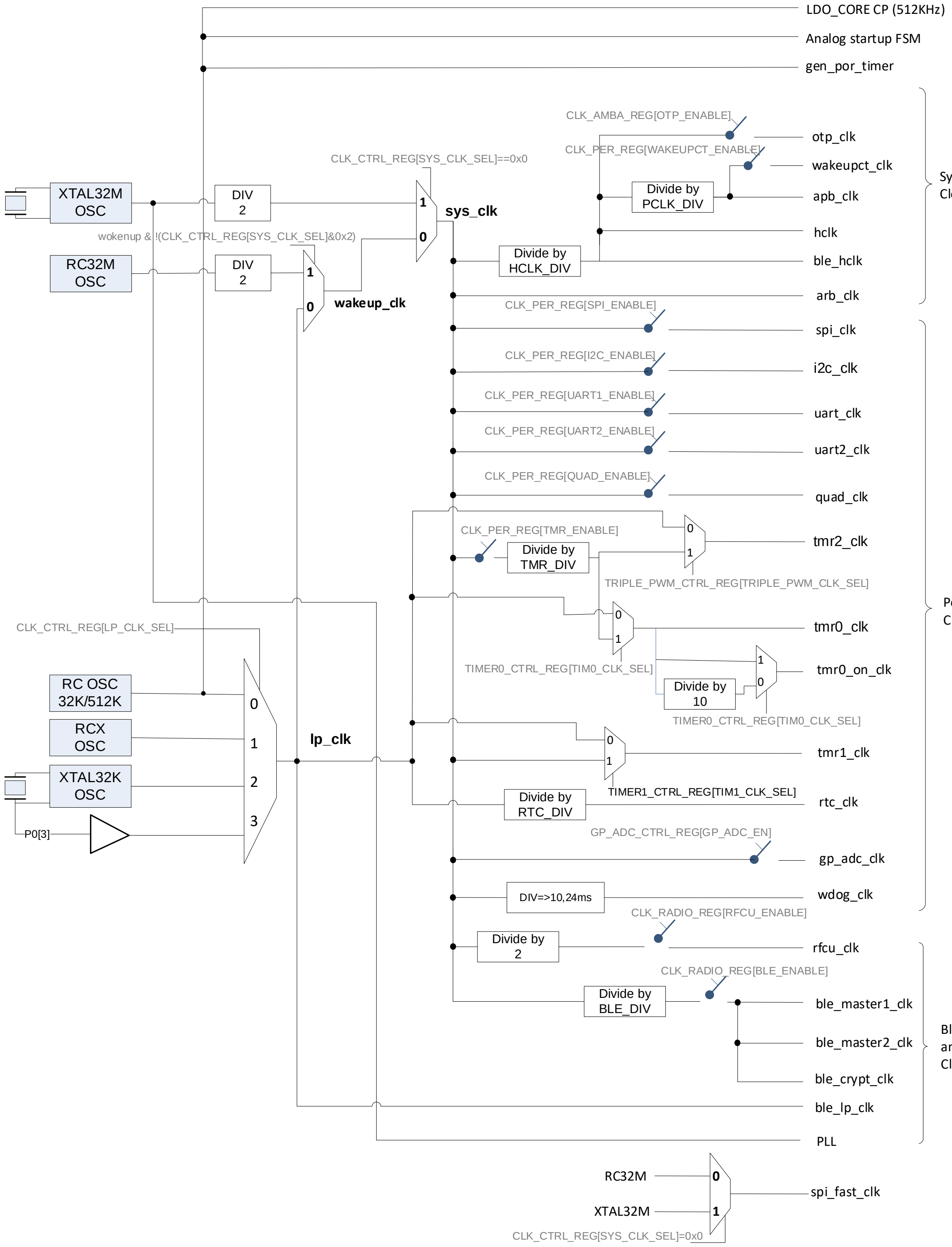

Figure 23. Clock tree diagram

Figure 23 shows the possible clock sources as well as all different divisions and multiplexing paths towards the generation of each block's clock. Furthermore, the required registers that have to be programmed are also shown in Figure 23.

Internal clock sources of the DA14535 are the RC32M, RC32K/512K, and RCX oscillators. External clock sources of DA14535 are the 32 MHz crystal oscillator (the pins XTAL32Mp and XTAL32Mm), the 32.768 kHz crystal oscillator (the pins XTAL32kp and XTAL32km mapped on P0_3 and P0_4, respectively), or an external digital clock (the pin P0_3).

There are two main clock lines which are of interest:

- lp_clk: this is the low power clock used for sleep modes and can only be either the RCX, the RC32K, the XTAL32K, or an externally supplied digital clock.
- sys_clk: this is the system clock used for the AMBA clock (hclk), which runs the CPU, the memories, and the bus. This clock source can be one of the oscillators or an externally supplied digital clock.

The clock names shown in Figure 23 are explained in Table 41.

Table 41: Generated clocks description

Clock name	Description
wakeupct_clk	Clocked wake-up controller clock.
apb_clk	AMBA APB interface clock.
otp_clk	OTP controller clock.
hclk	AMBA AHB interface clock.
ble_hclk	AMBA AHB clock for the Bluetooth® LE core.
wdog_clk	Watchdog clock.
pmu_rc_clk	Clock for the PMU and analog start-up FSM.
icp_clk_512_c (512KHz)	Clock for the Charge pump in the LDO_CORE.
gen_por_timer	Clock for the POR_FORCE Timer.
spi_clk	Clock for the SPI controller. This clock is further divided by 2, 4, 8, or 14 as defined by SPI_CTRL_REG[SPI_CLK].
spi_fast_clk	Fast 32 MHz clock for the SPI controller.
i2c_clk	Clock for the I2C controller. This clock is further divided to provide 100 kHz or 400 kHz as defined by I2C_CON_REG[I2C_SPEED].
uart_clk	Clock for the UART.
uart2_clk	Clock for the UART2.
quad_clk	Clock for the quadrature decoders.
rfcu_clk	Clock for the RF control unit of the Radio.
tmr0_clk, tmr2_clk	Timer0/2 clocks.
tmr1_clk	Timer1 clock.
tmr0_on_clk	Timer0 ON counter clock.
rtc_clk	Clock for Real Time Clock (RTC).
gp_adc_clk	General Purpose ADC conversion clock.
ble_crypt_clk	Clock for the Crypto block of the Bluetooth® LE core.
ble_master1_clk	Internal clock for the Bluetooth® LE core.
ble_master2_clk	Internal clock for the Bluetooth® LE core.
arb_clk	Clock for the memory controller arbiter.
ble_lp_clk	Bluetooth® LE core low power clock.
pll	PLL clock.

10.1.1 General Clock Constraints

There are certain constraints on various clocks regarding their frequency relations or the effectiveness. This section summarizes these rules:

- The minimum of the AMBA clock (hclk) should be 8 MHz when Bluetooth® LE is utilized. This is also the clock of the Cortex CPU and ensures the required MIPS for handling the Bluetooth® LE Protocol.
- The AMBA clock (hclk) should always be greater or equal to the ble_*_clks. This is required for the proper operation of the Bluetooth® LE protocol. For example, hclk at 16 MHz and Bluetooth® LE clocks at 8 MHz is an acceptable combination but not the other way around.

10.2 Crystal Oscillators

The Digital Controlled XTAL Oscillators (DXCO) are designed for low power consumption and high stability. There are two such crystal oscillators in the system, one at 32 MHz (XTAL32M) and the other at 32.768 kHz (XTAL32K). The XTAL32K has no trimming capabilities and is used as the clock of the Deep Sleep/Extended Sleep modes. The XTAL32M can be trimmed.

The principal schematic of the two oscillators is shown in Figure 24. No external components to DA14535 are required other than the crystal itself. If the crystal has a case connection, it is advised to connect the case to ground.

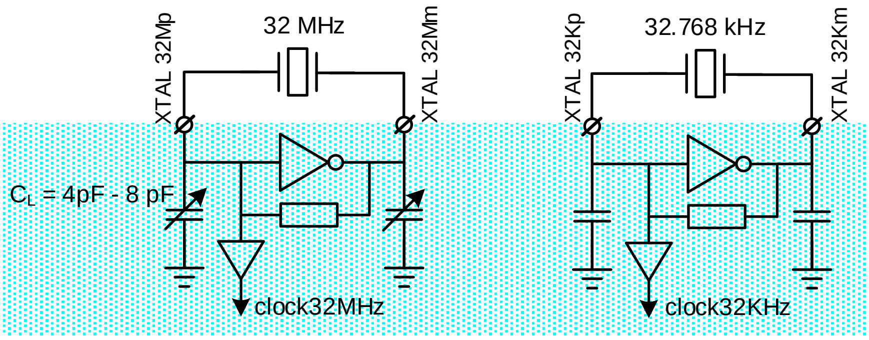

Figure 24. Crystal oscillator circuits

10.2.1 Frequency Control (32 MHz Crystal)

The 8-bit register CLK_FREQ_TRIM_REG controls the trimming of the 32 MHz crystal oscillator. The frequency is trimmed by two on-chip variable capacitor banks. Both capacitor banks are controlled by the same register.

The capacitance of both variable capacitor banks varies from the minimum to the maximum value in 256 equal steps. With CLK_FREQ_TRIM_REG[XTAL32M_TRIM] = 0x00, the minimum capacitance and thus the maximum frequency are selected. With CLK_FREQ_TRIM_REG[XTAL32M_TRIM] = 0xFF, the maximum capacitance and thus the minimum frequency are selected.

The five least significant bits of CLK_FREQ_TRIM_REG register (XTAL32M_TRIM<4:0>) directly control five binary weighted capacitors (Figure 25). The three most significant bits of CLK_FREQ_TRIM_REG register (XTAL32M_TRIM<7:5>) are binary to the thermometer decoded. Each of the seven outputs of the decoder controls a capacitor, of which the value is 32 times the value of the smallest capacitor.

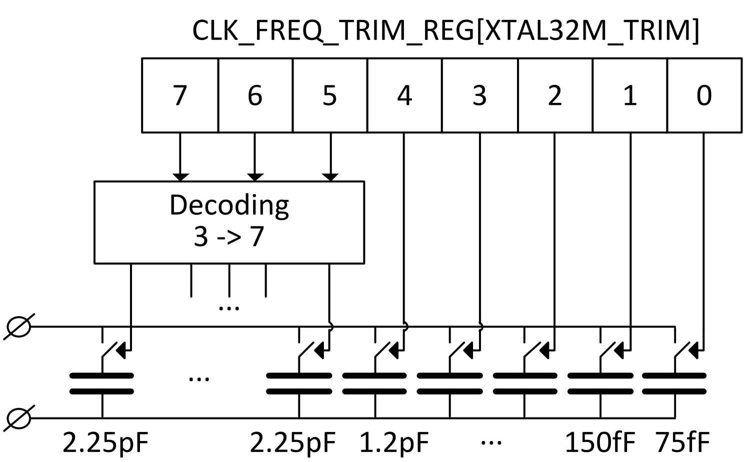

Figure 25. XTAL32MHz oscillator frequency trimming

10.2.2 Automated Trimming and Settling Notification

There is provision in DA14535 for automating the actual trimming of the 32 MHz crystal oscillator. This is a special hardware block that realizes the XTAL trimming in a single step. Notification about the XTAL oscillator being settled after applying the trim value is also provided in form of an interrupt, namely, the XTAL32RDY_IRQn line. The automated mechanism for applying the trim value and signaling that the oscillator is settled is described in Figure 26.

The XTAL32RDY_IRQn is always triggered as soon as an internal counter reaches the value programmed at XTALRDY_CTRL_REG. This counter runs on the RC32M clock if the system is powering up, or on a selected low power clock if the system is waking up. The enabling of the XTAL32M is always done by hardware. There are two sections until the interrupt notifies the software that the XTAL32M can be used:

- The start-up section, where the XTAL32M oscillator is slowly converging towards the initial frequency of the crystal. This section ends with the application of the trim value to achieve a <50 ppm, 32 MHz clock.
- The settling section, where the XTAL32M oscillator settles to the preferred frequency after the application of the trim value which is done automatically by hardware.

There are two ways of deciding when the start-up section ends and when the trim values are supposed to be applied. This decision is controlled by TRIM_CTRL_REG[XTAL_TRIM_SELECT] bit field:

- **Counter mode:** trim value stored in the CLK_FREQ_REG is applied as soon as an internal counter reaches the value XTAL_COUNT_N-1. This is the default mode.
- **Current mode:** trim value is applied as soon as the current drops.

The different modes are shown in Figure 26.

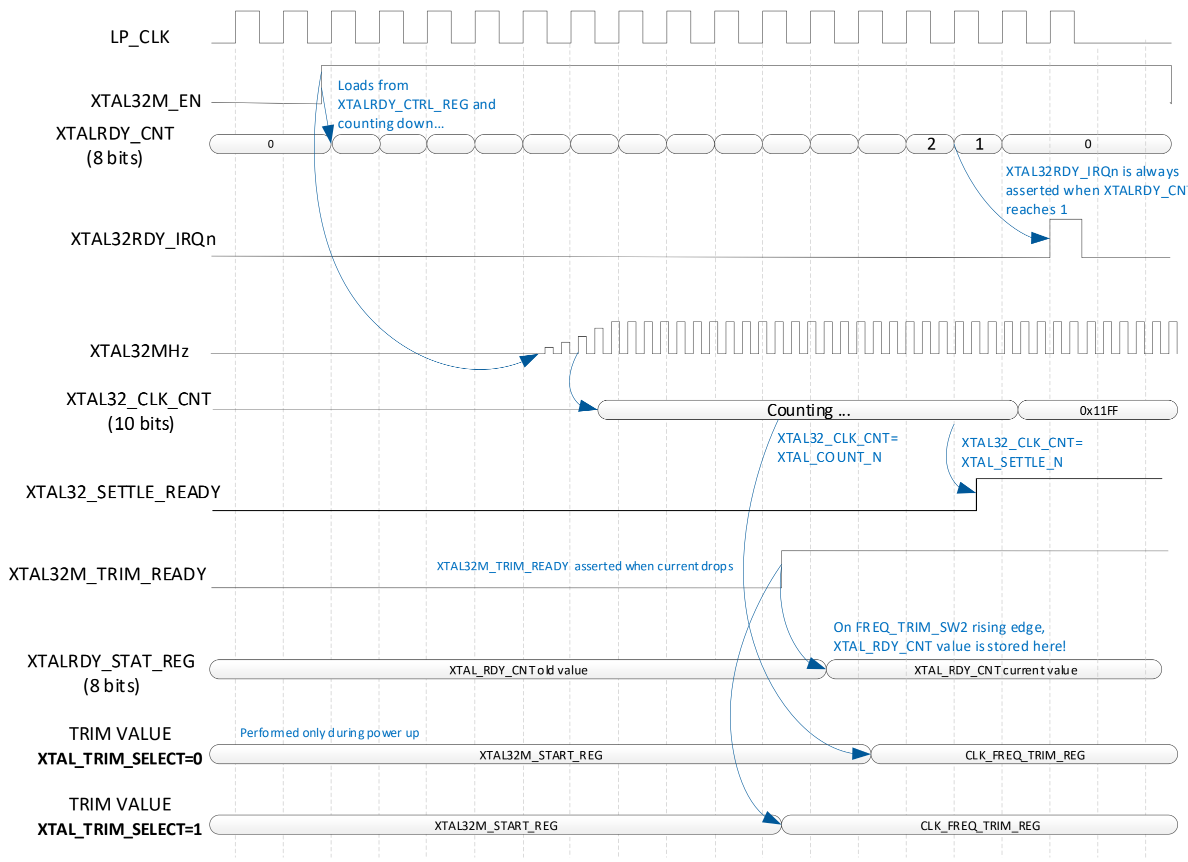

Figure 26. Automated mechanism for XTAL32M trim and settling

In both modes mentioned above, trimming is done by hardware. In the current mode, upon assertion of XTAL32M_TRIM_READY, the interrupt counter value is stored in a shadow register XTALRDY_STAT_REG to enable software understanding when the start-up section is finished.

The settling section usually takes no more than five to 10 clock cycles. Using the explanation above, fine tuning and reducing the XTAL32M latency is feasible. One feature of the XTAL32_CLK_CNT is that it asserts an observable signal (SYS_STAT_REG[XTAL32_SETTLE_READY]) as soon as the counter reaches a pre-defined threshold programmed at TRIM_CTRL_REG[XTAL_SETTLE_N]. This allows the software to have an indication of the status of the clock by adjusting the threshold accordingly.

10.3 RC Oscillators

There are three RC oscillators in DA14535:

- One providing 32 MHz (RC32M)
- One providing 32 kHz and 512 kHz (RC32K/512K)
- One providing a frequency of 15 kHz (RCX).

The RC32M is powered by V_{BAT_LOW} which is available during Active or Sleep mode. The output clock is slower than 32 MHz if untrimmed and it is used to clock the CPU and the digital part of the chip during powerup or wake-up, while the XTAL32M oscillator is settling.

The simple RC oscillator (RC32K/512K) operates on VDD and provides 32 kHz or 512 kHz. The main usage of the RC32K/512K oscillator is for internal clocking during powerup or start-up. It clocks the hardware FSM which brings up the power management system of the chip. In the powerup or start-up sequence, the clock dynamically changes from 32 kHz to 512 kHz to speed up the sequence. The enhanced RC oscillator (RCX) provides a stable 15 kHz frequency and operates on V_{BAT_LOW} which is available during Active or Sleep mode.

The RCX oscillator can be used to replace the 32.768 kHz crystal, because it has a precision of < 500 ppm, while its output frequency needs to be recalibrated over temperature.

Using the RCX requires the following registers to be set:

- Set GP_DATA_REG = 0x20 after the system wakes up.
- RCX calibration (the calibration is optional, see Section 10.3.1 for the RCX calibration).
- Go to sleep: set GP_DATA_REG = 0x40 after the sleep procedure is handled.

The procedure is also implemented as a part of the SDK.

10.3.1 Frequency Calibration

The output frequency of the 32 kHz crystal oscillator and the three RC-oscillators can be measured relative to the 32/2 (16) MHz crystal oscillator using the on-chip reference counter.

The measurement procedure is as follows:

1. REF_CNT_VAL = N (the larger number N is, the more accurate and longer the calibration is).
2. CLK_REF_SEL_REG = 0x0000 (RC32K) or
   CLK_REF_SEL_REG = 0x0001 (RC32M) or
   CLK_REF_SEL_REG = 0x0002 (XTAL32K) or
   CLK_REF_SEL_REG = 0x0003 (RCX)
3. Start the calibration: CLK_REF_SEL_REG[REF_CAL_START] = 1.
4. Wait until CLK_REF_SEL_REG[REF_CAL_START] = 0.
5. Read CLK_REF_VAL_H_REG and CLK_REF_VAL_L_REG = M (32-bit values).
6. Frequency = (N/M) × 32/2 MHz.

If the RCX is used as a sleep clock, the frequency calibration should be implemented on each active time of a connection interval to guarantee a correct operation.

11. OTP Controller

11.1 Introduction

The OTP controller realizes all functions of the OTP macro cell in an automated and transparent way. The controller facilitates all data transfers (reading and programming), comprises a DMA engine, which connects to the AHB bus as a master, and has the highest priority to copy code from OTP into SysRAM in mirrored mode. The block diagram is shown in Figure 27.

Features

- Implements all timing constraints for any access to the physical OTP cell.
- Automatic single Error Code Correction (ECC) – 6 bits (implemented in the OTP cell).
- 32-bit read in a single read access from the OTP cell.
- Single word buffer for programming. No burst programming supported.
- Empty words are 0xFFFFFFFF. Zeros are programmed per 32-bit word.
- Embedded DMA engine for fast mirroring of the OTP contents into the SysRAM.
- Embedded DMA supports reading in bursts of 4×32-bit words.
- Transparent random address access to the OTP memory cells through the AHB slave memory interface.
- Hardwired handshaking with the PMU to realize the mirroring procedure.

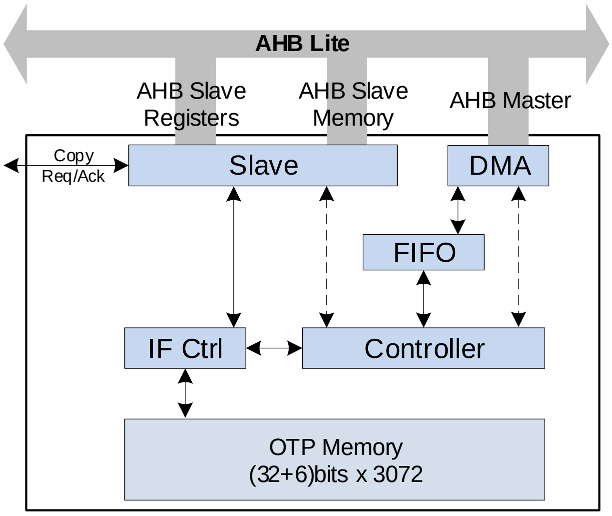

Figure 27. OTP controller block diagram

11.2 Architecture

The OTP controller block includes the OTP macro cell and pure digital logic implementing the controlling functions. The OTP memory communicates with the controller through a proprietary interface.

The internal organization of the OTP cell is 32-bit data and 6-bit ECC for each of the 3072 addressable positions. The six bits of the ECC are only accessible within the OTP cell. The ECC is generated by the OTP cell during programming and is used again by the OTP cell in a transparent way during reading.

The AHB master interface is controlled by a DMA engine with an internal FIFO of eight 32-bit words. The DMA engine supports AHB reads and writes. The AHB address, where memory access should begin, is programmed into the DMA engine at OTPC_AHBADR_REG[OTPC_AHBADR]. The number of the 32-bit words of a transfer minus 1 must be specified in OTPC_NWORDS_REG[OTP_NWORDS].

The DMA engine internally supports the following burst types:

- Eight words incremental burst (INCR8).
- Four words incremental burst (INCR4).
- Unspecified incremental burst (INCR) with a length different from 1, 4, or 8.

- Single word access (SINGLE).

The slave block combines two AHB slave interfaces: one is for the registers and can be read from/written to, and the other is for the contents of the OTP memory and is read-only.

The OTP controller configures the OTP cell to be in one of the following modes:

- **Deep Stand-by mode (DSTBY).** In this mode, the required power supplies are applied to the OTP cell, while the internal LDO of the OTP cell is inactive.
- **Stand-by mode (STBY).** In this mode, the OTP cell is disabled by deactivating the chip select signal. The OTP cell is powered and the internal LDO is enabled. The power consumption of the OTP cell in this mode is not the minimum possible but is less than in an active mode (RD, PROG, PVFY, RINI, AREAD). This is the state from which any active mode of operation can be transitioned into with the least delay.
- **Read mode (RD).** When this mode is used, the contents of the OTP cell can be read at the respective AHB address space. This mode can also be used to execute software in place (XIP). A read request is translated by the OTP controller into the corresponding control sequence for the OTP cell to retrieve the requested data.
- **Programming mode (PROG).** The PROG mode provides the functionality to program a 32-bit word into an OTP position. The OTP cell expands the 32-bit word by calculating and automatically appending a 6-bit checksum (ECC). Note that there is no way to access these extra six bits of the ECC information. Programming is performed only for bits equal to 0. Bits equal to 1 are bypassed to save programming time. Because the ECC value is unknown to the controller, there are always six extra programming pulses applied to the ECC bits. Programming is done by issuing a programming request stored in the Programming Buffer (PBUF). The PBUF consists of two configuration registers storing the 32-bit data value and the 13-bit address in the OTP cell where the value should be programmed. A new request can only be stored in the PBUF when the previous is served. A status bit indicates whether this has already been done, therefore programming should be monitored by software before a new programming request is issued.
- **Programming Verification mode (PVFY).** The PVFY mode forces the OTP cell to enter a special margin read mode. This mode is used to verify the content of the OTP positions that have been programmed using the PROG mode and to verify that the programmed data is retrieved correctly under all corner cases. When this mode is used, the contents of the OTP cell can be read at the respective AHB address space. The CPU must read all OTP positions that have been programmed by accessing the corresponding addresses and verify that all the retrieved words are equal to the expected values.
- **Read Initial State mode (RINI).** The RINI mode implements a production test of the initial margin read, which should be performed in the OTP cell, before the first programming is applied. This test verifies that the OTP cell is empty (all the bits are equal to 1). The OTP controller sends the required control sequence to the OTP cell to enable the test mode. Then the CPU should read all the content of the OTP cell at the respective AHB address space and verify that all the retrieved words are equal to 0xFFFFFFFF. The RINI mode should be used after the PROG mode to verify the content of the OTP positions that have been programmed and specify the bits that remain unprogrammed. This verification is required to ensure that the programming process has not affected the unprogrammed bits. This specific read mode is a margin read, which means that it is not an equivalent to the normal read and should only be used for the purpose of verification.
- **Automatic Read mode (AREAD).** This mode is used to mirror large parts of the OTP cell into RAM through the AHB master interface and the integrated DMA controller.

Transitioning from one mode to another automatically steps through the STBY mode.

11.2.1 OTP Accessing Considerations

Accesses to the OTP memory (read/write) can only be performed at certain voltage ranges. You are responsible for meeting these conditions while accessing the OTP. The recommended operation conditions of the OTP memory can be found under the Recommended Operating Conditions section.

11.3 Programming

To configure the OTP controller:

1. Enable clock for OTP controller by setting the CLK_AMBA_REG[OTP_ENABLE] bit.
2. Put the OTP in STBY mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x1).
3. Wait for OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).

4. Set OTP speed by writing OTPC_TIM1_REG and OTPC_TIM2_REG if system clock speed is to be reduced. These numbers basically generate asynchronous timing signals towards the OTP cell that comply to the default internal 16 MHz bus speed.
5. Perform an OTP access:
    a. Programming:
        i. Set up OTP write mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x3).
        ii. Wait for OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
        iii. Check OTPC_STAT_REG[OTPC_STAT_PBUF_EMPTY] = 1.
        iv. Write the data to be programmed to OTPC_PWORD_REG.
        v. Write the address to which the data is to be programed to OTPC_PADDR_REG.
        vi. Wait until the programming is finished (OTPC_STAT_REG[OTPC_STAT_PRDY] = 1).
        vii. Switch to OTP verify mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x4).
        viii. Wait for OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
        ix. Read back and compare the data written.
        x. Put the OTP in STBY mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x1).
        xi. Wait for OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
    b. Reading:
        i. Set up OTP read mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x2).
        ii. Wait for OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
        iii. Read OTP word.
        iv. Put the OTP in STBY mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x1).
        v. Wait for OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).

12. DMA Controller

12.1 Introduction

The 4-channel direct memory access (DMA) controller transfers data of eight bits, 16 bits, or 32 bits between the on-chip supported peripherals (SPI, UART, UART2, I2C, and ADC) and the on-chip RAM and supports regular memory-to-memory transfers. The DMA also supports a programmable interrupt generation to generate an interrupt after a certain number of transfers to offload the Cortex interrupt rate. The on-chip peripheral requests are multiplexed on the two available channel pairs to increase the DMA utilization. Figure 24 shows a block diagram of the controller.

Features

- Four channels with an optional peripheral trigger.
- Full 32-bit source and destination pointers.
- Flexible interrupt generation.
- Programmable length.
- Flexible peripheral request per channel.
- Option to initialize memory (DMA_INIT).
- Programmable edge-sensitive request support (recommended when writing to UART/UART2 and I2C).

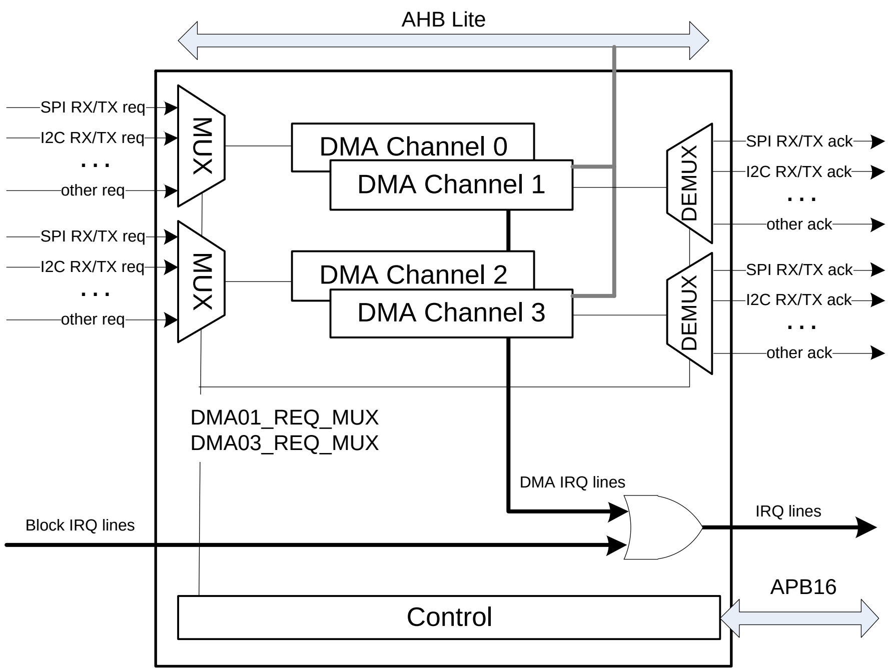

Figure 28. DMA controller block diagram

12.2 Architecture

12.2.1 DMA Peripherals

By default, the DMA assumes memory-to-memory transactions. Each DMA channel can also be connected with the hand-shaking signals or other request signals of the corresponding peripherals (Table 42).

Table 42: DMA served peripherals

Name	Direction
SPI	RX/TX
UART	RX/TX
UART2	RX/TX

Name	Direction
I2C	RX/TX
GP-ADC	RX

12.2.2 Input/Output Multiplexer

The multiplexing of peripheral requests is controlled by DMA_REQ_MUX_REG. Thus, if DMA_REQ_MUX_REG[DMAxy_SEL] is set to a certain (non-reserved) value, the TX/RX request from the corresponding peripheral is routed to DMA channels y (TX request) and x (RX request), respectively. Similarly, an acknowledging de-multiplexing mechanism is applied.

When two or more bit-fields (peripheral selectors) of DMA_REQ_MUX_REG have the same value, the lesser significant selector is given priority (see also the register's description).

12.2.3 DMA Channel Operation

A DMA channel is switched on with bit DMA_ON. This bit is automatically reset if the DMA channel's transfer is finished. The DMA channels can either be triggered by software or by a peripheral DMA request. If DREQ_MODE is 0, a DMA channel is immediately triggered.

If DREQ_MODE is 1, a DMA channel can be triggered by a hardware request coming from a selected peripheral. All DMA channels support either level (default) or edge-sensitive requests through the bit-field REQ_SENSE of DMAx_CTRL_REG (x = 0, 1, 2, 3). If this bit-field is set (recommended for Memory-to-UART/UART2 and Memory-to-I2C transfers), the channel detects a positive edge on the request signal of the selected peripheral to start up a new transfer cycle. The edge-sensitive requests can be used globally, if desired, for all the peripherals interfacing with the DMA.

When DMA starts, data is transferred from address DMAx_A_START_REG to address DMAx_B_START_REG for a length of DMAx_LEN_REG, which can be 8, 16, or 32 bits wide. The address increment is realized with an internal 16-bit counter DMAx_IDX_REG, which is set to 0 when the DMA transfer starts and is compared with the DMAx_LEN_REG after each transfer. The register value is multiplied by the values of the automatic increment of source address (AINC), the automatic increment of destination address (BINC), and bus transfer width (BW) before it is added to DMAx_A_START_REG and DMAx_B_START_REG. AINC or BINC must be 0 for register access.

If at the end of a DMA cycle, the DMA start condition is still true, the DMA continues. The DMA stops if DREQ_MODE is low or if DMAx_LEN_REG is equal to the internal index register. This condition also clears the DMA_ON bit if DREQ_MODE is 0 or if DREQ_MODE is set to 1 and CIRCULAR bit is not set.

If a hand shaking is attached to the specific DMA channel at the end of a DMA cycle, the channel is blocked for as long as the peripheral is not ready for the next transaction.

If the bit CIRCULAR is set to 1, the DMA controller automatically resets the internal index registers and continues from its starting address without intervention of the Arm Cortex-M0+. If the DMA controller is started with DREQ_MODE = 0, the DMA always stops, regardless of the state of CIRCULAR.

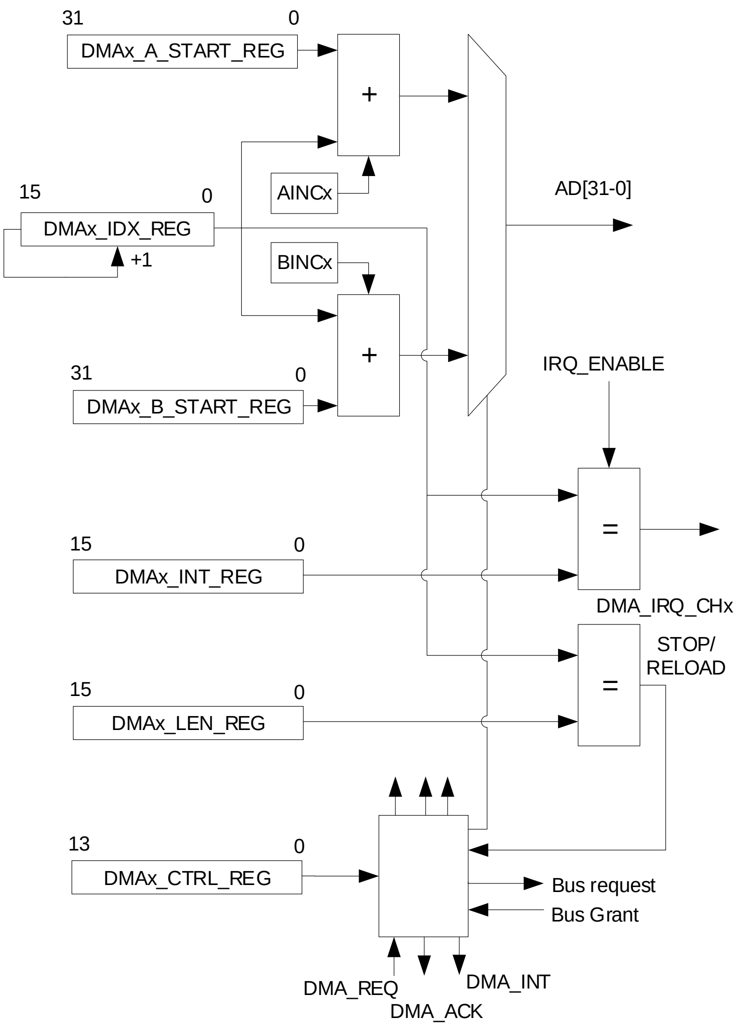

Figure 29. DMA channel diagram

Each DMA channel can generate an interrupt if the index counter DMAx_IDX_REG reaches the value of the channel's interrupt transfer length register, DMAx_INT_REG. After the transfer and before DMAx_IDX_REG is incremented, the interrupt is generated.

For example, if DMA_x_INT_REG = 0 and DMA_x_LEN_REG = 0, there is one transfer and an interrupt.

12.2.4 DMA Arbitration

The priority level of a DMA channel can be set with bits DMA_PRIO[2-0]. These bits determine which DMA channel is activated if more than one DMA channel requests DMA. If two or more channels have the same priority, an inherent priority applies (see register description).

With DREQ_MODE = 0, a DMA can be interrupted by a channel with a higher priority if the DMA_IDLE bit is set.

When DMA_INIT is set, however, the DMA channel currently performing the transfer locks the bus and cannot be interrupted by any other channels until the transfer is completed, regardless of whether DMA_IDLE is set. The purpose of DMA_INIT is to initialize a specific memory block with a certain value without any interruption from other active DMA channels that may request the bus at the same time. Consequently, DMA_INIT should be used only for memory initialization. When the DMA transfers data to/from peripherals, DMA_INIT should be set to 0.

NOTE
When DMA_INIT is enabled, AINC must be set to 0 and BINC to 1. Memory initialization could also be performed by simply setting AINC to 0 and BINC to 1 without enabling the DMA_INIT, provided that the source address of the memory will not change during the transfer. However, it is not guaranteed that the

NOTE
DMA transfer will not be interrupted by other channels of a higher priority when they request access to the bus at the same time.

12.2.5 Freezing DMA Channels

Each channel of the DMA controller can be temporarily disabled by writing a 1 to bit 4 SET_FREEZE_REG[FRZ_DMA] to freeze all channels.

To enable a frozen channel again, write a 1 to bit 4 RESET_FREEZE_REG[FRZ_DMA].

There is no hardware protection from erroneous programming of the DMA registers.

The on-going Memory-to-Memory transfers (DREQ_MODE = 0) cannot be interrupted, therefore the corresponding DMA channels are frozen after a Memory-to-Memory transfer is completed.

12.3 Programming

12.3.1 Memory to Memory Transfers

1. Set the length of data to be transferred (DMAx_LEN_REG).
2. Set the source address (DMAx_A_START_REG).
3. Set the destination address (DMAx_B_START_REG).
4. Configure the number of transfers until an interrupt is generated (DMAx_INT_REG).
5. Configure transfer options:
   a. DMAx_CTRL_REG[AINC]: Automatic increment of source address.
   b. DMAx_CTRL_REG[BINC]: Automatic increment of destination address.
   c. DMAx_CTRL_REG[BW]: Bus transfer width.
   d. DMAx_CTRL_REG[IRQ_ENABLE]: Enable the DMA interrupt generation for this channel.
6. Start the DMA transfer by setting the DMAx_CTRL_REG[DMA_ON] bit.
7. Wait until the transfer is finished (DMAx_CTRL_REG[DMA_ON] = 0).
8. Clear the IRQ status bit for channel x in DMA_INT_STATUS_REG.

12.3.2 Peripheral to Memory Transfers

1. Set the length of data to be transferred (DMAx_LEN_REG).
2. Set the source address (DMAx_A_START_REG) to the peripheral Rx register (for example, I2C_DATA_CMD_REG).
3. Set the destination address (DMAx_B_START_REG). This should point to a buffer in memory (for example, SYSRAM).
4. Configure the number of transfers until an interrupt is generated (DMAx_INT_REG).
5. Map the peripheral to the selected channels pair (DMA_REQ_MUX_REG[DMAxy_SEL]).
6. Configure transfer options:
   a. DMAx_CTRL_REG[AINC]: Disable automatic increment of source address.
   b. DMAx_CTRL_REG[BINC]: Automatic increment of destination address.
   c. DMAx_CTRL_REG[BW]: Bus transfer width.
   d. DMAx_CTRL_REG[DREQ_MODE]: Enable triggering by peripheral DMA request.
   e. DMAx_CTRL_REG[DMA_PRIO]: Set the channel's priority.
   f. DMAx_CTRL_REG[IRQ_ENABLE]: Enable the DMA interrupt generation for this channel.
   g. DMAx_CTRL_REG[REQ_SENSE]: Enable edge-sensitive requests for this channel. This is recommended for Memory-to-UART/UART2/I2C transfers but can also be used globally for all the supported peripherals and for both directions (TX/RX).
7. Start the DMA transfer by setting the DMAx_CTRL_REG[DMA_ON] bit.

8. Enable peripheral's DMA request (for example, I2C_DMA_CR_REG[TDMAE]).
9. Clear the IRQ status bit for channel x in DMA_INT_STATUS_REG.

13. I2C Interface

13.1 Introduction

The I2C Interface is a programmable control bus that provides support for the communications link between Integrated Circuits in a system. It is a simple 2-wire bus with a software-defined protocol for system control, which is used in temperature sensors and voltage level translators to EEPROMs, general-purpose I/O, and A/D and D/A converters.

Features

- 2-wire I2C serial interface consisting of a serial data line (SDA) and a serial clock (SCL).
- Two speeds are supported:
  - Standard mode (0 to 100 kbit/s).
  - Fast mode (≤ 400 kbit/s).
- Clock synchronization.
- 32 locations deep transmit/receive FIFOs (32× 8-bit RX and 32× 10-bit TX).
- Master transmit and Master receive operation.
- Slave transmit and Slave receive operation.
- 7-bit or 10-bit addressing.
- 7-bit or 10-bit combined format transfers.
- Bulk transmit mode.
- Default slave address of 0x055.
- Interrupt or polled-mode operation.
- Handles bit and byte waiting at both bus speeds.
- Programmable SDA hold time.
- DMA support.

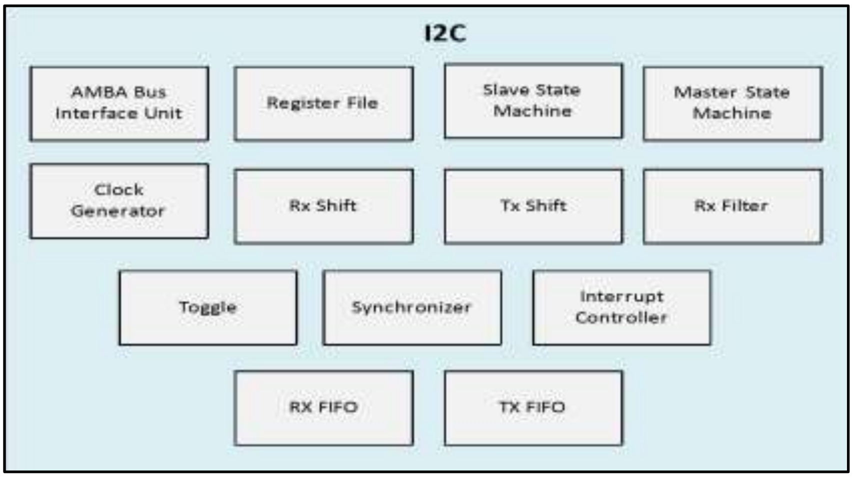

Figure 30. I2C controller block diagram

13.2 Architecture

The I2C Controller block diagram is shown in Figure 30 and contains the following subblocks:

- **AMBA Bus Interface Unit**: it accesses the register file through the APB interface.
- **Register File**: it contains configuration registers and is the interface with software.
- **Master State Machine**: it generates the I2C protocol for the master transfers.
- **Slave State Machine**: it generates the I2C protocol for the slave transfers.
- **Clock Generator**: it calculates the required time to do the following:

- Generate the SCL clock when configured as a master
- Check for bus idle
- Generate a START and a STOP
- Set up the data and hold the data.

- **RX Shift**: it takes data into the design and extracts it in byte format.
- **TX Shift**: it presents data supplied by CPU for transfer on the I2C bus.
- **RX Filter**: it detects the events in the bus, for example, start, stop, and arbitration lost.
- **Toggle**: it generates pulses on both sides and toggles to transfer signals across clock domains.
- **Synchronizer**: it transfers signals from one clock domain to another.
- **Interrupt Controller**: it generates the raw interrupt and interrupt flags, allowing them to be set and cleared.
- **RX FIFO/TX FIFO**: it holds the RX FIFO and TX FIFO register banks and controllers along with their status levels.

13.2.1 I2C Bus Terms

The following terms relate to what the role of an I2C device is and how it interacts with other I2C devices on the bus.

- **Transmitter** is the device that sends data to the bus. A transmitter can either initiate the data transmission to the bus (a master-transmitter) or respond to a request from the master to send data back (a slave-transmitter).
- **Receiver** is the device that receives data from the bus. A receiver can either receive data on its own request (a master-receiver) or respond to a request from the master to receive data (a slave-receiver).
- **Master** is the component that initializes a transfer (START command), generates the clock (SCL) signal, and terminates the transfer (STOP command). A master can be either a transmitter or a receiver.
- **Slave** is the device addressed by the master. A slave can be either a receiver or a transmitter.

These concepts are shown in Figure 31.

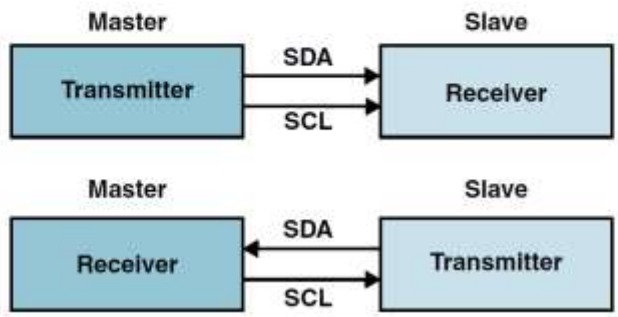

Figure 31. Master/slave and transmitter/receiver relationships

- Multi-master means the ability for more than one master to co-exist on the bus at the same time without collision or data loss.
- Arbitration is the predefined procedure that authorizes only one master at a time to take control of the bus. For more information, see Section 13.2.4.
- Synchronization is the predefined procedure that synchronizes the clock signals provided by two or more masters. For more information, see Section 13.2.5.
- SDA is the data signal line (Serial Data).
- SCL is the clock signal line (Serial Clock).

13.2.1.1 Bus Transfer Terms

The following terms are specific to data transfers that occur to/from the I2C bus.

- START (RESTART). Data transfer begins with a START or RESTART condition. The level of the SDA data line changes from high to low, while the SCL clock line remains high. When this occurs, the bus becomes busy.

- STOP. Data transfer is terminated by a STOP condition. This occurs when the level of the SDA data line changes from low to high, while the SCL clock line remains high. When the data transfer has been terminated, the bus is free or idle again. The bus stays busy if a RESTART is generated instead of a STOP condition.

NOTE
START and RESTART conditions are functionally identical.

13.2.2 I2C Behavior

The I2C can only be controlled through software to be an I2C master, communicating with other I2C slaves. The master is responsible for generating the clock and controlling the transfer of data. The I2C protocol also allows multiple masters to reside on the I2C bus and uses an arbitration procedure to determine the bus ownership. A slave is responsible for either transmitting or receiving data to/from the master. The acknowledgement of data is sent by the device that is receiving data, which can be either a master or a slave.

Each slave has a unique address that is determined by the system designer. When a master wants to communicate with a slave, the master transmits a START/RESTART condition that is then followed by the slave's address and a control bit (R/W) to determine whether the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge pulse (ACK) after the address.

If a master-transmitter writes to a slave-receiver, the receiver gets one byte of data. This transaction continues until the master terminates the transmission with a STOP condition. If a master-receiver reads from a slave-transmitter, the slave transmits a byte of data to the master, and the master then acknowledges the transaction with an ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging (NACK) the transaction after the last byte is received, and then the master issues a STOP condition or addresses another slave after issuing a RESTART condition. This behavior is shown in Figure 32.

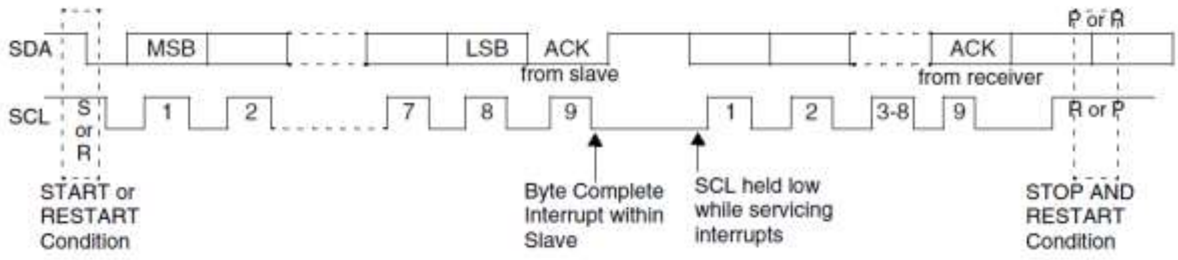

Figure 32. Data transfer on the I2C bus

The I2C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only when the SCL line is low, except for STOP, START, and RESTART conditions. The output drivers are open-drain or open-collector to perform wire-AND functions on the bus. The maximum number of devices on the bus is limited only by the maximum capacitance specification of 400 pF. Data is transmitted in byte packages.

13.2.2.1 START and STOP Generation

When operating as an I2C master, putting data into the transmit FIFO causes the I2C Controller to generate a START condition on the I2C bus. Allowing the transmit FIFO to empty causes the I2C Controller to generate a STOP condition on the I2C bus.

When operating as a slave, the I2C Controller does not generate START or STOP conditions, as per the protocol. However, if a read request is made to the I2C Controller, it holds the SCL line low until read data has been supplied to it. This stalls the I2C bus until read data is provided to the slave I2C Controller, or the I2C Controller slave is disabled by writing a 0 to I2C_ENABLE.

13.2.2.2 Combined Formats

The I2C Controller supports transactions in a read and write combined format in both 7-bit and 10-bit addressing modes.

The I2C Controller does not support mixed address and mixed address format, that is, a 7-bit address transaction followed by a 10-bit address transaction or vice versa.

To initiate combined format transfers, I2C_CON_REG[I2C_RESTART_EN] should be set to 1. With this value set and the I2C Controller operating as a master, when an I2C transfer is completed, the I2C Controller checks the transmit FIFO and executes the next transfer. If the direction of the new transfer differs from the previous one, the combined format is used to issue the transfer. If the transmit FIFO is empty when the current I2C transfer completes, a STOP is issued, and the next transfer is issued after a START condition.

13.2.3 I2C Protocols

The I2C Controller has the following protocols:

- START and STOP Conditions
- Addressing Slave Protocol
- Transmitting and Receiving Protocols
- START BYTE Transfer Protocol

13.2.3.1 START and STOP Conditions

When the bus is idle, both SCL and SDA signals are pulled high through external pull-up resistors on the bus. When a master wants to start a transmission on the bus, it issues a START condition. This is defined to be a high-to-low transition of the SDA signal while SCL is 1. When the master wants to terminate the transmission, it issues a STOP condition. This is defined to be a low-to-high transition of the SDA line while SCL is 1. Figure 33 shows the timing of the START and STOP conditions. When data is being transmitted on the bus, the SDA line must be stable when SCL is 1.

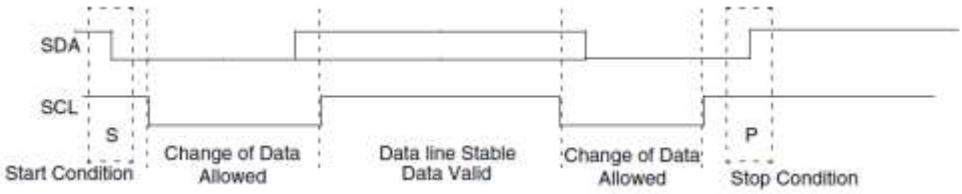

Figure 33. START and STOP conditions

NOTE
The signal transitions for the START/STOP conditions (Figure 33) reflect those observed at the output signals of the master driving the I2C bus. Be careful with observing the SDA/SCL signals at the input signals of the slave(s), because unequal line delays may result in an incorrect SDA/SCL timing relationship.

13.2.3.2 Addressing Slave Protocol

There are two address formats: 7-bit address format and 10-bit address format.

7-bit address format

In the 7-bit address format, the first seven bits (bits 7:1) of the first byte set the slave address and the LSB bit (bit 0) is the R/W bit (Figure 34). When bit 0 (R/W) is set to 0, the master writes to the slave. When bit 0 (R/W) is set to 1, the master reads from the slave.

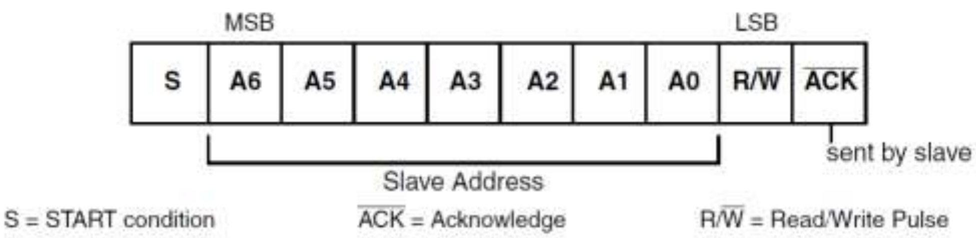

Figure 34. 7-bit address format

10-bit address format

In the 10-bit address format, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition: the first five bits (bits 7:3) notify the slaves that this is a 10-bit transfer, the next two bits (bits 2:1) set the slaves address bits 9:8, and the LSB bit (bit 0) is the R/W bit. The second byte transferred sets bits 7:0 of the slave address. Figure 35 shows the 10-bit address format and Table 43 defines the special purpose and the reserved first byte addresses.

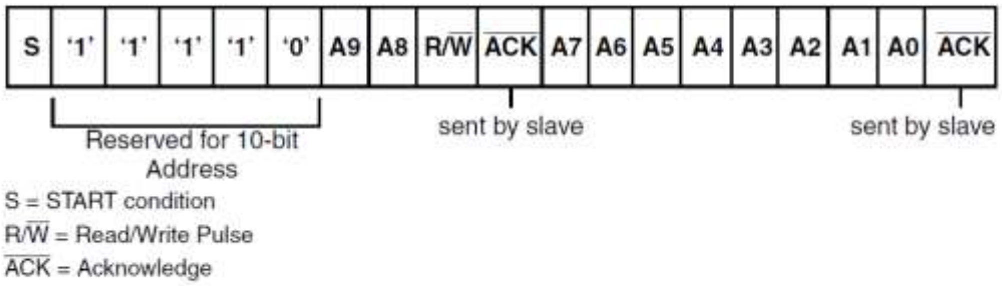

Figure 35. 10-bit address format

Table 43: I2C definition of bits in first byte in 10-bit address format

Slave Address	R/W Bits	Description
0000 000	0	General Call Address. I2C Controller places the data in the receive buffer and issues a General Call interrupt.
0000 000	1	START byte. For more details, see START BYTE Transfer Protocol.
0000 001	X	CBUS address. I2C Controller ignores these accesses.
0000 010	X	Reserved.
0000 011	X	Reserved.
0000 1XX	X	Reserved
1111 1XX	X	Reserved.
1111 0XX	X	10-bit slave addressing.

The I2C Controller does not restrict users from using these reserved addresses. However, if these reserved addresses are used, incompatibilities with other I2C components may occur.

13.2.3.3 Transmitting and Receiving Protocols

A master can initiate data transmission and reception to/from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from a master to either transmit data or receive data to/from the bus, acting as either a slave-transmitter or slave-receiver.

13.2.3.3.1 Master-Transmitter and Slave-Receiver

All data is transmitted in byte format with no limit on the number of bytes transferred per data transfer. After a master sends a slave address and a R/W bit or a master transmits a byte of data to a slave, the slave-receiver must respond with an acknowledge signal (ACK). When a slave-receiver does not respond with an ACK signal, the master aborts the transfer by issuing a STOP condition. The slave must leave the SDA line high so that the master can abort the transfer.

If a master-transmitter is transmitting data as shown in Figure 36, the slave-receiver responds to the master-transmitter with an ACK signal after every byte of data is received.

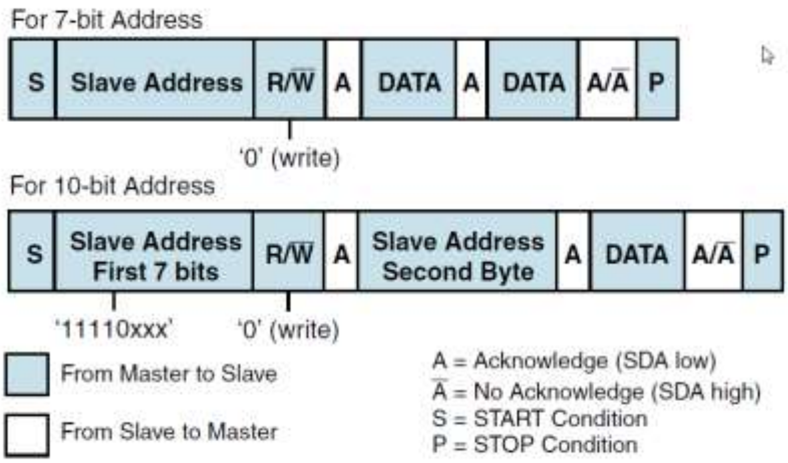

Figure 36. Master-transmitter protocol

13.2.3.3.2 Master-Receiver and Slave-Transmitter

If a master is receiving data as shown in Figure 37, the master responds to the slave-transmitter with an ACK signal after a byte of data has been received except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) so that the master can issue a STOP condition.

When a master does not want to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK signal. The master can then communicate with the same slave or a different slave.

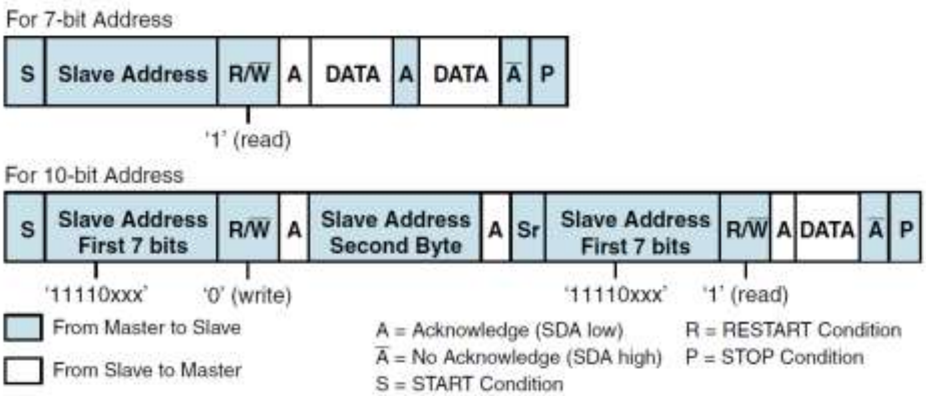

Figure 37. Master-receiver protocol

13.2.3.3.3 START BYTE Transfer Protocol

The START BYTE transfer protocol is set up for systems that do not have an on-board dedicated I2C hardware module. When the I2C Controller is addressed as a slave, it always samples the I2C bus at the highest speed supported so that it never requires a START BYTE transfer. However, when I2C Controller is a master, it supports the generation of START BYTE transfers at the beginning of every transfer in case a slave device requires it. This protocol consists of seven zeros followed by a 1 being transmitted (Figure 38). This allows the processor that is polling the bus to under-sample the address phase until 0 is detected. When the microcontroller detects a 0, it switches from the under-sampling rate to the correct rate of the master.

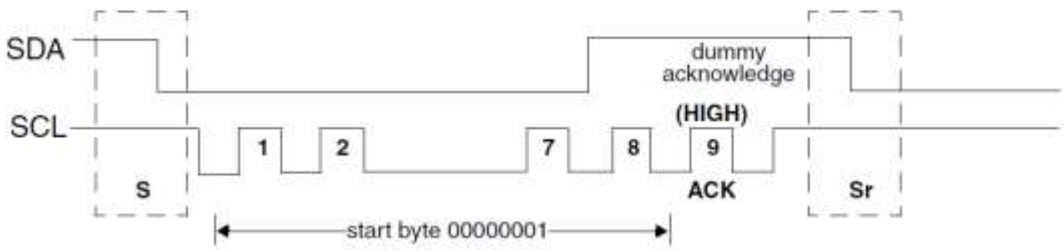

Figure 38. START BYTE transfer

The START BYTE procedure is as follows:

1. Master generates a START condition.
2. Master transmits the START byte (0000 0001).
3. Master transmits the ACK clock pulse. (Present only to conform with the byte handling format used on the bus)
4. No slave sets the ACK signal to 0.
5. Master generates a RESTART (R) condition.

A hardware receiver does not respond to the START BYTE because it is a reserved address and gets reset after the RESTART condition is generated.

13.2.4 Multiple Master Arbitration

The I2C Controller allows multiple masters to reside on the same bus. There is an arbitration procedure if two masters on the same I2C bus try to control the bus at the same time by generating a START condition simultaneously. When a master (for example, a microcontroller) has control of the bus, no other master can take control until the first master sends a STOP condition and places the bus in an idle state.

Arbitration takes place on the SDA line while the SCL line is 1. The master which transmits a 1 while the other master transmits a 0 loses the arbitration and turns off its data output stage. The master that has lost the arbitration can continue to generate clocks until the end of the byte transfer. If both masters are addressing the same slave device, the arbitration could go into the data phase. Figure 39 shows the timing of an arbitration between two masters on the bus.

Control of the bus is determined by the address or master code and data sent by the competing masters, so there is no central master or any order of priority on the bus.

Arbitration is not allowed between the following conditions:

- A RESTART condition and a data bit
- A STOP condition and a data bit
- A RESTART condition and a STOP condition

Slaves are not involved in the arbitration process.

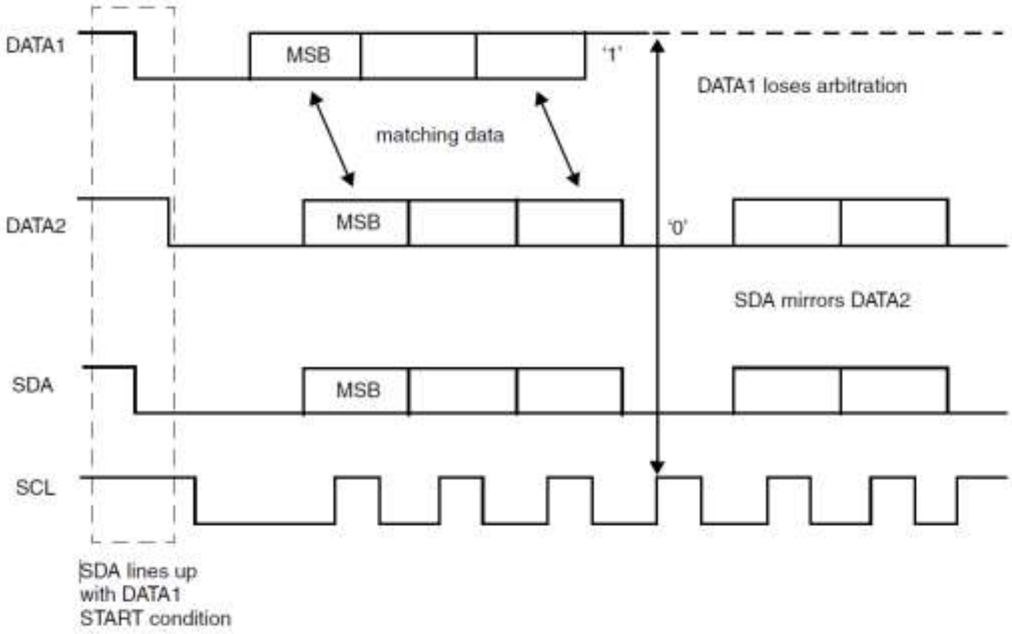

Figure 39. Multiple master arbitration

13.2.5 Clock Synchronization

All masters generate their own clock to transfer messages. Data is only valid during the HIGH period of the SCL clock. When two or more masters try to transfer information on the bus at the same time, they must synchronize the SCL clock. Clock synchronization is performed using the wired-AND connection to the SCL signal. When the master transitions the SCL clock to 0, the master starts counting the LOW period of the SCL clock and transitions the SCL clock signal to 1 at the beginning of the next clock period. However, if another master is holding the SCL line to 0, the first master goes into a HIGH wait state until the SCL clock line transitions to 1.

All masters then count out their HIGH time and the master with the shortest HIGH time transitions the SCL line to 0. The masters then count out their LOW time and the one with the longest LOW time forces the other master into a HIGH wait state. Therefore, a synchronized SCL clock is generated, which is illustrated in Figure 40. Optionally, slaves may hold the SCL line LOW to slow down the timing on the I2C bus.

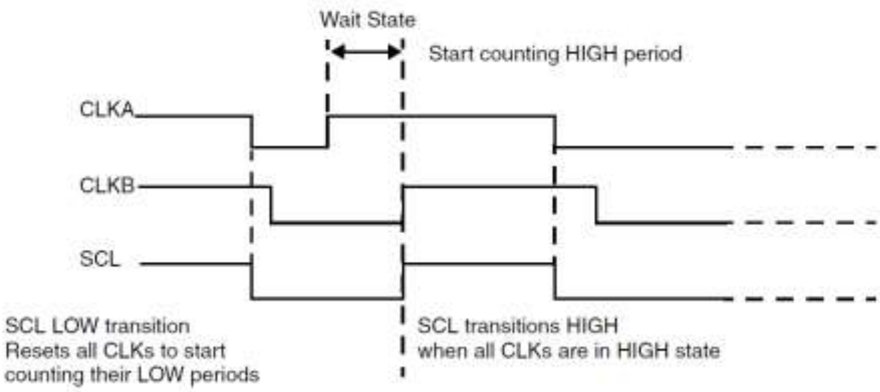

Figure 40. Multiple master clock synchronization

13.3 Programming

To configure and use the I2C Controller:

1. Set up the GPIOs to be used for the I2C interface (P0x_MODE_REG[PID] = 9 or 10).
2. Enable the clock for the I2C Controller (CLK_PER_REG[I2C_ENABLE] = 0x1).
3. Disable the I2C Controller (I2C_ENABLE_REG = 0).
4. Configure the I2C clock frequency:

   a. Standard mode (100 kbit/s): I2C_CON_REG[I2C_SPEED] = 1.
   b. Full speed mode (400 kbit/s): I2C_CON_REG[I2C_SPEED] = 2.
5. Set up the I2C Controller as:
   a. Master: I2C_CON_REG[I2C_MASTER_MODE] = 1 and I2C_CON_REG[I2C_SLAVE_DISABLE] = 1.
   b. Slave: I2C_CON_REG[I2C_MASTER_MODE] = 0 and I2C_CON_REG[I2C_SLAVE_DISABLE] = 0.
6. Choose whether the controller starts its transfers in the 7-bit or 10-bit addressing format when acting as a master (I2C_CON_REG[I2C_10BITADDR_MASTER]) or whether the controller responds to the 7-bit or 10-bit addresses when acting as a slave (I2C_CON_REG[I2C_10BITADDR_SLAVE]).
7. Set target slave address in:
   a. Master mode (I2C_TAR_REG[IC_TAR] = 0x55 (default)).
   b. Slave mode (I2C_SAR_REG[IC_SAR] = 0x55 (default)).
8. Set the threshold levels on RX and TX FIFO (I2C_RX_TL_REG and I2C_TX_TL_REG).
9. Enable the required interrupts (I2C_INTR_MASK_REG).
10. Enable the I2C Controller (I2C_ENABLE_REG = 0x1).
11. Read a byte:
    a. Prepare to transmit the read command byte (I2C_DATA_CMD_REG[I2C_CMD] = 1).
    b. Wait until TX FIFO is empty (I2C_STATUS_REG[TFE] = 1).
    c. Wait until master has finished reading the byte from slave device (I2C_STATUS_REG[MST_ACTIVITY] = 0).
12. Write a byte:
    a. Prepare to transmit the write command byte (I2C_DATA_CMD_REG[I2C_CMD] = 0 and I2C_DATA_CMD_REG[I2C_DAT] = command byte).
    b. Wait until TX FIFO is empty (I2C_STATUS_REG[TFE] = 1).
    c. Wait until master has finished reading the response byte from slave device (I2C_STATUS_REG[MST_ACTIVITY] = 0).

14. UART

14.1 Introduction

The DA14535 contains two instances of the UART block, that is, UART and UART2.

The UART is compliant to the industry-standard 16550 and is used for serial communication with a peripheral. Data is written from a master (CPU) over the APB bus to the UART and it is converted to the serial form and transmitted to the destination device. Serial data is also received by the UART and stored for the master (CPU) to read back.

There is also DMA support on the UART block, thus the internal FIFOs can be used. Only UART supports the hardware flow control signals (RTS and CTS) and the 9-bit mode.

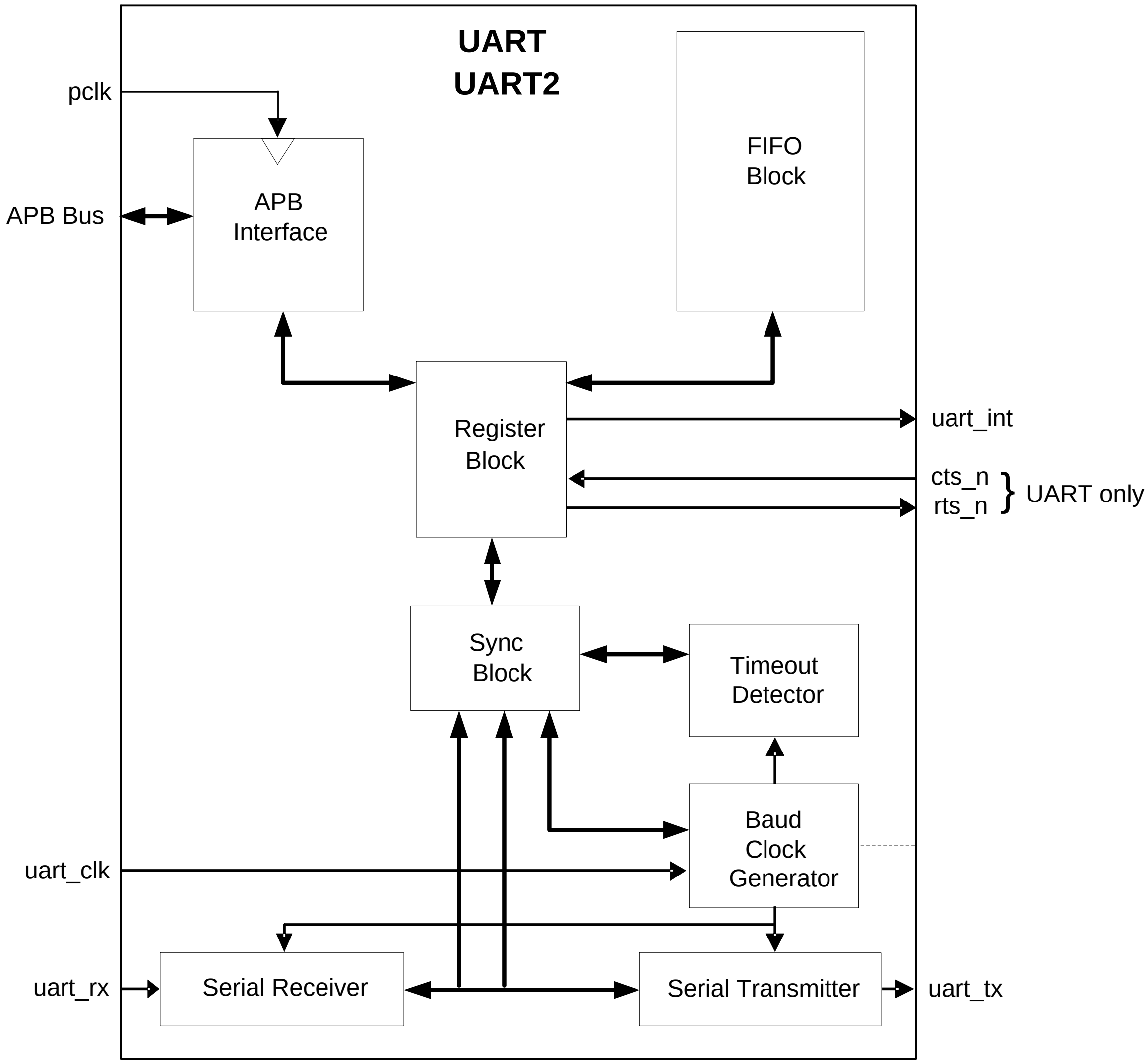

Figure 41. UART block diagram

Features

- 16-byte transmit and receive FIFOs
- Hardware flow control (CTS/RTS) (UART only)
- Shadow registers to reduce software overhead and a software programmable reset is included
- Transmitter Holding Register Empty (THRE) interrupt mode
- Functionality based on the 16550 industry standard:
  - Programmable character properties, such as number of data bits per character (5-8) (optional)
  - Parity bit (with odd or even select) and number of stop bits (1, 1.5, or 2)

- Line break generation and detection
- Prioritized interrupt identification

▪ Programmable serial data baud rate.

14.2 Architecture

14.2.1 UART (RS232) Serial Protocol

Because the serial communication between the UART and the selected device is asynchronous, additional bits (start and stop) are added to the serial data to indicate the beginning and end. Utilizing these bits allows two devices to be synchronized. This structure of serial data accompanied by the start and stop bits is referred to as a character (Figure 42).

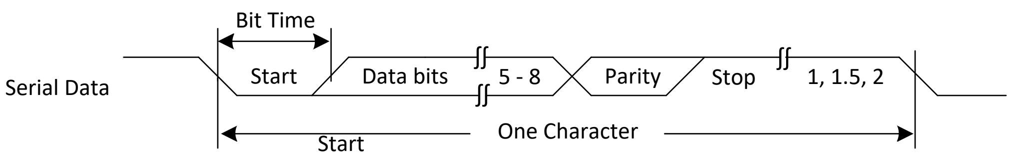

Figure 42. Serial data format

An additional parity bit may be added to the serial character. This bit appears after the last data bit and before the stop bit(s) in the character structure to provide the UART with the ability to perform simple error checking on the received data.

The UART Line Control Register (UART_LCR_REG) is used to control the serial character characteristics. The individual bits of the data word are sent after the start bit, starting with the least-significant bit (LSB). These are followed by the optional parity bit and then by the stop bit(s), which can be of 1, 1.5, or 2 bits.

All the bits in the transmission (except the half stop bit when the 1.5 stop bits are used) are transmitted for the same time duration. This is referred to as a Bit Period or Bit Time. One Bit Time equals to 16 baud clocks. To ensure stability on the line, the receiver samples the serial input data at approximately the mid-point of the Bit Time when the start bit has been detected. As the exact number of baud clocks that each bit has been transmitted for is known, the mid-point for sampling is every 16 baud clocks after the mid-point sample of the start bit. Figure 43 shows the sampling points of the first couple of bits in a serial character.

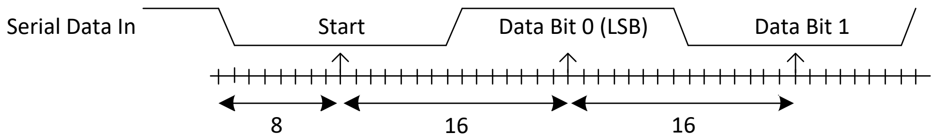

Figure 43. Receiver serial data sampling points

As part of the 16550 standards, an optional baud clock reference output signal (baudout_n) is supplied to provide timing information to the receiving devices that require it. The baud rate of the UART is controlled by the serial clock (*sclk* or *pclk* in a single clock implementation) and the Divisor Latch Register (DLH and DLL) in the following equation:

$$\text{baud rate} = (\text{serial clock frequency})/(16 \times \text{divisor}) \quad (2)$$

where the divisor is a 16-bit integer value plus 4-bit fractional value. The divisor range is 0 to 65535,9375 with steps of 1/16. Divisor High 8-bit integer part is in the DLH register. Divisor Low 8-bit integer part is in the DLL register. Divisor 4-bit fractional port is in the DLF register.

The registers settings for the common baud rate values are presented in Table 44.

Table 44: UART baud rate generation

Baud Rate	Divider	Divisor Latch	DLH Reg	DLL Reg	DLF Reg	Actual BR	Error (%)
1200	833,333	833,3125	3	65	5	1200,03	0,00

Baud Rate	Divider	Divisor Latch	DLH Reg	DLL Reg	DLF Reg	Actual BR	Error (%)
2400	416,667	416,6875	1	160	11	2399,88	0,00
4800	208,333	208,3125	0	208	5	4800,48	0,01
9600	104,167	104,1875	0	104	3	9598,08	0,02
19200	52,083	52,0625	0	52	1	19207,68	0,04
38400	26,042	26,0625	0	26	1	38369,30	0,08
57600	17,361	17,375	0	17	6	57553,96	0,08
115200	8,681	8,6875	0	8	11	115107,91	0,08
230400	4,340	4,3125	0	4	5	231884,06	0,64
460800	2,170	2,1875	0	2	3	457142,86	0,79
921600	1,085	1,0625	0	1	1	941176,47	2,12
1000000	1	1	0	1	0	1000000	0,00

14.2.2 Clock Support

The UART has two system clocks (*pclk* and *sclk*). Having a second asynchronous serial clock (sclk) allows for accurate serial baud rate settings and meeting APB bus interface requirements.

With the two-clock design, a synchronization module is implemented to synchronize all controls and data across the two system clock boundaries.

Although a serial clock faster than four-times the *pclk* does not leave enough time for a complete incoming character to be received and pushed into the receiver FIFO, in most cases the *pclk* signal is faster than the serial clock and this should never be an issue.

The serial clock modules must have time to see new register values and reset their respective state machines. This total time is guaranteed to be no more than eight clock cycles of the slower of the two system clocks. Therefore, no data should be transmitted or received before this maximum time expires after the initial configuration of the UART.

14.2.3 Interrupts

The assertion of the UART interrupt (UART_INT) occurs whenever one of the several prioritized interrupt types are enabled and active. The following interrupt types can be enabled with the IER register:

- Receiver Error
- Receiver Data Available
- Character Timeout (in FIFO mode only)
- Transmitter Holding Register Empty at/below threshold (in Programmable THRE interrupt mode).

When an interrupt occurs, the master accesses the UART_IIR_REG to determine the source of the interrupt before dealing with it accordingly. These interrupt types are described in more detail in Table 45.

Table 45: UART interrupt priorities

Interrupt ID Bits [3-0]	Interrupt set and reset functions			
	Priority	Interrupt type	Interrupt source	Interrupt reset control
0001	-	None		
0110	Highest	Receiver Line status	Overrun/parity/framing errors or break interrupt	Reading the line status register
0100	1	Receiver Data Available	Receiver data available (non-FIFO mode or FIFOs disabled) or RCVR FIFO trigger level reached (FIFO mode and FIFOs enabled)	Reading the receiver buffer register (non-FIFO mode or FIFOs disabled) or the FIFO drops below the trigger level (FIFO mode and FIFOs enabled)

Interrupt ID Bits [3-0]	Interrupt set and reset functions			
	Priority	Interrupt type	Interrupt source	Interrupt reset control
1100	2	Character timeout indication	No characters in or out of the RCVR FIFO during the last four-character times and there is at least one character in it during this time.	Reading the receiver buffer register
0010	3	Transmitter holding register empty	Transmitter holding register empty (Prog. THRE Mode disabled) or XMIT FIFO at or below threshold (Prog. THRE Mode enabled).	Reading the IIR register to check whether there is an interrupt and what its source is; or, writing into THR (FIFOs or THRE Mode not selected or disabled) or XMIT FIFO above threshold (FIFOs and THRE Mode selected and enabled).
0000	4	Reserved		
0111	Lowest	Reserved	-	-

14.2.4 Programmable THRE Interrupt

The UART can be configured to have a Programmable THRE Interrupt mode available to increase system performance.

When Programmable THRE Interrupt mode is selected, it can be enabled through the Interrupt Enable Register (IER[7]). When FIFOs and the THRE Mode are implemented and enabled, THRE Interrupts are active at and below a programmed transmitter FIFO empty threshold level, as shown in the flowchart in Figure 44. Figure 45 shows the programmed transmitter FIFO empty threshold level, where THRE Interrupts are active when the FIFO is empty. In this case the programmable THRE interrupt mode is disabled.

This threshold level is programmed into FCR[5:4]. The available empty thresholds are: empty, 2, ¼, and ½. See UART_FCR_REG for threshold setting details. Selection of the best threshold value depends on the system's ability to begin a new transmission sequence in a timely manner. However, one of these thresholds should prove optimum in increasing system performance by preventing the transmitter FIFO from running empty.

In addition to the interrupt change, Line Status Register (LSR[5]) also switches its function from indicating transmitter FIFO empty to FIFO full. This allows software to fill the FIFO in each transmit sequence by polling LSR[5] before writing another character. Instead of waiting until the FIFO is completely empty, the flow becomes "fill transmitter FIFO whenever an interrupt occurs and there is data to transmit". Waiting until the FIFO is empty causes a performance hit whenever the system is too busy to respond immediately.

Even if everything else is selected and enabled, if the FIFOs are disabled through FCR[0], the Programmable THRE Interrupt mode is also disabled. When not selected or disabled, THRE interrupts and LSR[5] function normally (both reflecting an empty THR or FIFO). The flowchart of THRE interrupt generation when not in programmable THRE interrupt mode is shown in Figure 45.

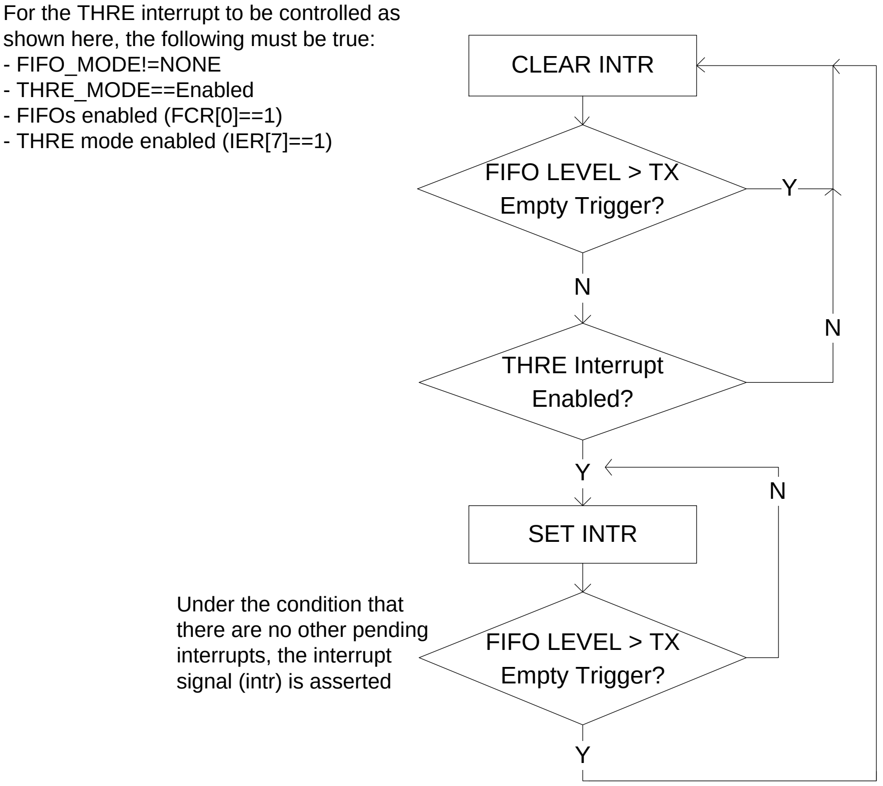

Figure 44. Flowchart of interrupt generation for programmable THRE interrupt mode

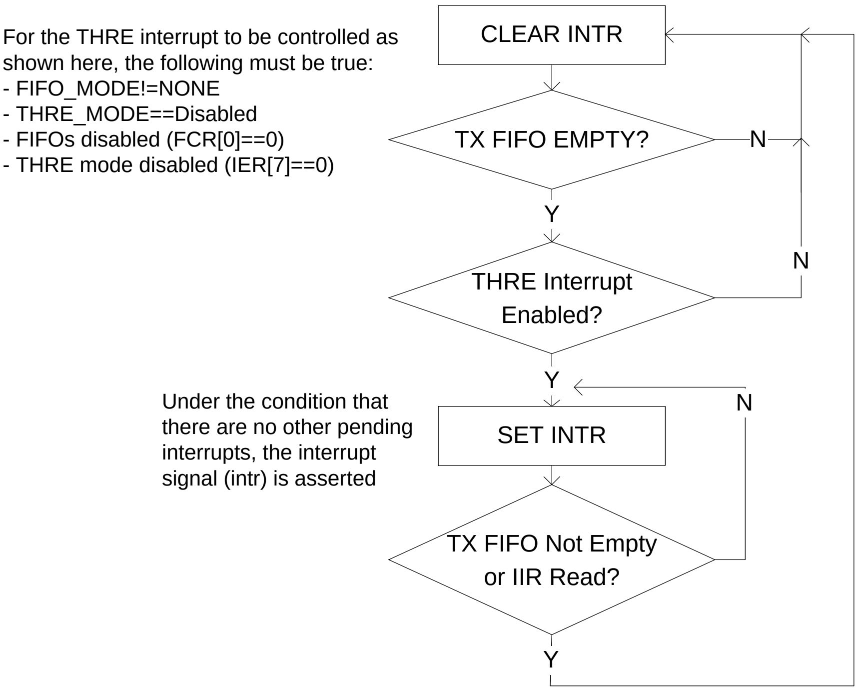

Figure 45. Flowchart of interrupt generation when not in programmable THRE interrupt mode

14.2.5 Shadow Registers

The shadow registers shadow some of the existing register bits that are regularly modified by software. These can be used to reduce the software overhead that is introduced by having to perform read-modify-writes.

- UART_SRBR_REG supports a host burst mode where the host increments its address but still accesses the same receive buffer register.
- UART_STHR supports a host burst mode where the host increments its address but still accesses the same transmit holding register.
- UART_SFE_REG accesses the FCR[0] register without accessing the other UART_FCR_REG bits.
- UART_SRT_REG accesses the FCR[7-6] register without accessing the other UART_FCR_REG bits.
- UART_STER_REG accesses the FCR[5-4] register without accessing the other UART_FCR_REG bits.

14.2.6 Direct Test Mode

The on-chip UARTs can be used for the Direct Test mode required for the final product PHY layer testing. It can be done either over the HCI layer, which engages a full CTS/RTS UART, or by using a 2-wire UART directly as described in the *Bluetooth Low Energy Specification (Volume 6, Part F)*.

14.3 Programming

To configure and use the UART controllers:

1. Set up the GPIOs to be used for the UART interface (P0x_MODE_REG[PID] = 1 to 4 and/or 19-20).
2. Enable the selected UART by setting the CLK_PER_REG[UARTx_ENABLE] bit.
3. Enable access to Divisor Latch Registers (DLL and DLH) by setting the UARTx_LCR_REG[UART_DLAB] bit.
4. Set the desired baud rate. To calculate the registers values for the desired baud rate, use the following equation:

$$\text{Divisor} = \text{UART CLK}/(16 \times \text{Baud rate}) \tag{3}$$

   a. UARTx_IER_DLH_REG: High byte of the Divisor integer part.
   b. UARTx_RBR_THR_DLL_REG: Low byte of the Divisor integer part.
   c. UARTx_DLF_REG: The fractional part of the Divisor.
5. Configure the break control bit, parity, number of stop bits, and data length (UARTx_LCR_REG).
6. Enable and configure the FIFO (UARTx_IIR_FCR_REG).
7. Configure the generated interrupts, if needed (UARTx_IER_DLH_REG).
8. Send a byte:
   a. Check if Transmit Hold Register (THR) is empty (UARTx_LSR_REG[UART_THRE]).
   b. Load the byte to THR (UARTx_RBR_THR_DLL_REG).
   c. Check if the byte has been transmitted (UARTx_LSR_REG[UART_TEMT]).
9. Receive a byte:
   a. Wait until serial data is ready (UARTx_LSR_REG[UART_DR]).
   b. Read the incoming byte from the THR (UARTx_RBR_THR_DLL_REG).

15. SPI Interface

15.1 Introduction

This controller implements the Serial Peripheral Interface (SPI™)[1] for Master and Slave modes. The serial interface can transmit and receive from four bits to up to 32 bits in Master/Slave mode. The controller comprises separate TX and RX FIFOs and DMA handshake support. Slave mode clock speed is independent from the system clock speed. Moreover, master's clock speed can be as fast as the system's clock speed. The controller can generate an interrupt upon data threshold reached in the TX or RX FIFOs.

Features

- Slave and Master mode.
- From 4-bit to up to 32-bit operation.
- SPI Master clock line speed up to 32 MHz, SPI Slave clock line speed up to 16 MHz.
- SPI mode 0, 1, 2, and 3 support (clock edge and phase).
- Built-in separate 8-bit wide and 4-byte deep RX/TX FIFOs for continuous SPI bursts.
- SPI delayed transactions support.
- Maskable interrupt generation based on TX or RX FIFO thresholds.
- DMA support.

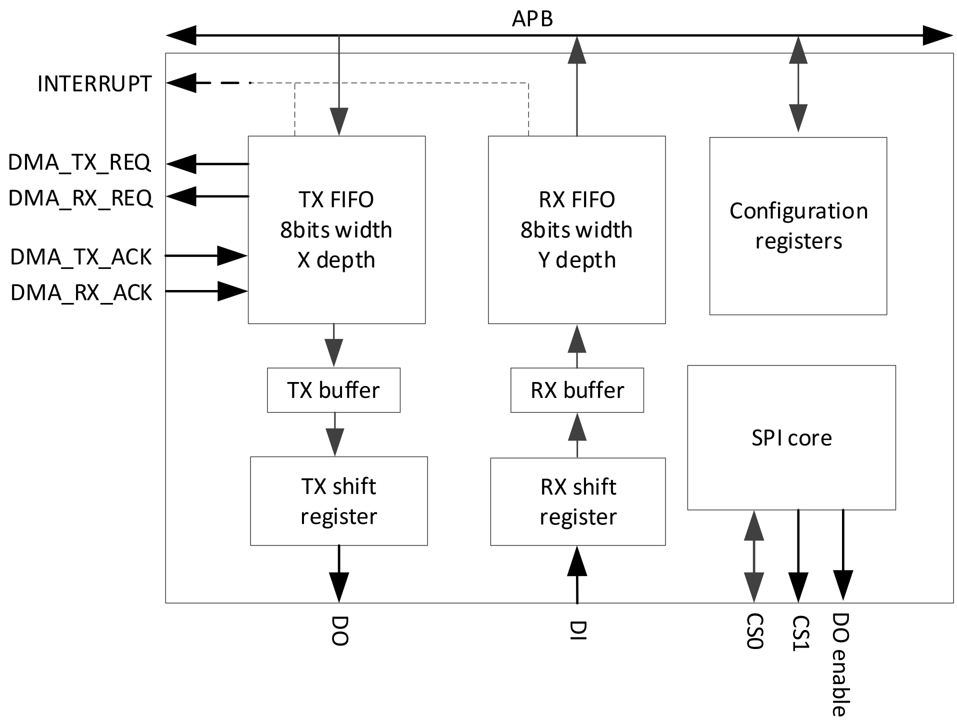

Figure 46. SPI block diagram

1 SPI is a trademark of Motorola, Inc.

15.2 Architecture

The SPI controller is an APB peripheral operating on the apb_clk clock. It contains a front end that is clocked by the spi_clk clock and is responsible for the serialization/deserialization of the data in the RX and TX streams.

Two separate FIFOs, each of eight bits wide and four bytes deep, are used to store data for RX and TX streams. Because an SPI word can be configured to be from four bits to up to 32 bits, one to four FIFO positions can be written/read at the same time. FIFOs contain logic implementing programmable thresholds comparison.

The SPI controller supports DMA requests and interrupt generation based on the FIFO thresholds. If enabled, a DMA request and/or interrupt is asserted with whether TX_FIFO level is low or RX_FIFO level is high.

The SPI interface supports all four modes of operation and the corresponding polarity (CPOL) and phase (CPHA) of the SPI clock (SPI_CLK) are defined in Table 46.

Table 46: SPI modes configuration and SCK states

SPI Mode	CPOL	CHPA	TX SPI_CLK	RX SPI_CLK	Idle SPI_CLK
0	0	0	Falling edge	Rising edge	Low
1	0	1	Rising edge	Falling edge	Low
2	1	0	Rising edge	Falling edge	High
3	1	1	Falling edge	Rising edge	High

To read from or to write to an external single byte Flash device in the SPI master mode, a byte swap mechanism is implemented to allow for a proper placement of the bytes in a 16-bit word for the DMA to write to/read from the internal RAM. More specifically, when the SPI controller is configured as a master with DMA support and a 16-bit word width so that the bus utilization is increased compared to reading from an 8-bit device, the byte swap mechanism brings the least significant byte read and place it in the most significant byte in the 16-bit word. The controller automatically swaps the bytes to allow for placing the first byte read in the least significant byte of the 16-bit word. This feature is programmable through SPI_CTRL_REG[SPI_SWAP_BYTES].

The SPI controller can operate at the highest speed (32 MHz on the SPI_CLK line) in a special master mode. The clock of the controller is then either the XTAL32M or the RC32M and can be used for fast booting from external Flash devices that support this frequency.

15.2.1 SPI Timing

The timing of the SPI interface when the SPI controller is in slave mode is shown in Figure 47.

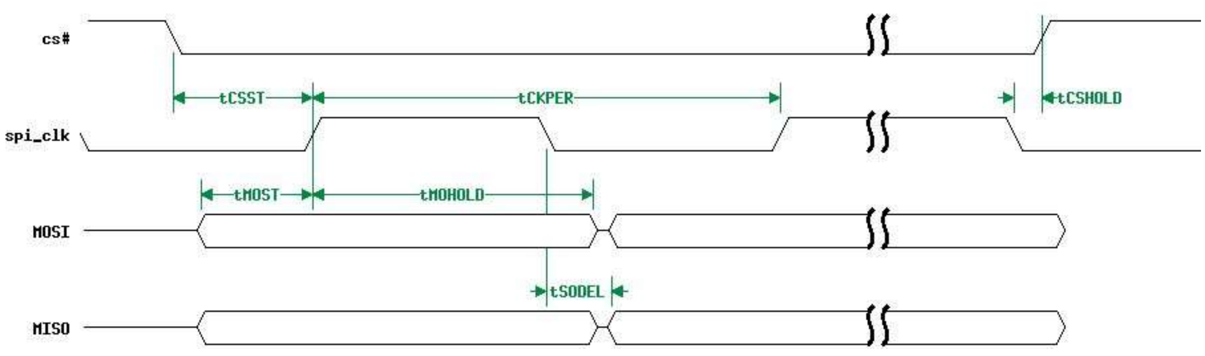

Figure 47. SPI Slave mode timing (CPOL = 0, CPHA = 0)

Table 47: SPI timing parameters

Parameter	Description	Typ	Unit
t_{CKPER}	spi_clk clock period	60	ns
t_{CSST}	CS active time before the first edge of spi_clk	1 spi_clk cycle	
t_{CSHOLD}	CS non-active time after the last edge of spi_clk	1 spi_clk cycle	
t_{MOST}	Master input data latching setup time	(spi_clk/2) – 5	ns

Parameter	Description	Typ	Unit
t_{MOHOLD}	Master input data hold time	0	ns
t_{SODEL}	Slave output data delay	25	ns

15.2.2 SPI Controller Enhancements

The SPI controller, when in master mode, can automatically read (without the use of the CPU) a known number (predefined) of bytes from an external slave device. This enhancement targets to simplify the software and allow the use of the Arm core for other critical operations happening in parallel instead of keeping it busy in the SPI transfer loop.

The feature is utilized using one of the following methods:

- In case the external slave device provides a data strobe signal (dedicated DATAREADY pin to indicate when in active-LOW or active-HIGH level (programmable) that the next byte is available for read), the SPI controller ends the active transfer and initiate the next one as soon as the strobe signal signals that slave is ready.
- In case the external slave device does not provide a data strobe signal but the time that a certain programmable number of bytes is available is known, the SPI controller can stall for a programmable delay (in SPI clock cycles) until the known time is elapsed and then initiate the data transfer.

Additionally, the combination of the above two methods is possible, meaning that the SPI controller can check the data strobe signal (DATAREADY) after the known time has elapsed.

15.3 Programming

15.3.1 Master Mode

To configure the SPI controller in master mode:

1. Set the appropriate GPIO ports in SPI clock mode (output), SPI Chip Select mode (output), SPI Data Out mode (output), and SPI Data In mode (input).
2. Enable SPI clock by setting CLK_PER_REG[SPI_ENABLE] = 1.
3. Reset SPI FIFO by setting SPI_CTRL_REG[SPI_FIFO_RESET] = 1.
4. Set the SPI clock mode (synchronous or asynchronous with APB clock) by programming SPI_CLOCK_REG[SPI_MASTER_CLK_MODE].
5. Set the SPI clock frequency by programming SPI_CLOCK_REG[SPI_CLK_DIV]. If SPI_CLK_DIV is not 0x7F, SPI_CLK = module_clk/2 × (SPI_CLK_DIV + 1). If SPI_CLK_DIV = 0x7F, SPI_CLK = module_clk.
6. Set the SPI mode (CPOL or CPHA) by programming SPI_CONFIG_REG[SPI_MODE].
7. Set the SPI controller in master mode by setting SPI_CONFIG_REG[SPI_SLAVE_EN] = 0.
8. Define the SPI word length (from 4-bit to 32-bit) by programming SPI_CONFIG_REG[SPI_WORD_LENGTH]. SPI_WORD_LENGTH = word length - 1.

To read/write the following sequence should be performed:

1. If a slave device is slow and does not give the data at the correct clock edge, configure the SPI module to capture data at the next clock edge by setting SPI_CTRL_REG[SPI_CAPTURE_AT_NEXT_EDGE] = 1. Otherwise, set SPI_CTRL_REG[SPI_CAPTURE_AT_NEXT_EDGE] = 0.
2. Release FIFO reset by setting SPI_CTRL_REG[SPI_FIFO_RESET] = 0.
3. Enable SPI TX path by setting SPI_CTRL_REG[SPI_TX_EN] = 1.
4. Enable SPI RX path by setting SPI_CTRL_REG[SPI_RX_EN] = 1.
5. Enable the SPI chip select by programming the SPI_CS_CONFIG_REG[SPI_CS_SELECT] = 1 or 2. This option allows the master to select the slave that is connected to the GPIO that has the function of SPI_CS0 or SPI_CS1.
6. Enable the SPI controller by setting SPI_CTRL_REG[SPI_EN] = 1.
7. Write to TX FIFO by programming SPI_FIFO_WRITE_REG[SPI_FIFO_WRITE]. Write access is permitted only when SPI_FIFO_STATUS_REG[SPI_TX_FIFO_FULL] = 0.

8. Read from RX FIFO by programming SPI_FIFO_READ_REG[SPI_FIFO_READ]. Read is permitted only when SPI_FIFO_STATUS_REG[SPI_RX_FIFO_EMPTY] = 0.
9. To disable the SPI chip select, set SPI_CS_CONFIG_REG[SPI_CS_SELECT] = 0 to deselect the slave and set SPI_CTRL_REG[SPI_FIFO_RESET] = 1 to reset the SPI FIFO.

15.3.2 Slave Mode

1. Set the appropriate GPIO ports in SPI clock mode (input), SPI Chip Select mode (input), SPI Data Out mode (output), and SPI Data In mode (input).
2. Enable SPI clock by setting CLK_PER_REG[SPI_ENABLE] = 1.
3. Reset SPI FIFO by setting SPI_CTRL_REG[SPI_FIFO_RESET] = 1.
4. Set the SPI mode (CPOL or CPHA) by programming SPI_CONFIG_REG[SPI_MODE].
5. Set the SPI module in slave controller by setting SPI_CONFIG_REG[SPI_SLAVE_EN] = 1.
6. Define the SPI word length (from 4-bit to 32-bit) by programming SPI_CONFIG_REG[SPI_WORD_LENGTH]. SPI_WORD_LENGTH = word length - 1.

To read/write the following sequence must be performed:

1. Set SPI FIFO in normal operation by setting SPI_CTRL_REG[SPI_FIFO_RESET] = 0.
2. Enable SPI TX path by setting SPI_CTRL_REG[SPI_TX_EN] = 1.
3. Enable SPI RX path by setting SPI_CTRL_REG[SPI_RX_EN] = 1.
4. Enable the SPI controller by setting SPI_CTRL_REG[SPI_EN] = 1.
5. Write the first data byte directly to TX buffer by programming the SPI_TXBUFFER_FORCE_L_REG[SPI_TXBUFFER_FORCE_L].
6. Write the rest of the data to TX FIFO by programming SPI_FIFO_WRITE_REG[SPI_FIFO_WRITE]. Write access is permitted only if SPI_FIFO_STATUS_REG[SPI_TX_FIFO_FULL] = 0.
7. Read from RX FIFO by programming SPI_FIFO_READ_REG[SPI_FIFO_READ]. Read is permitted only if SPI_FIFO_STATUS_REG[SPI_RX_FIFO_EMPTY] = 0.

16. Quadrature Decoder

16.1 Introduction

The DA14535 has an integrated Quadrature decoder that can automatically decode the signals for the X, Y, and Z axes of a HID input device, reporting step count and direction. It can also be programmed to simply count rising/falling edges on any of the channel pairs. This block can be used to wake up the chip as soon as there is a movement from the connected external device. Figure 48 shows the block diagram of the quadrature decoder.

Features

- Three 16-bit signed counters that provide the step count and direction on each of the axes (X, Y, and Z) and one 8-bit counter counting the overall edges from all the three counters.
- Programmable system clock sampling at a maximum of 16 MHz.
- APB interface for control and programming.
- Programmable source from the GPIOs.
- Digital filter on the channel inputs to avoid spikes.

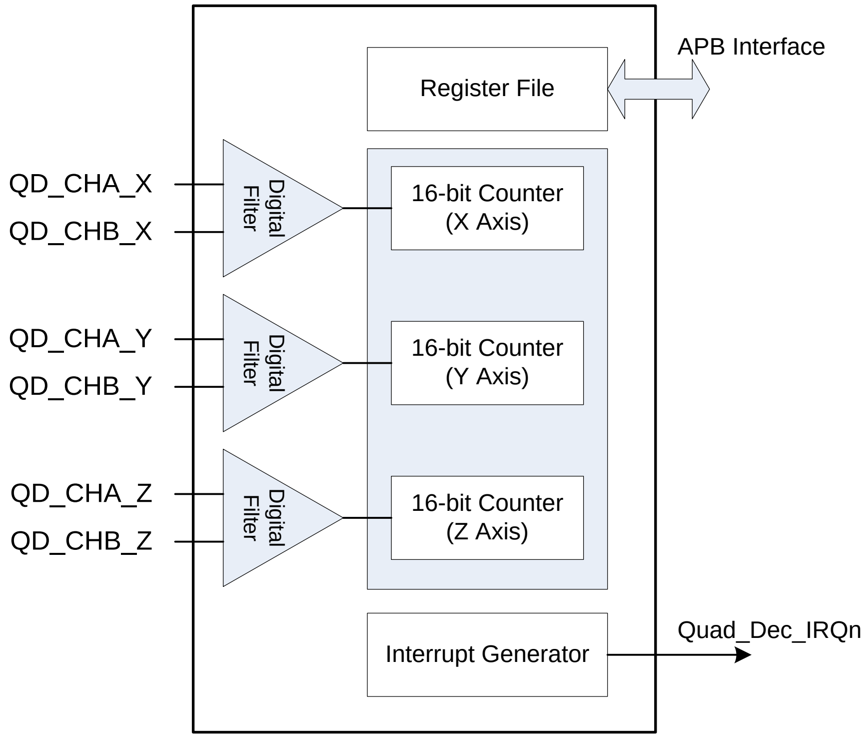

Figure 48. Quadrature decoder block diagram

16.2 Architecture

Channels are expected to provide a pulse train with 90 degrees rotation as displayed in Figure 49 and Figure 50.

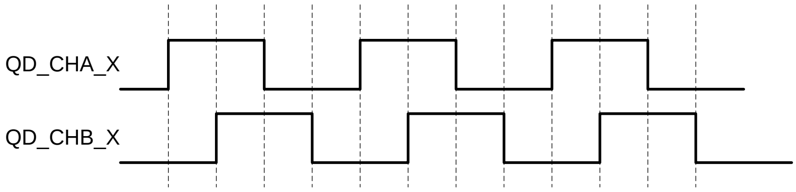

Figure 49. Moving forward on axis X

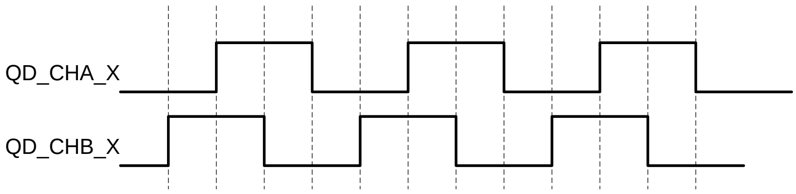

Figure 50. Moving backwards on axis X

Depending on whether channel A or channel B is leading in phase, the quadrature decoding block calculates the direction on the related axis. Furthermore, the signed counter value represents the number of steps moved.

Users can choose which GPIOs to use for the channels by programming the QDEC_CTRL2_REG register. The block supports two modes of operation: quadrature counting and edges counting. The quadrature counting mode reads the patterns of successive pulses as in Figure 49 and Figure 50, while the edges counting mode simply counts all positive and negative edges on any of the two channels of a pair.

NOTE
If two edges happen at the same time, the counter only counts one.

The digital filter eliminates spikes shorter than three clock periods. It is followed by an edge detection circuitry, and they are shown in Figure 51.

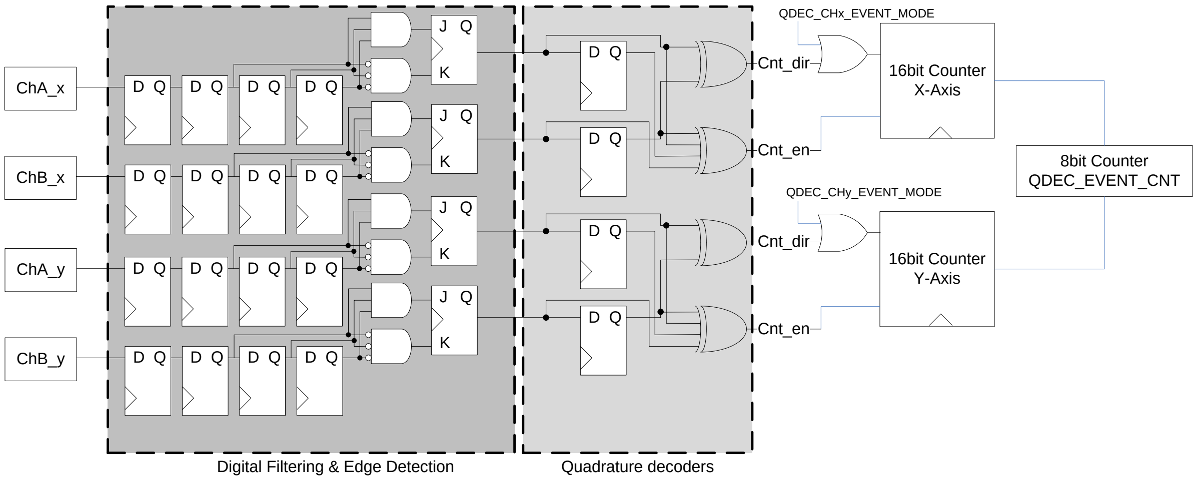

Figure 51. Digital filtering and edge detection circuit

A counter for its dedicated axis holds the movement events of the channels. When a channel is disabled, the counter is reset. The counters are accessible through the APB bus.

The QDEC_EVENT_CNT gathers all edges on all channels regardless of the mode of operation. If two edges happen at the same time, this counter is only increased by one.

The quadrature decoder operates on the system clock. The QDEC_CLOCKDIV register defines the number of clock cycles during which the decoding logic samples the data on the channel inputs. The division is automatically disabled when the lp_clk is used as the system clock.

16.3 Programming

To program the quadrature decoder for actual quadrature counting or edge counting:

1. Configure the clock frequency by configuring the QDEC_CLOCKDIV register. The value in this register is dividing the sys_clk. However, if sys_clk = lp_clk, this divider is completely bypassed.
2. Define which pin pairs represent the different channels for the X, Y, and Z axes or the GPIOs from which the edges are counted. Configure such information at QDEC_CHX_PORT_SEL, QDEC_CHY_PORT_SEL, and QDEC_CHZ_PORT_SEL registers.
3. Configure the interrupt threshold upon which an interrupt is generated at QDEC_CTRL_REG[QDEC_IRQ_THRES]. Note that the interrupt threshold is based on the value of QDEC_EVENT_CNT_REG which keeps on counting after the interrupt is generated.

4. Define the mode of operation by configuring the respective QDEC_CHx_EVENT_MODE field in the QDEC_CTRL2_REG.
5. Enable the clock of the block by writing at CLK_PER_REG[QUAD_ENABLE].
6. Wait for the interrupt and then read X, Y, and Z values at QDEC_XCNT_REG, QDEC_YCNT_REG, and QDEC_ZCNT_REG (in the quadrature counting case) or the QDEC_EVENT_CNT_REG (in the edges counting case).
7. Clear the interrupt (by writing at QDEC_CTRL_REG[QDEC_IRQ_STATUS]) and the edge counter (by writing at QDEC_CTRL_REG[QDEC_EVENT_CNT_CLR) if needed.

17. Clockless Wake-up Controller

17.1 Introduction

The clockless wake-up controller implements a circuit that enables the RC32K clock which in turn triggers the hardware startup FSM to allow the system to be woken up by an external event (a GPIO toggle). This controller is only used when the system is in hibernation mode, that is, when all clocks are stopped.

Features

- Wake up the system from specific GPIOs (P0_1, P0_2, P0_3, P0_4, and P0_5).
- Configurable polarity of the GPIO signals (single configuration for all GPIOs).
- Special RC filtered inputs feeding the wake-up control circuit (Type B pads).
- Always powered.

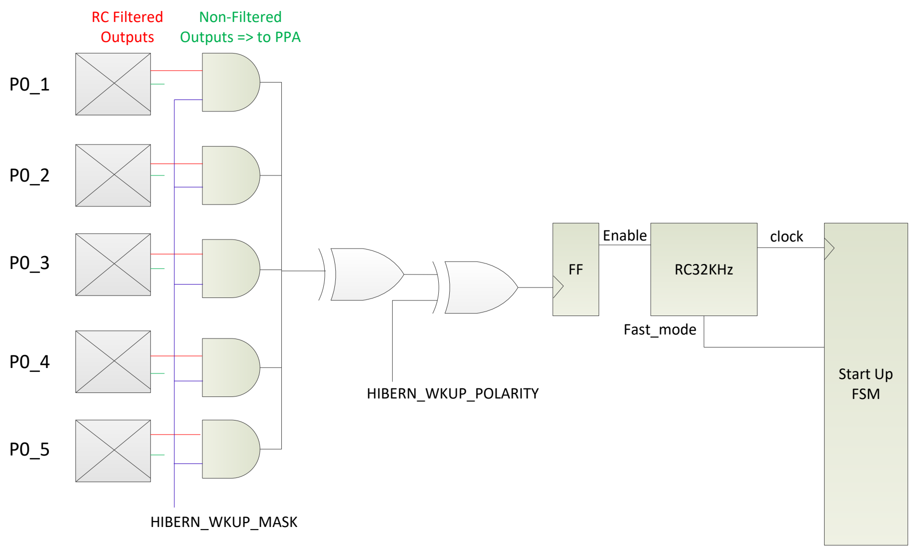

Figure 52. Clockless wake-up controller circuit

17.2 Architecture

The clockless wake-up controller automatically enables the RC32K oscillator when a toggle is identified in one of the five specific GPIOs (P0_1, P0_2, P0_3, P0_4, and P0_5). These GPIO signals are connected to the Type B pads, which provide two outputs to the digital domain. One output goes through a Schmitt trigger and the other one goes through an RC filter with a cutoff frequency of 100 kHz and a Schmitt trigger. The output going through the RC filter and a Schmitt trigger feeds the clockless wake-up controller circuitry. Hence, any spikes larger than 100 kHz are filtered out without waking up the system.

The triggering GPIO can be defined by means of a masking register. If no GPIO is masked out, any toggle in any of these GPIOs creates an edge which serves as a clock for a Flip-Flop to lock and enable the oscillator. The polarity of the edge is also programmable, but it is common for the GPIOs and not a dedicated bit per GPIO.

Note that, if two opposite edges occur exactly at the same time on two GPIOs that are allowed to wake up the system, the XOR output does not change its value and no wake-up occurs. However, even if the GPIO signal events have a couple of ns difference, the circuit still understands it and the clock is started.

17.3 Programming

To program the clockless wake-up controller before setting the system into hibernation mode:

1. Define which pins are allowed to wake up the system from hibernation by configuring the HIBERN_CTRL_REG[HIBERN_WKUP_MASK].
2. Define the polarity of the waking up events at HIBERN_CTRL_REG[HIBERN_WKUP_POLARITY]. This should be done by reading the value of the unmasked GPIOs and programming the polarity register with their XOR'ed state.

3. Enable the hibernation mode by programming the HIBERN_CTRL_REG[HIBERNATION_ENABLE] bit field. Note that this action stops all clocks when the system drops to sleep.
4. Allow RAM to be retained by programming the RAM_PWR_CTRL_REG accordingly.
5. Define where address 0 is to be mapped at SYS_CTRL_REG[REMAP_ADR0] so that the CPU can execute code right after waking up. If RAM is retained, REMAP_ADR0 should point to 0x2 or 0x3.
6. Clear RESET_STAT_REG. Clear this register means that the system wakes up from hibernation mode.
7. Put the system to sleep by executing the WFI command with the SCR bit set.

18. Clocked Wake-up Controller

18.1 Introduction

The clocked wake-up controller can be programmed to wake up DA14535 from Deep Sleep mode and Extended Sleep mode upon a pre-programmed number of GPIO events on a maximum of two pins in parallel. This wake-up controller resides in the PD_SLP power domain and operates on the LP_CLK.

Figure 53 shows the block diagram illustrating the wake-up function.

Features

- Monitors GPIO state changes.
- Implements debouncing time from 0 ms up to 63 ms on two GPIOs in parallel.
- Accumulates external events and compares the number to a programmed value.
- Generates an interrupt to the CPU's WIC.

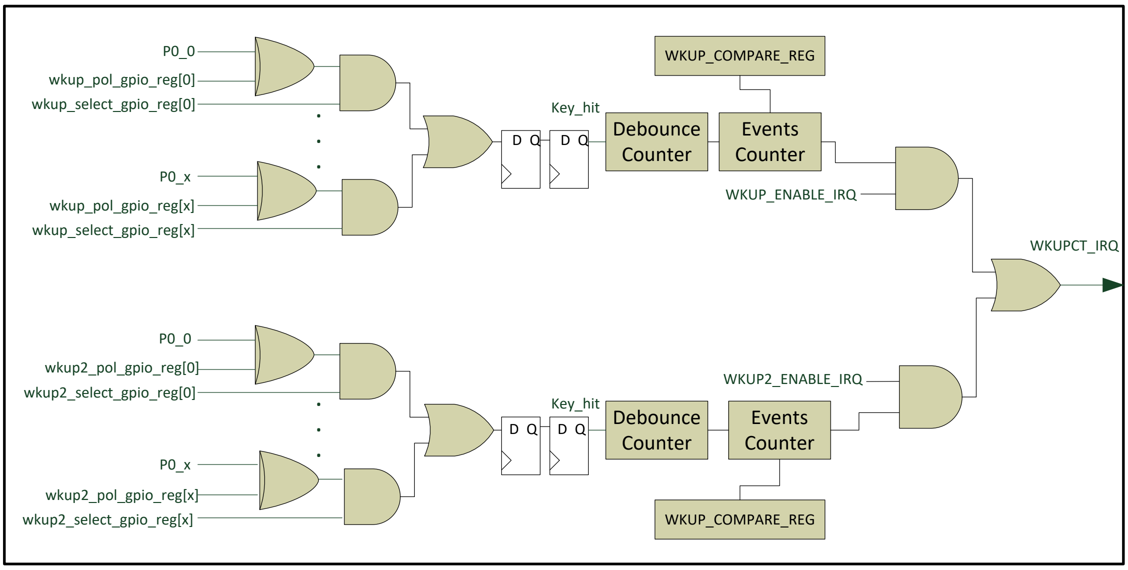

Figure 53. Clocked wake-up controller block diagram

18.2 Architecture

The controller comprises two identical circuits that implement the edge detection, debouncing, and event counting before generating a wake-up interrupt towards the CPU.

A LOW to HIGH level transition on the selected input port sets internal signal "key_hit" to 1, while WKUP_POL_GPIO_REG[y] = 0. This signal triggers the event counter state machine as shown in Figure 54. The debounce counter is loaded with the value of WKUP_CTRL_REG[WKUP_DEB_VALUE]. The timer counts down every 1 ms or 125 µs. The signal state is constantly monitored. If the debounce counter reaches 0, it means that the key_hit signal state has been stable over the amount of clock cycles counted by the debounce counter.

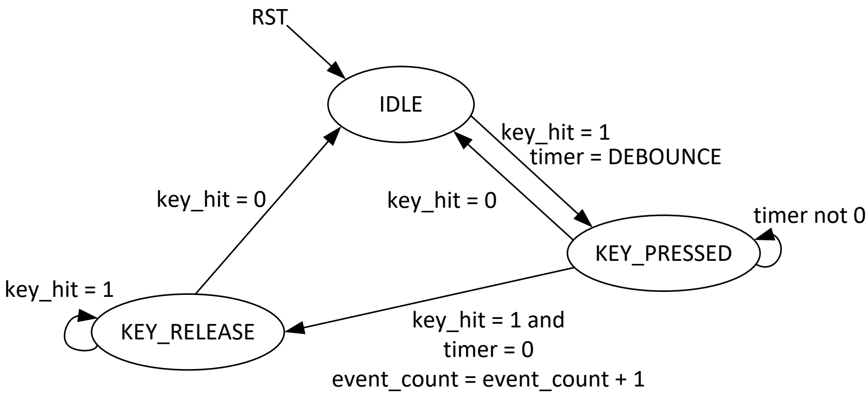

Figure 54. Event counter state machine for the wake-up interrupt generator

The event counter is edge sensitive. After an active edge is detected, a reverse edge must be detected first before the event counter goes back to the IDLE state and from there starts waiting for a new active edge.

If the event counter is equal to the value set in the WKUP_COMPARE_REG register, the counter is reset and an interrupt is generated, if the interrupt generation has been enabled by WKUP_ENABLE_IRQ and WKUP2_ENABLE_IRQ in the WKUP_CTRL_REG.

NOTE
There is only one register for both circuits that contains the number of events before an interrupt is issued.

The interrupt can be cleared by writing a value to the register WKUP_RESET_IRQ_REG. The event counter can be reset by writing a value to the register WKUP_RESET_CNTR_REG. The value of the event counter can be read at any time by reading register WKUP_COUNTER_REG.

Any of the GPIO inputs can be selected to generate an event by programming the corresponding WKUP/WKUP2_SELECT_GPIO_REG register. When both WKUP/WKUP2_SELECT_GPIO_REG registers are configured to generate a wake-up interrupt, a toggle on any GPIO wakes up the system.

The input signal edge can be selected by programming the WKUP/WKUP2_POL_GPIO_REG registers.

NOTE
A minimum of 2 low power clocks pulse is required on a GPIO to be correctly identified as a wake-up edge trigger.

18.3 Programming

To configure the clocked wake-up controller:

1. Define the polarity of the triggering GPIOs at WKUP/WKUP2_POL_GPIO_REG.
2. Configure the debouncing counters by programming WKUP_CTRL_REG[WKUP_DEB_VALUE] with the amount of time (ms) during which the signal should be re-sampled before its state is decided. Note that there is a single bit field for both debouncing counters.
3. Define the number of events that are needed to trigger the wake-up interrupt by programming the WKUP_COMPARE_REG. Note there is only one register for both circuits.
4. Allow the interrupt generation by configuring the WKUP_ENABLE_IRQ and WKUP2_ENABLE_IRQ bit fields, respectively, in the WKUP_CTRL_REG.
5. Define which GPIOs are allowed to trigger a wake-up event at WKUP/WKUP2_SELECT_GPIO_REG.
6. Set the system to Deep Sleep mode or Extended Sleep mode by executing the WFI command with the SCR bit set.

19. Timer 0

19.1 Introduction

Timer 0 is a 16-bit general purpose software programmable timer, which has the ability of generating Pulse Width Modulated (PWM) signals PWM0 and PWM1. It also generates the SWTIM_IRQ interrupt to the Arm Cortex-M0+. It can be configured in various modes regarding output frequency, duty cycle, and the modulation of the PWM signals. Figure 55 shows the block diagram of Timer 0.

Features

- 16-bit general purpose timer
- Ability to generate two PWM signals (PWM0 and PWM1)
- Programmable output frequency (f) with N = 0 to (2^{16}-1) and M = 0 to (2^{16}-1)

$$f = \frac{(16,\ 8,\ 4,\ 2\ MHz\ or\ 32\ kHz)}{(M+1)+(N+1)}$$

- Programmable duty cycle (δ):

$$\delta = \frac{M+1}{(M+1)+(N+1)} \times 100\ \%$$

- Separately programmable interrupt timer:

$$T = \frac{(16,\ 8,\ 4,\ 2\ MHz\ or\ 32\ kHz)}{(ON+1)}$$

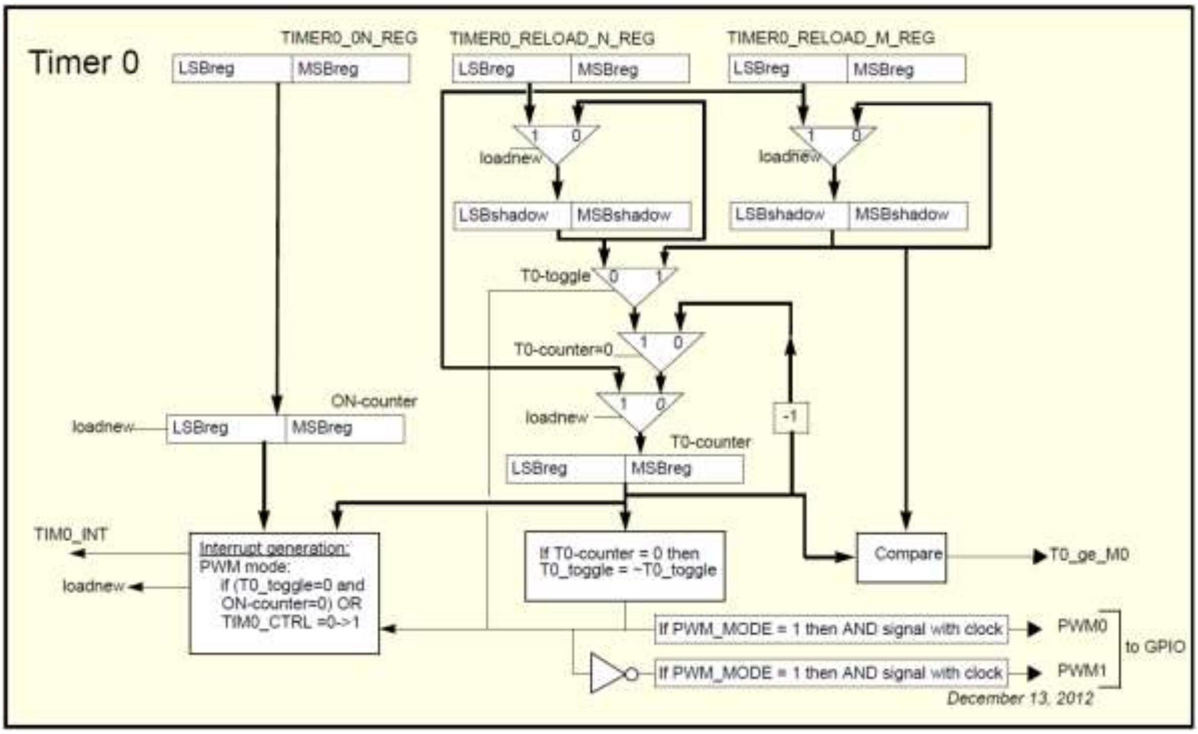

Figure 55. Timer 0 block diagram

19.2 Architecture

The 16-bit Timer 0 consists of two counters, that is, T0-counter and ON-counter, and three registers, that is, TIMER0_RELOAD_M_REG, TIMER0_RELOAD_N_REG, and TIMER0_ON_REG. Upon reset, the counter and register values are 0x0000. Timer 0 generates a PWM signal PWM0, of which the frequency and duty cycle are determined by the contents of the TIMER0_RELOAD_N_REG and the TIMER0_RELOAD_M_REG registers. The PWM1 signal is the inverted version of PWM0.

Timer 0 can run at five different clocks: 16 MHz, 8 MHz, 4 MHz, 2 MHz, or 32 kHz. The 32 kHz clock is selected by default with bit TIM0_CLK_SEL in the TIMER0_CTRL_REG register. This slow clock has no enabling bit. The other four options can be selected by setting the TIM0_CLK_SEL bit and the TMR_ENABLE bit in the CLK_PER_REG (by default the TMR_ENABLE bit is disabled). This register also controls the four higher clock frequency on which Timer 0 runs through the TMR_DIV bits. An extra clock divider is available and can be activated through bit TIM0_CLK_DIV of the timer control register TIMER0_CTRL_REG. This clock divider is only used for the ON-counter and always divides the clock for the ON-counter by 10.

NOTE
If the LP clock is selected as system clock, the CLK_AMBA_REG[HCLK] bit field should always be 0 to ensure the proper operation of the timer.

Timer 0 operates in PWM mode. The signals PWM0 and PWM1 can be mapped to any GPIOs.

Timer 0 PWM mode

If bit TIM0_CTRL in the TIMER0_CTRL_REG is set, Timer 0 starts running. SWTIM_IRQ is generated, and the T0-counter loads its start value from the TIMER0_RELOAD_M_REG register and decrements on each clock cycle. The ON-counter also loads its start value from the TIMER0_ON_REG register and decrements with the selected clock.

When the T0-counter reaches zero, the internal signal T0-toggle is toggled to select the TIMER0_RELOAD_N_REG whose value is loaded in the T0-counter. Each time the T0-counter reaches zero, it is alternately reloaded with the values of the M0- and N0-shadow registers. PWM0 is high when the M0-value decrements and low when the N0-value decrements. For PWM1 the opposite is applicable because it is inverted. If bit PWM_MODE in the TIMER0_CTRL_REG register is set, the PWM signals are not HIGH during the "high time" but output a clock in that stage. The frequency is based on the clock settings defined in the CLK_PER_REG register (also when the 32-kHz clock is used), but the selected clock frequency is divided by two to get a 50% duty cycle.

If the ON-counter reaches zero, it remains zero until the T0-counter also reaches zero, while decrementing the value loaded from the TIMER0_RELOAD_N_REG register (PWM0 is low). The counter then generates an interrupt (SWTIM_IRQ). The ON-counter is reloaded with the value of the TIMER0_ON_REG register. The T0-counter as well as the M0-shadow register is loaded with the value of the TIMER0_RELOAD_M_REG register. At the same time, the N0-shadow register is loaded with the value of TIMER0_RELOAD_N_REG register. Both counters are decremented on the next clock again and the sequence is repeated.

NOTE
It is possible to generate interrupts at a high rate by selecting a high clock frequency and low counter values. This could result in missed interrupt events.

During the time when the ON-counter is non-zero, new values for the ON-register, M0-register, and N0-register can be written, but they are not used by the T0-counter until a full cycle is finished. More specifically, the newly written values in the TIMER0_RELOAD_M_REG and TIMER0_RELOAD_N_REG registers are only stored into the shadow registers when the ON-counter and the T0-counter have both reached zero and the T0-counter is decrementing the value loaded from the TIMER0_RELOAD_N_REG register (Figure 56).

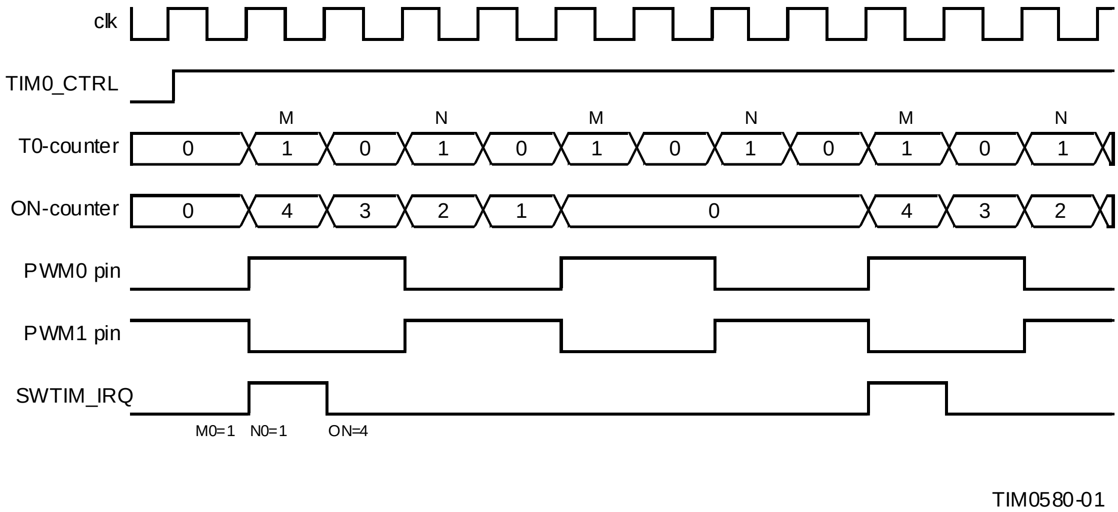

Figure 56. Timer 0 PWM mode

At start-up, both counters and the PWM0 signal are LOW, so at start-up an interrupt is also generated. If Timer 0 is disabled, all flip-flops, counters, and outputs are in reset state except the ON-register, the TIMER0_RELOAD_N_REG register, and the TIMER0_RELOAD_M_REG register.

The timer input registers, that is, ON-register, TIMER0_RELOAD_N_REG, and TIMER0_RELOAD_M_REG can be written, and the counter registers ON-counter and T0-counter can be read. When reading from the address of the ON-register, the value of the ON-counter is returned. Reading from the address of either the TIMER0_RELOAD_N_REG or the TIMER0_RELOAD_M_REG register returns the value of the T0-counter.

It is possible to freeze Timer 0 with bit FRZ_SWTIM of the register SET_FREEZE_REG. When the timer is frozen, the timer counters are not decremented. This freezes all the timer registers at their last value. The timer continues its operation again when bit FRZ_SWTIM is cleared through register RESET_FREEZE_REG.

19.3 Programming

When LP clock is selected as system clock, CLK_AMBA_REG[HCLK_DIV] should be set to 0.

When LP clock is selected as Timer clock, CLK_PER_REG[TMR_DIV] should be set to 0

19.3.1 Timer Functionality

Timer 0 supports the functionality of a timer for generating interrupts after specific time intervals. To configure the timer operation:

1. Select the timer clock by programming TIMER0_CTRL_REG[TIM0_CLK_SEL]. The system or the LP clock can be selected using this option.
2. Select the clock division scaler by programming CLK_PER_REG[TMR_DIV]. Note that this setting only applies to the system clock.
3. Define whether Timer 0 will use the clock frequency as is or divided by 10 by programming TIMER0_CLK_REG[TIM0_CLK_DIV].
4. Enable Timer 0 clock by programming CLK_PER_REG[TMR_ENABLE].
5. For 16-bit counting, program TIMER0_ON_REG with the expired time in timer 0 clock cycles TIMER0_RELOAD_M_REG and TIMER0_RELOAD_N_REG must be set to 0.
6. For 17-bit counting, load 65535 to TIMER0_RELOAD_M_REG and the rest to TIMER0_RELOAD_N_REG. TIMER0_ON_REG must be set to 0.
7. Enable Timer 0 by programming TIMER0_CTRL_REG[TIM0_CTRL].

19.3.2 PWM Generation

Timer 0 also supports PWM generation. To configure the PWM generation functionality:

1. Select the timer clock by programming TIMER0_CTRL_REG[TIM0_CLK_SEL]. The system or the LP clock can be selected by this option.

2. Select the clock division scaler by programming CLK_PER_REG[TMR_DIV]. Note that this setting only applies to the system clock.
3. Select the PWM mode by programming TIMER0_CTRL_REG[PWM_MODE]. There are two modes supported. In the first one, the PWM signals are 1 during high time. In the second one, the PWM signals send out the (fast) clock divided by two during high time, so the clock frequency will be in the range of 1 to 8 MHz.
4. Enable Timer 0 clock by programming CLK_PER_REG[TMR_ENABLE].
5. Load the "high" value of the duty cycle in TIMER0_RELOAD_M_REG and the "low" value of the duty cycle in TIMER0_RELOAD_N_REG.
6. Set the desired GPIOs in PWM0/PWM1 and output mode.
7. Enable Timer 0 by programming TIMER0_CTRL_REG[TIM0_CTRL].

Timer 1

20.1 Introduction

Timer 1 is an 11-bit timer that can count up or down. It supports a free-running mode with an interrupt generated when zero is reached (also by wrapping around). It can be configured to use the system clock (sys_clk) or the LP clock (lp_clk) as the clock source. It supports capturing events on two GPIO channels when the number of clock cycles between these events is known. It can also generate an interrupt after a programmable number of clock cycles after an event. Figure 57 shows the block diagram of Timer 1.

Features

- 11-bit up/down counter with free running mode.
- Selectable system or LP clock as source.
- Two channels for capture input triggered by GPIOs.
- Capture capability from two GPIO events with programmable polarity.
- Programmable number of events between the two GPIOs for capturing.
- Timer 1 or RTC snapshot on capture events.
- Interrupt generation.

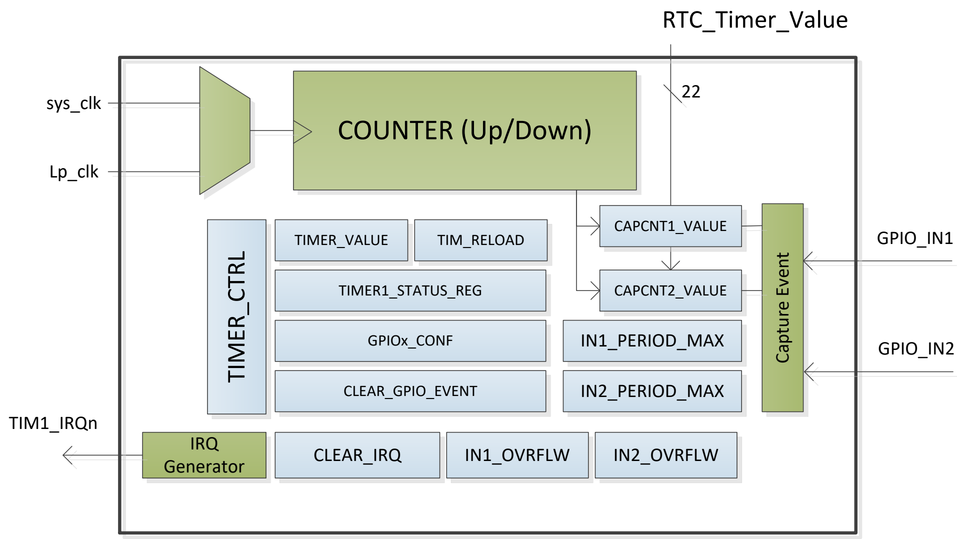

Figure 57. Timer 1 block diagram

20.2 Architecture

Timer 1 is placed in the PD_TIM power domain which can be kept powered even when the system power domain (containing the CPU) is shut down.

The main operation of Timer 1 is to count up or down, generating an interrupt when it reaches the maximum/minimum value or the threshold that has been programed as the reload value.

Moreover, Timer 1 comes with a sense block that allows for sensing positive or negative edges on two GPIOs (configurable). The sense block operates in two modes:

- **Counting mode**: Timer1 generates an interrupt upon a configurable amount of edges on a GPIO has been detected. The number of clock cycles was counted between timer start and captured the N - events is stored to CAPCNTx_VALUE register. When timer detects the first N-events, automatically starts to detect the next N-events until timer is disabled.
- **Capture mode**: Timer 1 saves a snapshot of either its own counter (11 bits) or the RTC port (22 bits) after an edge on a GPIO has been detected. If there is a pending interrupt, a new snapshot is not saved and the TIMER1_STATUS_REG[TIMER1_INx_OVRFLW] bit is set. This bit is cleared together with the TIMER1_STATUS_REG[TIMER1_Inx_EVENT] bit.

The same GPIO can be used for both modes.

If Timer 1 is used in the counting mode, it can measure the frequency applied to a GPIO port (see Section 20.3.3).

20.3 Programming

When LP clock is selected as system clock, CLK_AMBA_REG[HCLK_DIV] should be set to 0.

20.3.1 Timer Functionality

Timer 1 supports the functionality of a timer for generating interrupts after specific time intervals. To configure the timer functionality:

1. Select the timer clock by programming TIMER1_CTRL_REG[TIMER1_USE_SYS_CLK]. The system or the LP clock can be selected by this option.

NOTE
If the LP clock is selected as system clock, the CLK_AMBA_REG[HCLK] bit field should always be 0 to ensure the proper operation of the timer.

2. Enable or disable the free run mode by programming TIMER1_CTRL_REG[TIMER1_FREE_RUN_MODE_EN]. The free run mode can only be used when Timer 1 counts up.
3. Enable Timer 1 interrupt by programming TIMER1_CTRL_REG[TIMER1_IRQ_EN].
4. Set Timer 1 to count up or down by programming TIMER1_CTRL_REG[TIMER1_COUNT_DOWN_EN].
5. Specify the reload value by programming TIMER1_CTRL_REG[TIMER1_RELOAD].
6. Enable the Timer 1 clock by programming TIMER1_CTRL_REG[TIMER1_CLK_EN].
7. Enable Timer 1 by programming TIMER1_CTRL_REG[TIMER1_ENABLE].

20.3.2 Capture Functionality

Timer 1 can capture a snapshot of the value of RTC or Timer1 counts after a GPIO edge is detected. To configure the capture functionality:

1. Depending on the source of the snapshot value, configure and enable RTC or Timer 1 or both in the capture mode.
2. Set the edge type (rising or falling edge) by programming TIMER1_CAPTURE_REG[TIMER1_IN1_EVENT_FALL_EN] or TIMER1_CAPTURE_REG[TIMER1_IN2_EVENT_FALL_EN], depending on the channel that is used.
3. Set the timer in capture mode by setting TIMER1_CAPTURE_REG[TIMER1_IN1_COUNT_EN] = 0 or TIMER1_CAPTURE_REG[TIMER1_IN2_COUNT_EN] = 0, depending on the channel that is used.
4. Enable capture interrupt by setting TIMER1_CAPTURE_REG[TIMER1_IN1_IRQ_EN] = 1 or TIMER1_CAPTURE_REG[TIMER1_IN2_IRQ_EN] = 1, depending on the channel that is used.
5. Set the source of the snapshot value (RTC or Timer 1 count) by setting TIMER1_CAPTURE_REG[TIMER1_IN1_STAMP_TYPE] or TIMER1_CAPTURE_REG[TIMER1_IN2_STAMP_TYPE], depending on the channel that is used.
6. Set the GPIO that will be used to trigger the capture by setting TIMER1_CAPTURE_REG[TIMER1_GPIO1_CONF] or TIMER1_CAPTURE_REG[TIMER1_GPIO1_CONF], depending on the channel that is used.
   Note that the values from 1 to 12 define the P0 pins from 0 to 11.
7. When an interrupt is generated, the capture value will be saved in TIMER1_CAPCNT1_VALUE_REG[TIMER1_CAPCNT1_VALUE] or TIMER1_CAPCNT2_VALUE_REG[TIMER1_CAPCNT2_VALUE], depending on the channel that is used.
8. Write 1 to TIMER1_CLR_EVENT_REG[TIMER_CLR_Inx_EVENT] to clear the event.

20.3.3 Frequency Measuring Functionality

Timer 1 can measure the frequency applied to a GPIO port. To configure the frequency measure functionality:

1. Configure and enable Timer 1 in count up free mode using the system clock.

2. Set Timer 1 in count mode by setting TIMER1_CAPTURE_REG[TIMER1_IN1_COUNT_EN] = 1 or TIMER1_CAPTURE_REG[TIMER1_IN2_COUNT_EN] = 1, depending on the channel that is used.
3. Enable capture interrupt by setting TIMER1_CAPTURE_REG[TIMER1_IN1_IRQ_EN] = 1 or TIMER1_CAPTURE_REG[TIMER1_IN2_IRQ_EN] = 1, depending on the channel that is used.
4. Set the rising edge by programming TIMER1_CAPTURE_REG[TIMER1_IN1_EVENT_FALL_EN] = 0 or TIMER1_CAPTURE_REG[TIMER1_IN2_EVENT_FALL_EN] = 0, depending on the channel that is used.
5. Set the number of periods plus one, in which Timer 1 counts, by setting TIMER1_CAPTURE_REG[TIMER1_IN1_PERIOD_MAX] or TIMER1_CAPTURE_REG[TIMER1_IN2_PERIOD_MAX], depending on the channel that is used.
6. Set the GPIO that will be used to trigger the capture by setting TIMER1_CAPTURE_REG[TIMER1_GPIO1_CONF] or TIMER1_CAPTURE_REG[TIMER1_GPIO1_CONF], depending on the channel that is used.
   Note that the values from 1 to 12 define the P0 pins from 0 to 11.
7. After the interrupt is triggered, read the value in TIMER1_CAPCNT1_VALUE_REG[TIMER1_CAPCNT1_VALUE] or TIMER1_CAPCNT2_VALUE_REG[TIMER1_CAPCNT2_VALUE], depending on the channel that is used.
   This value indicates the number of cycles that have passed during the period defined in step 5.
8. To calculate the frequency applied to the GPIO, divide the number of periods (step 5) by the cycles (step 7) and multiply the result with the frequency of Timer 1 clock.
9. Write 1 to TIMER1_CLR_EVENT_REG[TIMER_CLR_Inx_EVENT] to clear the event.

Timer 2

21.1 Introduction

Timer 2 is basically a PWM generator. It has six PWM outputs. Figure 58 shows the block diagram.

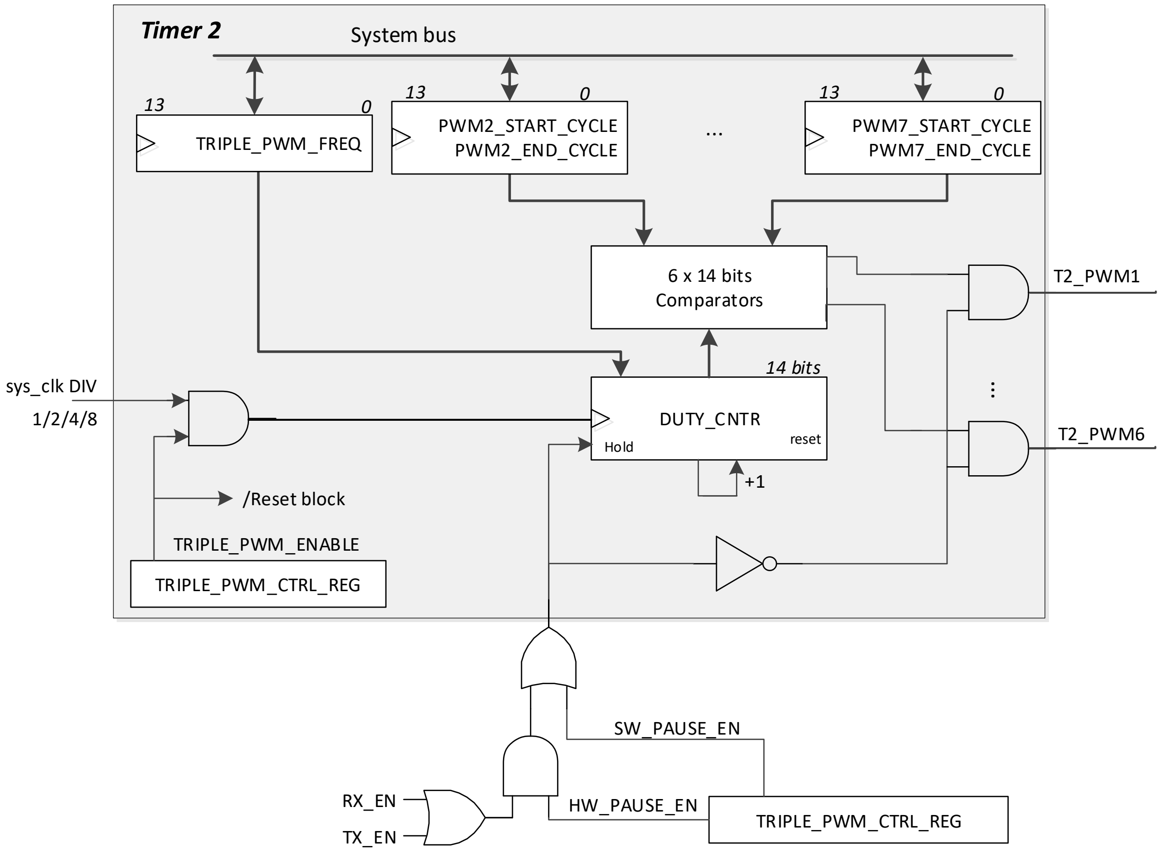

Figure 58. Timer 2 block diagram

Features

- 14-bit general purpose timer
- Ability to generate six PWM signals (PWM2, PWM3, PWM4, PWM5, PWM6, and PWM7,)
- Input clock frequency (f_{IN}) with N = 1, 2, 4, or 8 and sys_clk = 16 MHz or 32 kHz:

$$f_{IN} = \frac{sys\_clk}{N}$$

- Programmable output frequency (f_{OUT}):

$$f_{OUT} = \left(\frac{f_{IN}}{2}\right) \text{ to } \left(\frac{f_{IN}}{2^{14}-1}\right)$$

- Six outputs with a programmable duty cycle from 0% to 100%
- Used for white LED intensity (on/off) control or motor control.

21.2 Architecture

Timer 2 is clocked with the system clock divided by TMR_DIV (1, 2, 4, or 8) and can be enabled with TRIPLE_PWM_CTRL_REG[TRIPLE_PWM_ENABLE].

TRIPLE_PWM_FREQUENCY determines the output frequency of the PWM outputs.

NOTE
There is a single frequency register for all six PWM outputs.

DUTY_CNTR is an up-counter counting from 0 up to TRIPLE_PWM_FREQUENCY.

If DUTY_CNTR is equal to the value stored in the respective PWMn_END_CYCLE register, it resets the PWMn output to 0.

If DUTY_CNTR is equal to the value stored in the respective PWMn_START_CYCLE register, it sets the PWMn output to 1.

Note that the value of PWMn_END_CYCLE and PWMn_START_CYCLE must be less than or equal to TRIPLE_PWM_FREQUENCY.

The Timer 2 is enabled/disabled by programming the TRIPLE_PWM_CTRL_REG[TRIPLE_PWM_EN] bit.

The timing diagram of Timer 2 is shown in Figure 59.

Freeze function

During RF activity it may be desirable to temporarily suppress the PWM switching noise. This can be done by setting TRIPLE_PWM_CTRL_REG[HW_PAUSE_EN] = 1. The effect is that whenever there is a transmission or a reception process from the Radio, DUTY_CNTR is frozen and PWMx output is switched to 0 to disable the selected PWMn. As soon as the Radio is idle, that is, RX_EN or TX_EN signals are zero, DUTY_CNTR resumes counting and finalizes the remaining part of the PWM duty cycle.

TRIPLE_PWM_CTRL_REG[SW_PAUSE_EN] can be set to 0 to disable the automatic, hardware driven freeze function of the duty counter and keep the duty cycle constant.

Note that the RX_EN and TX_EN signals are not software driven but controlled by the Bluetooth® LE core hardware.

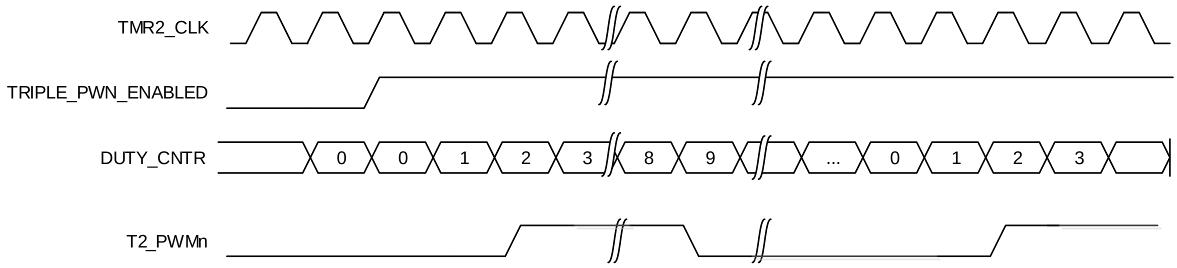

Figure 59. Timer 2 timing diagram

21.3 Programming

When LP clock is selected as system clock, CLK_AMBA_REG[HCLK_DIV] should be set to 0.

When LP clock is selected as Timer clock, CLK_PER_REG[TMR_DIV] should be set to 0.

21.3.1 PWM Generation

Timer 2 only supports PWM generation and does not support normal, interrupt generating, timer functionality as the previous timers do. To configure the PWM generation functionality:

1. Select the clock source for Timer 2 by programming TRIPLE_PWM_CTRL_REG[TRIPLE_PWM_CLK_SEL]. System clock or LP clock can be selected. Note that if the (fast) system clock is selected, the division scaler is the same as the one for Timer 0.
2. Define the GPIOs to which the PWM signals are mapped by programming the respective PID number.
3. Define the frequency of Timer 2 that feeds the PWM waveforms by programming the TRIPLE_PWM_FREQUENCY with a value that conforms to the following equation. For example, if Timer 2 clock is 32000 Hz (lp_clk) and the required frequency for the PWM is 16 kHz, this register should be written with 0x1.

$$\mathrm{Timer2\_clk\_freq\_Hz}/\mathrm{Required\_freq\_Hz} - 1 \tag{4}$$

NOTE
There is a single frequency register for all six PWM outputs.

4. Define the duty cycle of each PWM signal. Program the start and end cycle of the pulse at PWMx_START_CYCLE and PWMx_END_CYCLE, respectively. The available amount of cycles is depicted in the contents of TRIPLE_PWM_FREQUENCY register. For example, if the TRIPLE_PWM_FREQUENCY has a value of 0x8 and the START/END_CYCLE bit fields have a value of 3 and 5, respectively, the PWM signals will rise after three Timer 2 clock cycles and fall after five clock cycles. Every PWM signal has its own register to configure its duty cycle.
5. Enable the PWM signals by programming TRIPLE_PWM_CTRL_REG[TRIPLE_PWM_ENABLE] = 1.

21.3.2 Freeze Functionality

There is provision to allow hardware to pause PWM signals while RF is active. This can be done by programming TRIPLE_PWM_CTRL_REG[HW_PAUSE_EN] = 1. It can also be done through software control by programming TRIPLE_PWM_CTRL_REG[SW_PAUSE_EN] = 1.

22. Watchdog Timer

22.1 Introduction

The Watchdog Timer is an 8-bit timer with a sign bit that can be used to detect an unexpected execution sequence caused by a software run-away and can generate a full system reset (WDOG reset) or a Non-Maskable Interrupt (NMI). Figure 60 shows the block diagram of the Watchdog Timer.

Features

- 8-bit down counter with a sign bit, clocked with a 10.24 ms clock for a maximum 2.6 s time-out.
- Non-Maskable Interrupt (NMI) or WDOG reset.
- Optional automatic WDOG reset if NMI handler fails to update the Watchdog register.
- Non-maskable Watchdog freeze of the Cortex-M0+ Debug module when the Cortex-M0+ is halted in Debug state. Maskable Watchdog freeze by user program.

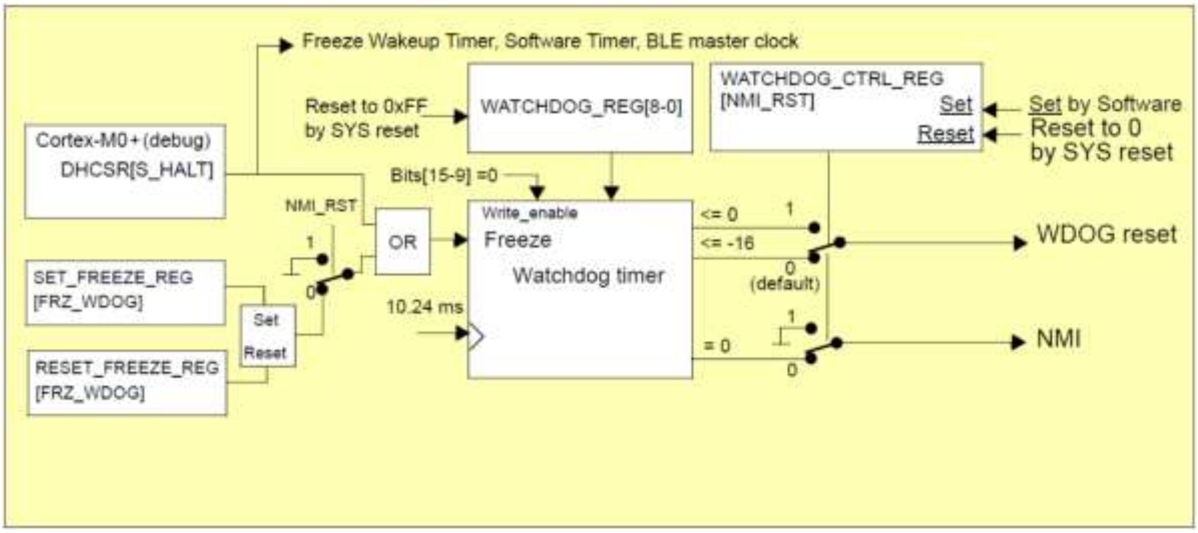

Figure 60. Watchdog timer block diagram

22.2 Architecture

The 8-bit watchdog timer is decremented by 1 every 10.24 ms. The timer value can be accessed through the WATCHDOG_REG register which is set to 255 (FF_{16}) at reset. This results in a maximum watchdog time-out of ~2.6 s. During write access, the WATCHDOG_REG[WDOG_WEN] bit must be 0. This provides extra filtering for a software run-away by writing ones to all the bits in the WATCHDOG_REG. If the watchdog timer reaches 0, its value gets a negative value by setting bit 8. The counter sequence becomes 1, 0, $1FF_{16}$ (-1), $1FE_{16}$(-2), till $1F0_{16}$ (-16).

If WATCHDOG_CTRL_REG[NMI_RST] = 0, the watchdog timer generates an NMI when it reaches 0 and a WDOG reset when it becomes less or equal to -16 ($1F0_{16}$). The NMI handler must write a value that is larger than -16 to the WATCHDOG_REG to prevent the generation of a WDOG reset when the watchdog timer reaches the value -16 after 16 × 10.24 = 163.8 ms.

If WATCHDOG_CTRL_REG[NMI_RST] = 1, the watchdog timer generates a WDOG reset when it becomes less than or equal to 0.

The WDOG reset is one of the system (SYS) reset sources and resets almost the whole device, including resetting the WATCHDOG_REG register to 255. For an overview of the complete reset circuit and conditions, see the POR, hardware, and software reset Section.

For debugging purposes, the Cortex-M0+ Debug module can always freeze the watchdog by setting the DHCSR[DBGKEY | C_HALT | C_DEBUGEN] control bits (reflected by the status bit S_HALT, see Table 42). This is automatically done by the debugging tool, for example, during step-by-step debugging. Note that this bit also freezes the Wake-up Timer, the Software Timer, and the Bluetooth LE master clock. For additional information see the DEBUG_REG[DEBUGS_FREEZE_EN] mask register. The C_DEBUGEN bit cannot be accessed by the user software so that freezing the watchdog is prevented.

In addition to the S_HALT bit, the watchdog timer can also be frozen if NMI_RST = 0 and SET_FREEZE_REG[FRZ_WDOG] is set to 1. The watchdog timer resumes counting when RESET_FREEZE_REG[FRZ_WDOG] is set to 1. The WATCHDOG_CTRL_REG[NMI_RST] bit can only be set by software and is only reset on a SYS reset. Note that if the system is not remapped, that is, the SysRAM is at address 0x07FC0000, a watchdog fire triggers the BootROM code to be executed again.

22.3 Programming

To program the Watchdog Timer:

1. Freeze watchdog by setting the SET_FREEZE_REG[FRZ_WDOG] bit (optional).
2. Select NMI and reset events (WATCHDOG_CTRL_REG[NMI_RST]).
3. Enable writing of the watchdog timer (WATCHDOG_REG[WDOG_WEN] = 0]).
4. Write the reload value of the watchdog timer (WATCHDOG_REG[WDOG_VAL], see the register description).
5. Resume watchdog (RESET_FREEZE_REG[FRZ _WDOG] = 1), if frozen.

23. Temperature Sensor

23.1 Introduction

The DA14535 features a built-in temperature sensor.

Features

- Temperature range -40 °C to 105 °C
- Absolute accuracy after one-point calibration +/- 4 °C (assuming 25 °C reference temperature)
- 25 °C single point calibration reference value provided in OTP memory.

23.2 Architecture

The temperature sensor can be read out through the GP_ADC.

Figure 61 shown the relationship between the actual ambient temperature and the calculated temperature from the GP_ADC readout, including possible inaccuracies in $T_{SENSE\_ACC\_OTP}$ (offset) and TC_{SENSE} (angle).

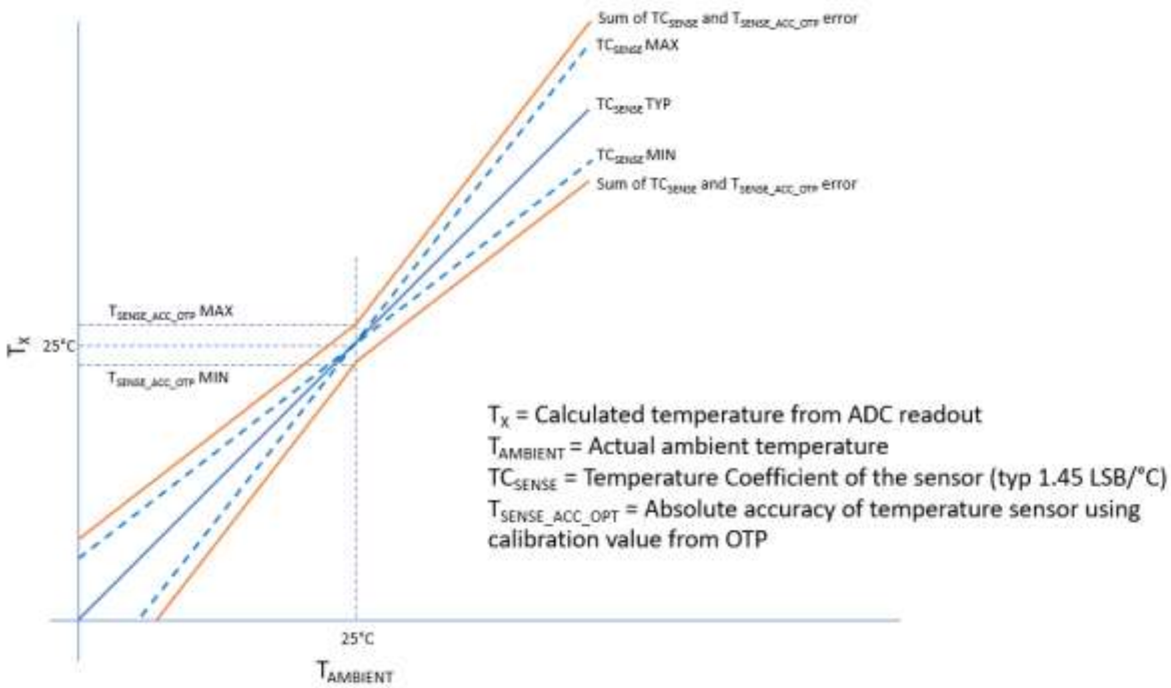

Figure 61. Temperature sensor behavior

The recommended formula for single point calibrated temperature reading is as follows:

$$\mathbf{T_x = 25 + (ADC_x - ADC_{OTP\_CAL\_25C})/(TC_{SENSE} \times 64)}$$

Where:

- T_x = calculated single point calibrated die temperature in [°C]
- ADC_x = 16-bit GP_ADC_VAL readout (converted to decimal) at temperature T_x
- $ADC_{OTP\_CAL\_25C}$ = 25 °C OTP calibration value recorded during production testing (based on the 16-bit readout)
- TC_{SENSE} = temperature coefficient in [LSB/°C], typical value is 1.45 LSB/°C
- 25 = reference base value in [°C]
- 64 = correction for 16-bit to 10-bit ADC values

For uncalibrated temperature sensor measurements, ADC_{OTP_CAL25C} can be replaced by the default value using the formula below:

$T_x = 25 + (ADC_x - 30272)/(TC_{SENSE} \times 64)$

Note that this is not recommended because it can result in large offsets.

NOTE
While measuring and/or calibration, the system's power dissipation should be kept the same, otherwise, the measurement is affected by the internal thermal gradient.

23.3 Programming

There is a certain programming sequence required to read the temperature sensor. There are two reading options available: absolute temperature (single-point calibration) and relative temperature.

23.3.1 Absolute Temperature

A calibration value at 25 °C is stored in OTP for absolute temperature measurements. When the calibration value from OTP is used, the default GP_ADC offset calibration settings should be used.

- To enable OTP in normal read mode:
  - CLK_AMBA_REG[OTP_ENABLE] = 1
  - OTPC_MODE_REG[OTPC_MODE_MODE] = 2
- To read the calibration value at 25 °C:
  - Read $ADC_{OTP\_CAL\_25C}$: the content of $ADC_{OTP\_CAL\_25C}$ is at the address 0x7F87F28 of the OTP
- To disable OTP:
  - OTPC_MODE_REG[OTPC_MODE_MODE] = 0
  - CLK_AMBA_REG[OTP_ENABLE] = 0
- To read back the offset calibration:
  - Offp = GP_ADC_OFFP_REG[GP_ADC_OFFP]
  - Offn = GP_ADC_OFFN_REG[GP_ADC_OFFN]

(Store the data if the original values are need later for the application)

- To overwrite the defaults (the settings during factory calibration) with the ADC offset values:
  - GP_ADC_OFFP_REG[GP_ADC_OFFP] = 200
  - GP_ADC_OFFN_REG[GP_ADC_OFFN] = 200
- To enable the temperature sensor:
  - GP_ADC_CTRL_REG[DIE_TEMP_EN] = 1
- Wait 25 µs for the temperature sensor to start up
- To set the advised ADC settings:
  - GP_ADC_TRIM_REG[GP_ADC_LDO_LEVEL] = 4
  - GP_ADC_CTRL_REG[GP_ADC_CHOP] = 1
  - GP_ADC_CTRL_REG[GP_ADC_SE] = 1
  - GP_ADC_CTRL_REG[GP_ADC_EN] = 1
  - GP_ADC_CTRL2_REG[GP_ADC_I20U] = 1
  - GP_ADC_SEL_REG[GP_ADC_SEL_P] = 4
  - GP_ADC_CTRL2_REG[GP_ADC_STORE_DEL] = 0
- To set sample time and averaging of the ADC sampling:
  - GP_ADC_CTRL2_REG[GP_ADC_SMPL_TIME] = F
  - GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS] = 6
- To perform ADC conversion:
  - GP_ADC_CTRL_REG[GP_ADC_START] = 1

- To wait for the conversion to finish, read the register
  - GP_ADC_RESULT_REG[GP_ADC_VAL]
- To write back the original offset values:
  - GP_ADC_OFFP_REG[GP_ADC_OFFP] = Offp
  - GP_ADC_OFFN_REG[GP_ADC_OFFN] = Offn

(Restore the original data if need by the application)

23.3.2 Relative Temperature

For relative temperature measurements, single-point calibration is not needed. The programming sequence is the following:

- To enable GP_ADC:
  - GP_ADC_CTRL_REG[DIE_TEMP_EN] = 1
- Wait 25 μs for the temperature sensor to start up
- To set the advised ADC settings:
  - GP_ADC_TRIM_REG[GP_ADC_LDO_LEVEL] = 4
  - GP_ADC_CTRL_REG[GP_ADC_CHOP] = 1
  - GP_ADC_CTRL_REG[GP_ADC_SE] = 1
  - GP_ADC_CTRL_REG[GP_ADC_EN] = 1
  - GP_ADC_CTRL2_REG[GP_ADC_I20U] = 1
  - GP_ADC_SEL_REG[GP_ADC_SEL_P] = 4
  - GP_ADC_CTRL2_REG[GP_ADC_STORE_DEL] = 0
- To set sample time and averaging of the ADC sampling
  - GP_ADC_CTRL2_REG[GP_ADC_SMPL_TIME] = F
  - GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS] = 6
- To perform ADC conversion:
  - GP_ADC_CTRL_REG[GP_ADC_START] = 1
- To wait for the conversion to finish, read the register
  - GP_ADC_RESULT_REG[GP_ADC_VAL]

24. Keyboard Controller

24.1 Introduction

The Keyboard controller can be used for debouncing the incoming GPIO signals when implementing a keyboard scanning engine. It generates an interrupt to the CPU (KEYBR_IRQ).

In parallel, five extra interrupt lines can be triggered by a state change on up to nine selectable GPIOs (GPIO_IRQx).

Features

- Monitors the 9 available GPIOs.
- Generates a keyboard interrupt on key press or key release.
- Implements debouncing time from 0 up to 63 ms.
- Supports five separate interrupt generation lines from GPIO toggling.

Figure 62 shows the block diagram of the Keyboard Controller.

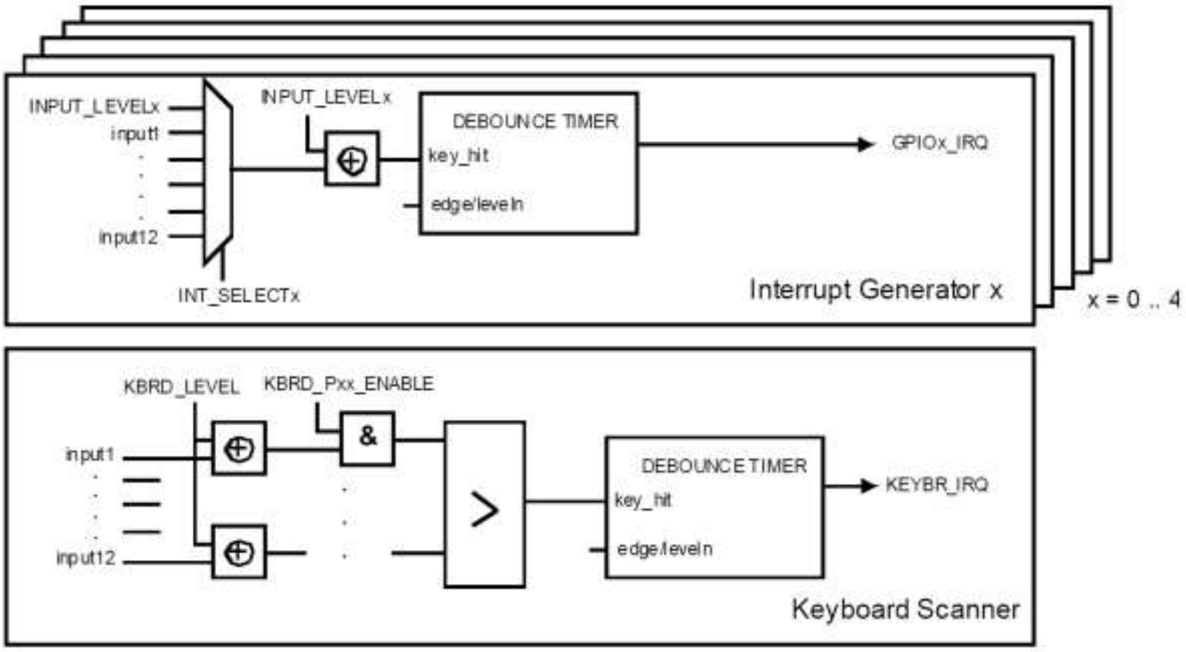

Figure 62. Keyboard controller block diagram

24.2 Architecture

24.2.1 Keyboard Scanner

A HIGH-to-LOW transition on one of the GPIO inputs sets the internal signal "key_hit" to 1, while KBRD_IRQ_IN_SEL0_REG[KBRD_LEVEL] = 0 and KBRD_IRQ_IN_SELx_REG[KBRD_Pyy_EN] = 1. This signal triggers the state machine of the keyboard interface shown in Figure 63. The debounce timer is loaded with the value of GPIO_DEBOUNCE_REG[DEB_VALUE]. The timer counts down every 1 ms. When the timer reaches 0 and the "key_hit" signal is still 1, the timer is loaded with the value of KBRD_IRQ_IN_SEL0_REG[KEY_REPEAT], generating a repeating sequence of interrupts every time when the timer reaches 0.

When the key is released (key_hit = 0) and the bit KBRD_REL (key release) is set to 1, a new debounce sequence is started and a KEYBR_IRQ interrupt is generated after the debounce time.

The debounce timer can be disabled with GPIO_DEBOUNCE_REG[DEB_ENABLE_KBRD] = 0. The key repeat function can be disabled by setting KEY_REPEAT to 0.

The level for generating an interrupt is programmable through bit KBRD_IRQ_IN_SEL0_REG[KBRD_LEVEL]. The key release function can be disabled by setting bit KBRD_IRQ_IN_SEL0_REG[KBRD_REL] to 0. The inputs for the keyboard interface can be selected by setting the corresponding bits KBRD_IRQ_IN_SEL0_REG[KBRD_Pxx_EN] to 1.

The keyboard interrupt service routine can distinguish which input has caused the interrupt by reading the Px_DATA_REG registers.

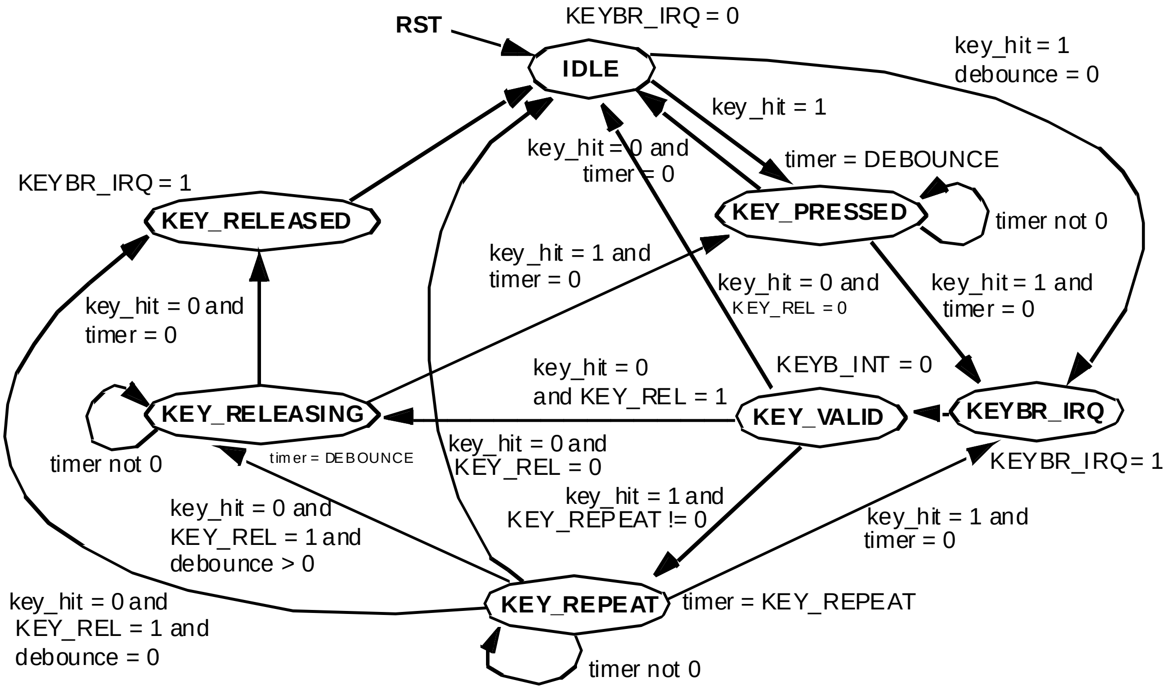

Figure 63. Keyboard scanner state machine

24.2.2 GPIO Interrupt Generator

Five identical GPIO interrupt generators support the generation of up to five interrupts (GPIO0_IRQ to GPIO4_IRQ). One of the GPIO inputs can be selected to generate an interrupt by programming the corresponding GPIO_IRQx_IN_SEL_REG register. The input level can be selected by GPIO_INT_LEVEL_CTRL_REG[INPUT_LEVELx].

A LOW-to-HIGH level transition on one of the GPIO inputs sets the internal signal "key_hit" to 1, while the bit INPUT_LEVELx = 0. This signal triggers the state machine of the GPIO Interrupt Generator shown in Figure 64. The debounce timer is loaded with the value of GPIO_DEBOUNCE_REG[DEB_VALUE]. The timer counts down every 1 ms. If the timer reaches 0 and the "key_hit" signal is still 1, an interrupt is generated. The debounce timer for each interrupt can be disabled with GPIO_DEBOUNCE_REG[DEB_ENABLEx].

The interrupt flag remains set until it is reset by writing to the corresponding bit in the GPIO_RESET_IRQ_REG register. If the GPIO interrupt is edge sensitive selected with bit GPIO_INT_LEVEL_CTRL_REG[EDGE_LEVELNx], the state machine progresses to the state WAIT_FOR_RELEASE when the interrupt is reset. It progresses to the IDLE state only after the non-active edge is detected.

To detect both signal edges, the edge polarity INPUT_LEVELx must be inverted in the WAIT_FOR_RELEASE state. This results in "key_hit" = 0 and advances the state machine to the IDLE state, allowing the next inverted edge to be detected.

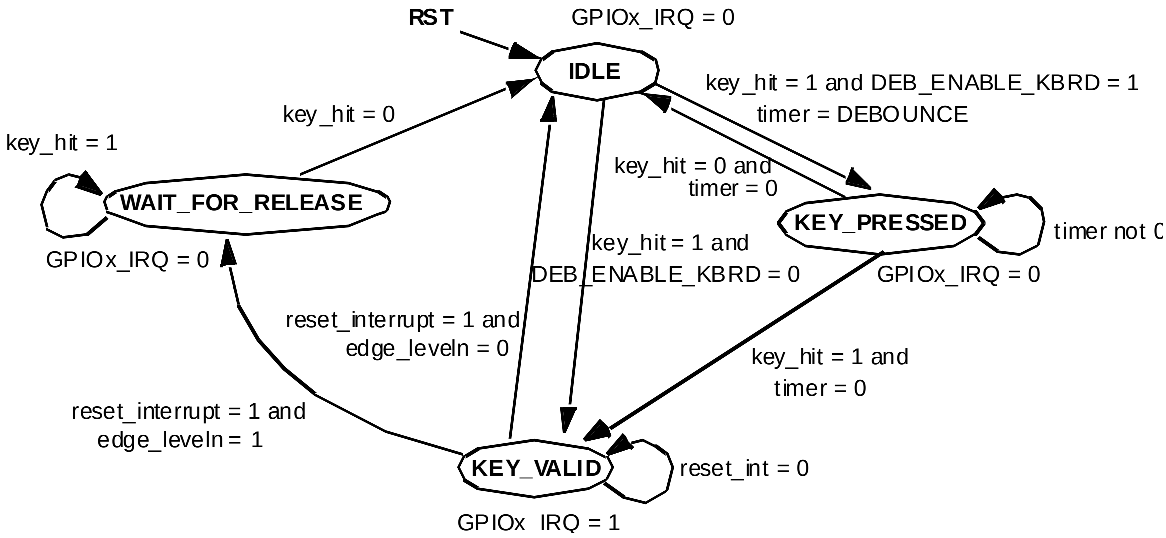

Figure 64. GPIO interrupt generator state machine

24.3 Programming

To configure and use the keyboard controller, follow the steps under each subsection.

24.3.1 Keyboard Scanner

1. Enable a keyboard interrupt for the P0_x by setting the KBRD_IRQ_IN_SEL0_REG[KBRD_Px_EN] bit.
2. Select the logic level by which the interrupt is generated (KBRD_CTRL_REG[KBRD_LEVEL]).
3. Select whether a key release also generates an interrupt (KBRD_CTRL_REG[KBRD_REL]).
4. Select whether repeated interrupts are generated when a key is held pressed (KBRD_CTRL_REG[KEY_REPEAT]).
5. Set up the debounce time for each key stroke (GPIO_DEBOUNCE_REG[DEB_VALUE]).
6. Enable the debounce timer (GPIO_DEBOUNCE_REG[DEB_ENABLE_KBRD]).

24.3.2 GPIO Interrupts

1. Enable a GPIO interrupt for the P0_x by setting the GPIO_IRQx_IN_SEL_REG[KBRD_IRQ0_SEL] bit.
2. Select the logic level by which the interrupt is generated (GPIO_INT_LEVEL_CTRL_REG[INPUT_LEVELx]).
3. Select whether a key release is needed for an interrupt to be generated after a generated IRQ is cleared (GPIO_INT_LEVEL_CTRL_REG[EDGE_LEVELNx]).
4. Set up the debounce time for GPIO trigger (GPIO_DEBOUNCE_REG[DEB_VALUE]).
5. Enable the debounce timer for the selected IRQ (GPIO_DEBOUNCE_REG[DEB_ENABLEx]).

25. Input/Output Ports

25.1 Introduction

The DA14535 has an I/O pin assignment that can be configured by the software and is organized into the Port 0. Pins from P0_0 to P0_11 are available on the FCGQFN24 package. Figure 65 shows the block diagram of the IO and its programmability options.

Features

- Twelve GPIOs (including RST, SW_CLK, SWDIO, XTAL32Km, and XTAL32Kp).
- Fully programmable pin assignment.
- Selectable 25 kΩ pull-up and pull-down resistors per pin.
- Programmable driving strength outputs.
- Fixed assignment for analog pin ADC[3:0].
- Pins can retain their last state when system enters a Sleep mode.

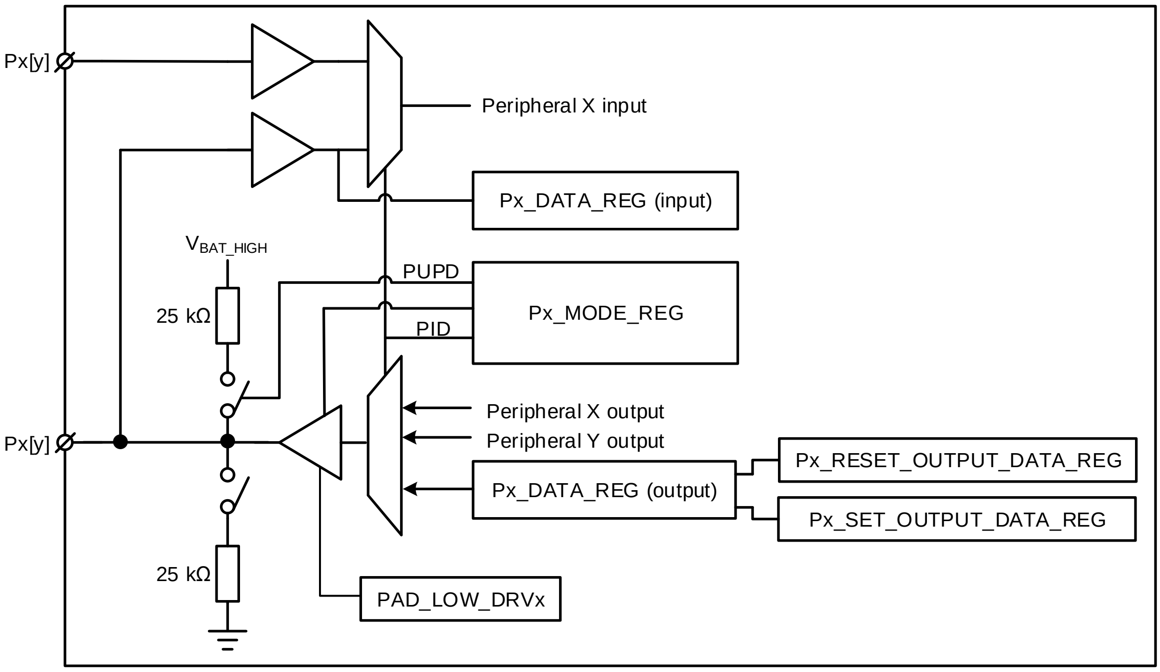

Figure 65. Port P0 with programmable pin assignment and driving strength

25.2 Architecture

25.2.1 Programmable Pin Assignment

The Programmable Pin Assignment (PPA) provides a multiplexing function to the I/O pins of on-chip peripherals. Any peripheral input or output signal can be freely mapped to any I/O port bit by setting Pxy_MODE_REG[4-0]:

```
0x00 to 0x1F: Peripheral IO ID (PID)
```

See the registers of Px_MODE_REG (x = 00, 01, 02, to 11) for an overview of the available PIDs. The analog ADC has a fixed pin assignment so that the interference with the digital domain is limited. The SWD interface (JTAG) is mapped on P0_10 for the SWDIO and on P0_2 for the SWCLK.

25.2.1.1 Priority

The firmware can assign the same peripheral output to more than one pin. It is the users' responsibility to make a unique assignment.

If more than one input signal is assigned to a peripheral input, the left most pin in the lowest port pin number has priority.

25.2.1.2 Direction Control

The port direction is controlled by setting Pxy_MODE_REG[9-8] to:

- 00 = Input, no resistors selected.
- 01 = Input, pull-up resistors selected.
- 10 = Input, pull-down resistors selected.
- 11 = Output, no resistors selected.

In output mode and analog mode, the pull-up/down resistors are automatically disabled.

25.2.2 General-Purpose Port Registers

The general-purpose ports are selected with PID = 0. The port function is accessible through registers:

- Px_DATA_REG: Port data input/output register.
- Px_SET_OUTPUT_DATA_REG: Port set output register.
- Px_RESET_OUTPUT_DATA_REG: Port reset output register.

25.2.2.1 Port Data Register

The registers input Px_DATA_REG and output Px_DATA_REG are mapped to the same address.

The data input register (Px_DATA_REG) is a read-only register that returns the current state on each port pin, even if the output direction is selected, regardless of the programmed PID, unless the analog function is selected (in this case it reads 0). The Cortex CPU can read this register at any time, even when the pin is configured as an output.

The data output register (Px_DATA_REG) holds the data to be driven on the output port pins. In this configuration, writing to this register changes the output value.

25.2.2.2 Port Set Data Output Register

Writing a 1 in the set data output register (Px_SET_DATA_REG) sets the corresponding output pin. Writing a 0 is ignored.

25.2.2.3 Port Reset Data Output Register

Writing a 1 in the reset data output register (Px_RESET_DATA_REG) resets the corresponding output pin. Writing a 0 is ignored.

25.2.3 Fixed Assignment Functionality

Certain signals have a fixed mapping on specific general purpose IOs. This assignment is shown in Table 48.

Table 48: Fixed assignment of specific signals

GPIO	Reset/SWD (Note 1)	Quadrature decoder (Note 2)	ADC (Note 3)
P0_0	RST	CH6_A	
P0_1	SWDIO (alternative)	CH1_A	ADC_0
P0_2	SWCLK	CH1_B	ADC_1
P0_3		CH2_A	
P0_4		CH2_B	
P0_5	SWDIO (alternative)	CH3_A	
P0_6		CH3_B	ADC_2
P0_7		CH4_A	ADC_3
P0_8		CH6_B	
P0_9		CH5_A	
P0_10	SWDIO	CH5_B	
P0_11		CH4_B	

Note 1 The SWD signal mapping is defined by SYS_CTRL_REG[DEBUGGER_ENABLE]. However, these signals are mapped on the ports by default. The alternative SWD mapping is selected by the SYS_CTRL_REG[DEBUGGER_ENABLE] bit field. The RST default functionality can be disable by the HWR_CTRL_REG[DISABLE_HWR] bit.

Note 2 The mapping of the quadrature decoder signals on the respective pins is overruled by the QDEC_CTRL2_REG[CHx_PORT_SEL] register.

Note 3 The ADC function can be selected by the PID bit field on the respective Px port.

25.2.4 Types of GPIO Pads

There are two different types for the GPIO pads, namely, type A and type B. Their block diagrams are presented in Figure 66 and Figure 67.

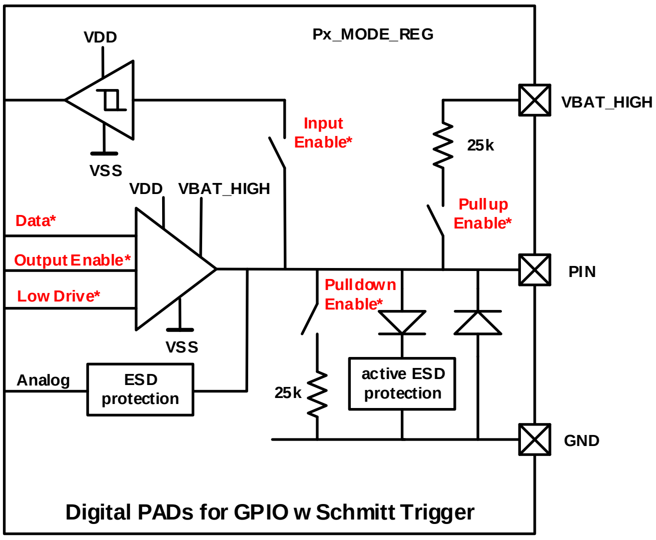

Figure 66. Type A GPIO pad – GPIO with Schmitt trigger on input

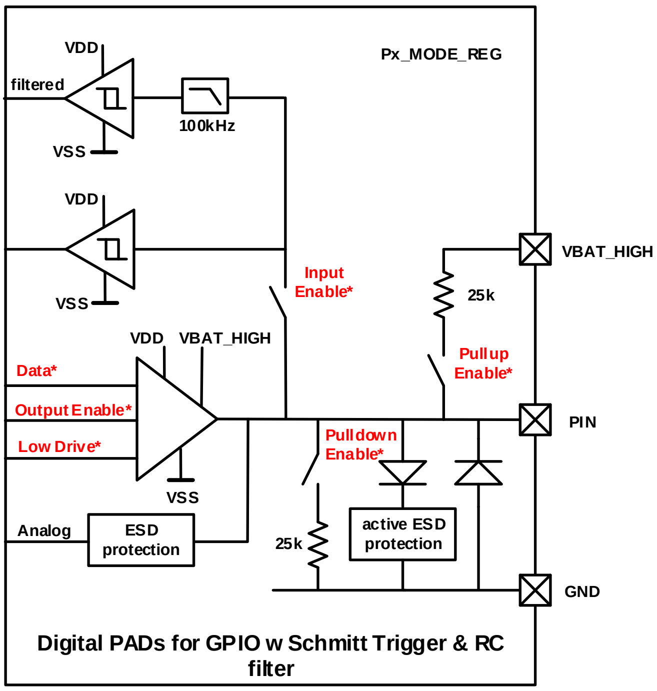

Figure 67. Type B GPIO pad – GPIO with Schmitt trigger and RC filter on input

Red signals are latched when the system enters a Sleep mode.

25.2.5 Driving Strength

Pads can be configured regarding their driving capability using PAD_WEAK_CTRL_REG. There are only two levels available for the load that the pad can support, namely normal = 3.5 mA typical, and reduced = 0.35 mA typical.

25.2.6 Special I/O Considerations

When in boost mode, it is recommended that P0_1 is used as the chip select signal for the external SPI Flash memory. Moreover, to ensure a proper boot sequence after a hardware reset (when the device is in Boost configuration and P0_1 is used as chip select signal for an external SPI Flash), the PULL_HW_BYPASS_REG[PULL_HW_BYPASS] bit (reset value is 0x1 but the booter clears this bit) must be set before the PMU_CTRL_REG[RESET_ON_WAKEUP] is set.

26. General-Purpose ADC

26.1 Introduction

The DA14535 is equipped with a high-speed ultra-low-power 10-bit general purpose Analog-to-Digital Converter (GPADC). It can operate in unipolar (single ended) mode as well as in bipolar (differential) mode. The ADC has its own voltage regulator (LDO) of 0.9 V, which represents the full-scale reference voltage. Figure 68 shows the block diagram of the GPADC.

Features

- 10-bit dynamic ADC with 125 ns typical conversion time
- Maximum sampling rate 1 Msample/s
- 128× averaging; conversion time 1 ms, up to 11b ENOB
- Ultra-low power (20 µA typical supply current at 100 ksample/s)
- Four single-ended or two differential external input channels (GPIOs)
- Battery, DCDC outputs, and the internal V_{DD} monitoring channels
- Chopper function
- Offset adjust
- Common-mode input level adjust
- Configurable attenuator: 1×, 2×, 3× and 4×.

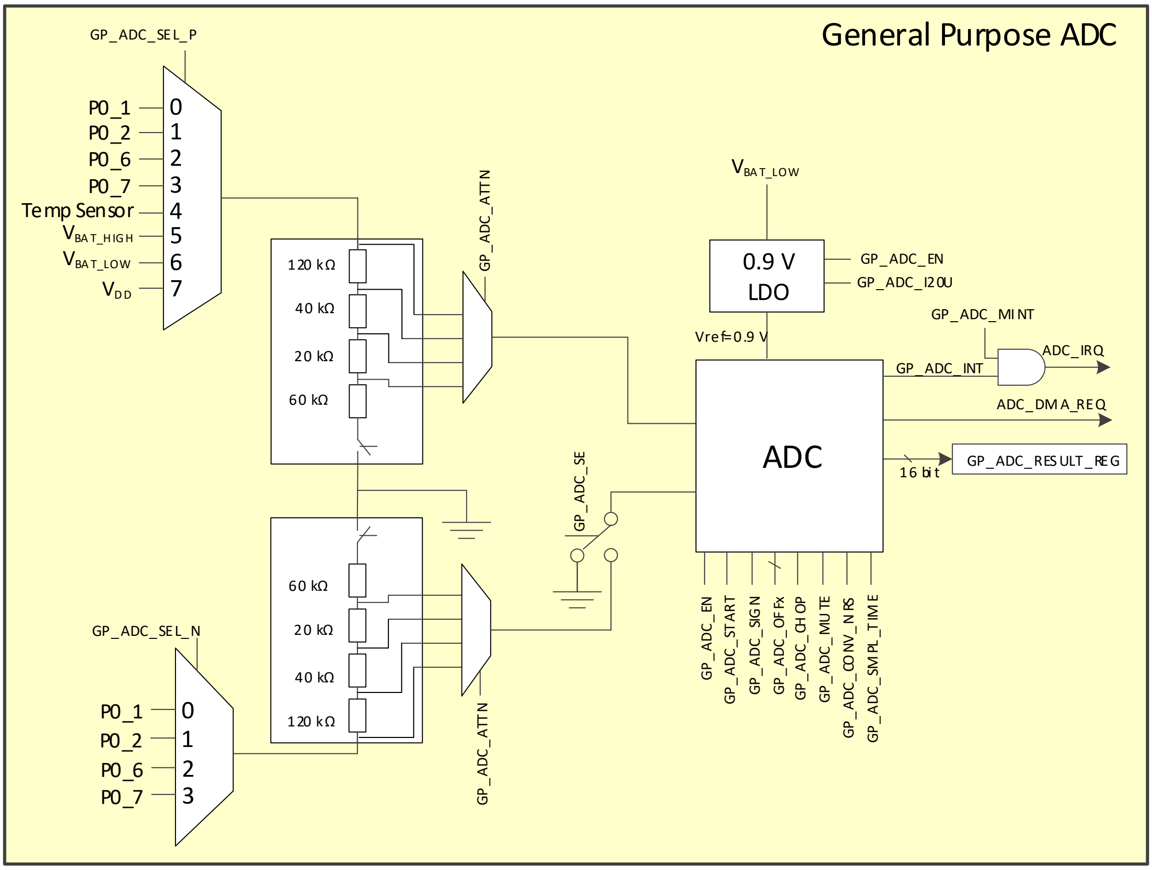

Figure 68. Block diagram of GPADC

26.2 Architecture

The ADC architecture shown in Figure 68 has the following sub-blocks:

- Analog to Digital converter (ADC):
  - ADC analog part internally clocked with 100 MHz.
  - ADC logic part clocked with the ADC_CLK which is the 16 MHz system clock (sys_clk).
- 0.9 V LDO for the ADC supply with a high PSRR enabled with GP_ADC_CTRL_REG[GP_ADC_EN].
- Configurable attenuator with 1×, 2×, 3×, and 4× attenuation controlled by GP_ADC_CTRL2_REG[GP_ADC_ATTN].
- APB Bus interface clocked with the APB clock. Control and status registers are available through registers GP_ADC_*
- Maskable Interrupt (ADC_IRQ) and DMA request (ADC_DMA_REQ).
- ADC input channel selector. Up to four GPIO ports, the battery and DCDC output (V_{BAT_HIGH} and V_{BAT_LOW}), the internal V_{DD}, and the analog ground level (AVS) can be measured.

26.2.1 Input Channels

Table 49 summarizes the ADC input channels. The GPIO signals at the channels [3:0] can be monitored both single-ended and differentially. The signals at the 4-7 inputs can be monitored single-ended or differentially with respect to the GPIOs.

Table 49: ADC input channels

Channel	Signal	Description
3:0	GPIO [P0_1, P0_,2, P0_6, P0_7]	General Purpose Inputs
4	Temperature Sensor	Temperature Sensor
5	V_{BAT_HIGH}	V_{BAT_HIGH} rail
6	V_{BAT_LOW}	V_{BAT_LOW} rail
7	V_{DD}	V_{DD} rail for the digital power domain

Table 50 summarizes the voltage ranges which can be handled with the single-ended or differential operation for different attenuation values. The single-ended/differential mode is controlled by the bit GP_ADC_CTRL_REG[GP_ADC_SE], and the attenuation is handled by the bit GP_ADC_CTRL2_REG[GP_ADC_ATTN].

Table 50: GPADC external input channels and voltage range

GP_ADC_ATTN	GP_ADC_SE	Input Scale	Input Limits
0 (1 ×)	0	-0.9 V to +0.9 V	-1 V to +1 V
	1	0 V to +0.9 V	-0.1 V to 1V
1 (2 ×)	0	-1.8 V to +1.8 V	-1.9 V to +1.9 V
	1	0 V to +1.8 V	-0.1 V to 1.9 V
2 (3 ×)	0	-2.7 V to +2.7 V	-2.8 V to +2.8 V
	1	0 V to +2.7 V	-0.1 V to 2.8 V
3 (4 ×)	0	-3.6 V to +3.6 V	-3.45 V to +3.45 V
	1	0 V to +3.6 V	-0.1 V to 3.45 V

26.2.2 Operating Modes

The GPADC operation flow diagram is shown in Figure 69.

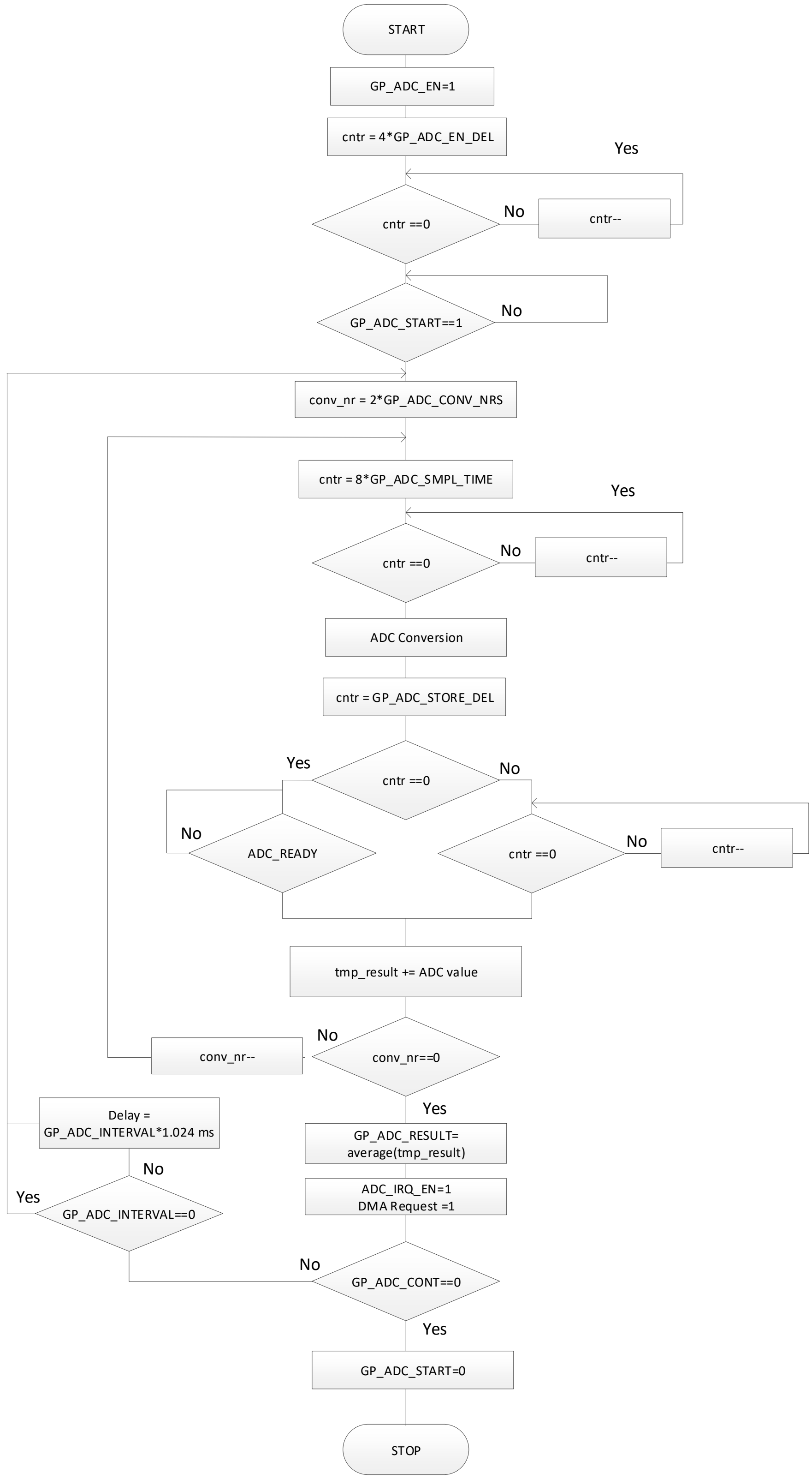

Figure 69. GPADC operation flow diagram

26.2.2.1 Enabling the ADC

Enabling/disabling of the ADC is triggered by configuring bit GP_ADC_CTRL_REG[GP_ADC_EN]. When the bit is set to 1, first the LDO is enabled. Then after the delay value set in GP_ADC_CTRL3_REG[GP_ADC_EN_DEL] (typically 16 µs to account for the LDO settling time), the ADC is enabled, and an AD conversion can be started. See Table 51 for recommended values.

Table 51: ADC_LDO start-up delay

f_{ADC_CLK}	GP_ADC_EN_DEL	$\tau_{ADC\_EN\_DEL}$
16 MHz	0x40	16 µs

Formula:

GP_ADC_EN_DEL = $\tau_{ADC\_EN\_DEL} \times f_{ADC\_CLK} / 4$

This value must be rounded up to the nearest integer.

The GPADC is a dynamic ADC and consumes no static power, except for the **ADC_LDO** which consumes approximately 20 µA. Therefore, GP_ADC_EN must be set to 0 if the ADC is not used.

26.2.2.2 Manual Mode

An AD conversion can be started by setting GP_ADC_START to 1. While a conversion is active, GP_ADC_START remains 1. When a conversion is finished, the hardware sets GP_ADC_START to 0 and GP_ADC_INT to 1 (interrupt), and GP_ADC_RESULT_REG contains the valid ADC value. While a conversion is active, writing 1 to GP_ADC_START does not start a new conversion. Software should always check that bit GP_ADC_START = 0 before starting a new conversion.

26.2.2.3 Continuous Mode

Setting GP_ADC_CTRL_REG[GP_ADC_CONT] to 1 enables the continuous mode, which automatically starts a new AD conversion when the current conversion has been completed. The GP_ADC_START bit is only needed once to trigger the first conversion. As long as the continuous mode is active, GP_ADC_RESULT_REG always contains the latest ADC value.

To correctly terminate the continuous mode, it is required to disable the GP_ADC_CONT bit first and then wait until the GP_ADC_START bit is cleared to 0, so the ADC is in a defined state.

NOTE
Before making any changes to the ADC settings, users must disable the continuous mode by setting bit GP_ADC_CONT to 0 and waiting until bit GP_ADC_START = 0.

At full speed the ADC consumes approximately 50 to 60 µA. If the data rate is less than 100 ksample/s, the current consumption is in the order of 25 µA.

The time interval between two successive AD conversions is programmable with GP_ADC_CTRL3_REG[GP_ADC_INTERVAL] in steps of 1.024 ms. If GP_ADC_INTERVAL = 0, the conversion restarts immediately. If GP_ADC_INTERVAL is not zero, the ADC first synchronizes to the delay clock before starting the conversion. This can take up to 1 ms.

26.2.3 Conversion Modes

26.2.3.1 AD Conversion

Each AD conversion has three phases:

- Sampling
- Conversion
- Storage.

The AD conversion starts with the sampling phase. This phase ends after the time set in GP_ADC_CTRL2_REG[GP_ADC_SMPL_TIME] and triggers the conversion phase. If GP_ADC_CTRL2_REG[GP_ADC_STORE_DEL] = 0, handshaking is used, that is, the ADC result is stored when a conversion is finished. Otherwise, a fixed (programmable) delay is used, and the result is stored regardless of whether the conversion is finished or not.

The total conversion time of an AD conversion depends on various settings. In short, it is as follows.

$$T_{ADC} = \frac{N_{CONV} \cdot \left(N_{CYCL\_SMPL} + N_{CYLC\_STORE}\right)}{f_{ADC\_CLK}} \qquad (5)$$

Where

- N_{CONV} = the number of conversions. This is related to the value programmed in GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS], following $2^{GP\_ADC\_CONV\_NRS}$. When GP_ADC_CTRL_REG[GP_ADC_CHOP] is set, the minimum value for N_{CONV} is always 2.
- N_{CYLC_SMPL} = the number of ADC_CLK cycles used for sampling, which is 8 × GP_ADC_CTRL2_REG[GP_ADC_SMPL_TIME].
- N_{CYCL_STORE} = the number of ADC_CLK cycles until the result is stored. When GP_ADC_CTRL2_REG[GP_ADC_STORE_DEL] = 0, handshaking is used. With handshaking, the number of ADC_CLK cycles is typically three. This value may spread from sample to sample and over temperature, otherwise, the number of ADC_CLK cycles is GP_ADC_CTRL2_REG[GP_ADC_STORE_DEL] + 1.

26.2.3.1.1 Sampling Phase

The sampling time can be programmed through GP_ADC_CTRL2_REG[GP_ADC_SMPL_TIME] and depends on the sampling time constant in combination with the desired sampling accuracy. This sampling time constant, τ_{ADC_SMPL} (Table 52), then depends on the output impedance of the source, the internal resistive dividers, and the internal sampling capacitor. And the number of required time constants is given by the natural logarithm of the desired accuracy, that is, ln(2^N_{BIT}). For N_{BIT} = 10-bit accuracy, 7-time constants are required.

Table 52: ADC sampling time constant (τ_{ADC_SMPL})

ADC Input	τ_{ADC_SMPL}
GPADC0, GPADC1 (GP_ADC_ATTN = 0)	R_{OUT} × 0.5 pF (Differential Input) R_{OUT} × 1 pF (Single-Ended Input)
GPADC0, GPADC1 (GP_ADC_ATTN = 1)	(R_{OUT} + 120 kΩ) × 0.5 pF (Differential Input) (R_{OUT} + 120 kΩ) × 1 pF (Single-Ended Input)

Formula:

GP_ADC_SMPL_TIME = ln(2^N_{BIT}) × τ_{ADC_SMPL} × f_{ADC_CLK}/8

This value must be rounded up to the nearest integer.

26.2.3.1.2 Conversion and Storage Phase

One AD conversion typically takes around 125 ns with a 100 MHz clock. The result can be stored either by handshaking or after a fixed number of cycles (programmable).

- Handshake mode (GP_ADC_STORE_DEL = 0):

In handshake mode the conversion result is available in GP_ADC_RESULT_REG after two sampling ADC_CLK cycles plus two conversion ADC_CLK cycles plus two ADC_CLK cycles for synchronization.

- Fixed delay mode (GP_ADC_STORE_DEL > 0):

In fixed delay mode the conversion result is available in GP_ADC_RESULT_REG after the programmed storage delay, regardless of whether the conversion is ready or not. Note that when the delay is too short (that is, the conversion is not finished in the allocated time), the old (previous) ADC result is stored.

26.2.3.2 Averaging

In order to reduce noise and improve performance, multiple samples can be averaged out (assuming the time average of noise equals zero). This is handled by hardware and can be controlled by setting GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS] to a non-zero value. The actual number of the consecutive samples taken is by $2^{GP\_ADC\_CONV\_NRS}$.

Because the internal noise also acts as a form of dither, the actual accuracy can be improved. Therefore, the ADC result is not truncated to 10-bit but stored as 16-bit left aligned, and truncation is left for the user. The expected Effective Number of Bits (ENOB) is shown in Table 53.

Table 53: ENOB in oversampling mode

GP_ADC_CONV_NRS	ENOB (Left Aligned) in GP_ADC_RESULT_REG
0	> 9
1	> 9
2	> 9
3	> 10
4	> 10
5	> 10
6	> 11
7	> 11

26.2.3.3 Chopper Mode

Inherently, the ADC has a DC offset (E_{OFS}). When GP_ADC_CTRL_REG[GP_ADC_CHOP] is set to 1, the hardware triggers two consecutive AD conversions and flips the sign of the offset in-between. Summing the two samples effectively cancels out the inherent ADC offset. This method also smooths other non-ideal effects and is recommended for DC and the slowly changing signals.

When combined with averaging, every other AD conversion is taken with the opposite sign. Without averaging two AD conversions are always triggered.

Note that a DC offset causes saturation effects at zero scale or full scale. When chopping is used without offset calibration, non-linear behavior is introduced towards zero scale and full scale.

26.2.4 Additional Settings

The hardware also supports pre-ADC attenuation through GP_ADC_CTRL2_REG[GP_ADC_ATTN]:

- Setting 0 disables the attenuator
- Setting 1 scales the input range by a factor of two
- Setting 2 scales the input range by a factor of three
- Setting 3 scales the input range by a factor of four

With bit GP_ADC_CTRL_REG[GP_ADC_MUTE] = 1, the input is connected to 0.5 × ADC reference. So, the ideal ADC result should be 511.5. Any deviation from this is the ADC offset.

With bit GP_ADC_CTRL_REG[GP_ADC_SIGN] = 1, the sign of the offset is inverted. When chopper is used, the hardware alternates GP_ADC_SIGN = 0 and 1. This bit is typically only used for the offset calibration routine described in Section 26.2.6 and has no specific use to the end user.

26.2.5 Non-ideal Effects

Besides Differential Non-Linearity (DNL) and Integral Non-Linearity (INL), each ADC has a gain error (linear) and an offset error (linear). The gain error (E_G) of the GPADC affects the effective input range. The offset error (E_{OFS}) causes the effective input scale to become non-centered. The offset error can be reduced by chopping and/or by offset calibration.

The ADC result also includes some noise. If the input signal itself is noise free (inductive effects included), the average noise level is ±1 LSB. Reducing noise effects can be done by taking more samples and calculating the average value. This can be done by programming GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS] to a non-zero value.

With a "perfect" input signal (for example, if a filter capacitor is placed close to the input pin), most of the noise comes from the low-power voltage regulator (LDO) of the ADC. Because the DA14535 is targeted for ultra-compact applications, there is no pin available to add a capacitor at this voltage regulator output.

The dynamic current of the ADC causes extra noise at the regulator output. This noise can be reduced by setting bits GP_ADC_CTRL2_REG[GP_ADC_I20U]. Bit GP_ADC_I20U enables a constant 20 μA load current at the regulator output so that the current does not drop to zero. This, obviously, increases power consumption by 20 μA.

26.2.6 Offset Calibration

A relative high offset error (E_{OFS}, up to 30 mV, so approximately 30 LSB) is caused by a very small dynamic comparator. This offset error can be cancelled with the chopping function, but it still causes unwanted saturation effects at zero scale or full scale. With GP_ADC_OFFP_REG and GP_ADC_OFFN_REG, the offset error can be compensated in the ADC network itself. To calibrate the ADC, follow the steps in Table 54. In this routine, 0x200 is the target mid-scale of the ADC.

Table 54: GPADC calibration procedure for single-ended and differential modes

Step	Single-Ended Mode (GP_ADC_SE = 1)	Differential Mode (GP_ADC_SE = 0)
1	Set GP_ADC_OFFP = GP_ADC_OFFN = 0x200; GP_ADC_MUTE = 0x1; GP_ADC_SIGN = 0x0.	Set GP_ADC_OFFP = GP_ADC_OFFN = 0x200; GP_ADC_MUTE = 0x1; GP_ADC_SIGN = 0x0.
2	Start conversion.	Start conversion.
3	adc_off_p = GP_ADC_RESULT - 0x200	adc_off_p = GP_ADC_RESULT - 0x200
4	Set GP_ADC_SIGN = 0x1.	Set GP_ADC_SIGN = 0x1.
5	Start conversion.	Start conversion.
6	adc_off_n = GP_ADC_RESULT - 0x200	adc_off_n = GP_ADC_RESULT - 0x200
7	GP_ADC_OFFP = 0x200 - 2 × adc_off_p GP_ADC_OFFN = 0x200 - 2 × adc_off_n	GP_ADC_OFFP = 0x200 - adc_off_p GP_ADC_OFFN = 0x200 - adc_off_n

To increase the accuracy, it is recommended to set the GP_ADC_CTRL2_REG[GP_ADC_SMPL_TIME] = 2 or 3 and GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS] = 3 or 4 prior to this routine.

It is recommended to implement the above calibration routine during the initialization phase of DA14535. To verify the calibration results, check whether the GP_ADC_RESULT value is close to 0x200 while bit GP_ADC_CTRL_REG[GP_ADC_MUTE] = 1.

26.2.7 Zero-scale Adjustment

The GP_ADC_OFFP and GP_ADC_OFFN registers can also be used to set the zero-scale or full-scale input level at a certain target value. For instance, they can be used to calibrate GP_ADC_RESULT to 0x000 at an input voltage of exactly 0.0 V, or to calibrate the zero scale of a sensor.

26.2.8 Common Mode Adjustment

The common mode level of the differential signal must be 0.45 V = Full Scale/2 (or 1.35 V with GP_ADC_ATTN = 2, that is, 3× attenuation). If the common mode input level of 0.45 V cannot be achieved, the common mode level of the GPADC can be adjusted through GP_ADC_OFFP_REG and GP_ADC_OFFN_REG according to Table 55. The GPADC can tolerate a common mode margin of up to 50 mV.

Table 55: Common mode adjustment

CM Voltage (V_{ccm})	GP_ADC_OFFP = GP_ADC_OFFN
0.225 V	0x300
0.450 V	0x200
0.675 V	0x100

Any other common mode levels between 0.0 V and 0.9 V can be calculated from Table 55. Offset calibration can be combined with common mode adjustment by replacing the 0x200 value in the offset calibration routine with the value required to get the appropriate common mode level.

26.2.9 Input Impedance, Inductance, and Input Settling

The GPADC has no input buffer stage. During the sampling phase, a capacitor of 0.5 pF in differential mode or 1 pF in single-ended mode is switched to the input line(s). The pre-charge of this capacitor is at midscale level, so the input impedance is infinite.

During the sampling phase, a certain settling time is required. A 10-bit accuracy requires at least seven-time constants τ_{ADC_SMPL}, determined by the output impedance of the input signal source, the internal resistive dividers, and the 0.5 pF or 1 pF sampling capacitor. See Table 52.

The inductance from the signal source to the ADC input pin must be very small. Otherwise, filter capacitors are required from the input pins to ground (single-ended mode) or from pin to pin (differential mode).

26.3 Programming

To program and use the GPADC:

1. Enable the GPADC by setting the GP_ADC_CTRL_REG[GP_ADC_EN] bit.
2. Set up the GPIO input (P0_x_MODE_REG[PID] = 15).
3. Select the input channel (GP_ADC_SEL_REG).
4. Select the sampling mode (differential or single ended) by writing the GP_ADC_CTRL_REG[GP_ADC_SE] bit.
5. Select between the manual mode and the continuous mode of sampling (GP_ADC_CTRL_REG[GP_ADC_CONT].
6. Set up extra options (see GP_ADC_CTRLx_REG description)
7. Start the conversion by setting GP_ADC_CTRL_REG[GP_ADC_START] bit.
8. Wait for GP_ADC_CTRL_REG[GP_ADC_START] to become 0 or interrupt being triggered (when used).
9. Clear the ADC interrupt by writing any value to GP_ADC_CLEAR_INT_REG.
10. Get the ADC result from the GP_ADC_RESULT_REG.

27. Real Time Clock

27.1 Introduction

The DA14535 is equipped with a Real Time Clock (RTC) which provides the complete clock and calendar information with automatic time units adjustment and easy configuration.

Features

- Complete time of day clock: 12/24 hour, hours, minutes, seconds, and hundredths of a second.
- Calendar function: day of week, date of month, month, year, century, leap year compensation, and year 2000 compliant.
- Alarm function: month, date, hour, minute, second, and hundredths of a second.
- Event interrupt on any calendar or time unit.
- Available during sleep if the power domain PD_TIM is kept alive.
- Granularity of 10 ms (RTC_CLK).
- Provides 22 LSB to Timer 1 upon a capture trigger.

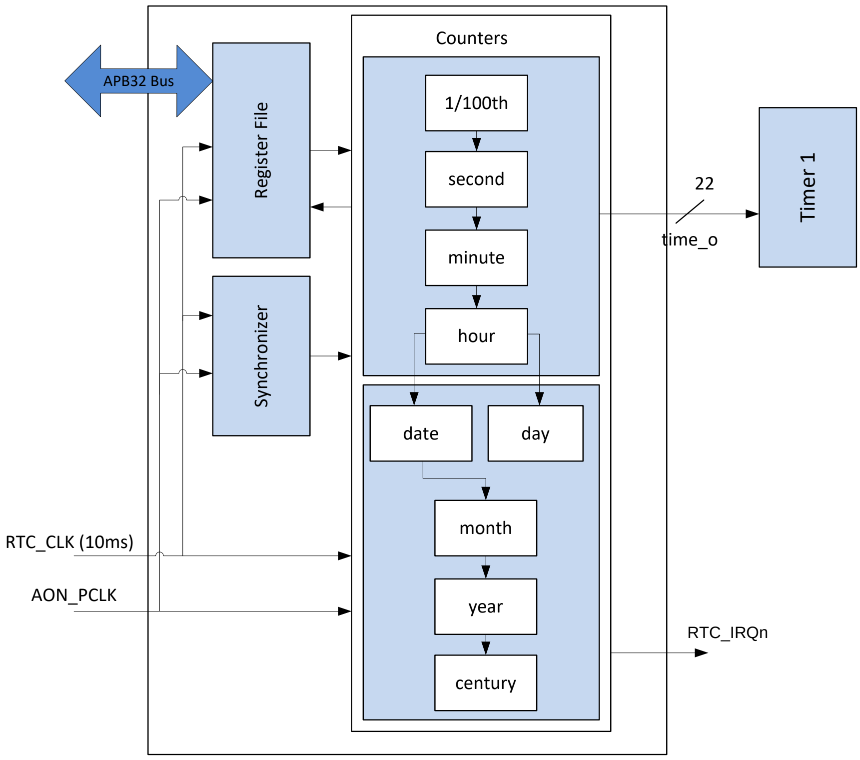

Figure 70. Real Time Clock block diagram

27.2 Architecture

The architecture of the RTC is shown in Figure 70.

The RTC supports a year range from 1900 to 2999 as well as full month, date, minute, second, and hundredth of second ranges. It also supports hour ranges of 0 to 23 (24-hour format) or 1 to 12 with a.m./p.m. flag (12-hour format).

Alarms can be generated in two ways, as a one-time alarm or as a recurring alarm. In addition to alarms, the RTC can detect when a particular event occurs. Each field of the calendar and time counter can generate an event when it rolls over. For example, an event can be generated every new month, new week, new day, new half day (12-hour mode), new minute, or new second. Both alarms and events can generate an interrupt. All the interrupts can be set, enabled, disabled, or masked at any time.

The LSB (22) of the port showing a full of 32-bit information on the current time is latched by Timer 1 (TIMER1_CAPCNT1/2_VALUE_REG) if instructed by Timer 1 configuration. This allows for storing an RTC based snapshot upon an event on a GPIO.

27.3 Programming

To configure the RTC:

1. Configure the 100 Hz RTC granularity if needed:
   a. Based on the selected LP clock (for example, 32768 kHz), set the CLK_RTCDIV_REG[RTC_DIV_INT] = 327 (= 0x147).
   These values should be equal to the integer divisor part of the formula
   F_{LP_CLK}/100 = **327**.680.
   b. Based on the selected LP clock (for example, 32768 kHz), set the CLK_RTCDIV_REG[RTC_DIV_FRAC] = 680 (= 0x2A8).
   These values should be equal to the fractional divisor part of the formula
   F_{LP_CLK}/100 = 327.**680**.
   c. To achieve a better accuracy of the divisor, configure the denominator for the fractional division accordingly (CLK_RTCDIV_REG[RTC_DIV_DENOM]).
   d. Enable the 100 Hz RTC granularity by setting the CLK_RTCDIV_REG [RTC_DIV_ENABLE] bit.
2. Enable the time functionality by clearing the RTC_CONTROL_REG[RTC_TIME_DISABLE].
3. Enable the calendar functionality by clearing the RTC_CONTROL_REG[RTC_CAL_DISABLE].
4. Choose between 12-hour or 24-hour mode (RTC_HOUR_MODE_REG[RTC_HMS]).
5. Configure the time (RTC_TIME_REG).
6. Configure the date (RTC_CALENDAR_REG).
7. Set up a time alarm if needed (RTC_ALARM_ENABLE_REG).
8. Set up a calendar alarm if needed (RTC_CALENDAR_ALARM_REG).
9. Enable the configured alarms (RTC_ALARM_ENABLE_REG[RTC_ALARM_xxxx_EN]).
10. Configure the interrupt generation when an alarm happens (RTC_INTERRUPT_ENABLE_REG). Disable the interrupt generation with RTC_INTERRUPT_DISABLE_REG.
11. Configure the event flag generation when an alarm happens (RTC_EVENT_FLAGS_REG).
12. Define whether a software reset resets the RTC (RTC_KEEP_RTC_REG[RTC_KEEP]).

28. Power

As discussed in Section 4.2, the integrated power management unit (PMU) comprises the DCDC converter and various LDOs, the V_{DD} Clamp, and the POR circuitry. The details of these blocks are discussed in the following sections.

28.1 DCDC Converter

The DA14535 can be configured in three configurations: buck, boost, and DCDC bypass. The integrated part of the DCDC is the same for all three configurations, that is, the black building blocks in Figure 71 and Figure 72. The buck configuration and the boost configuration are configured on the PCB, distinguished with the red external components in Figure 71 and Figure 72, respectively. In the bypass configuration the V_{BAT_HIGH} and V_{BAT_LOW} rails are connected, so the DCDC is bypassed.

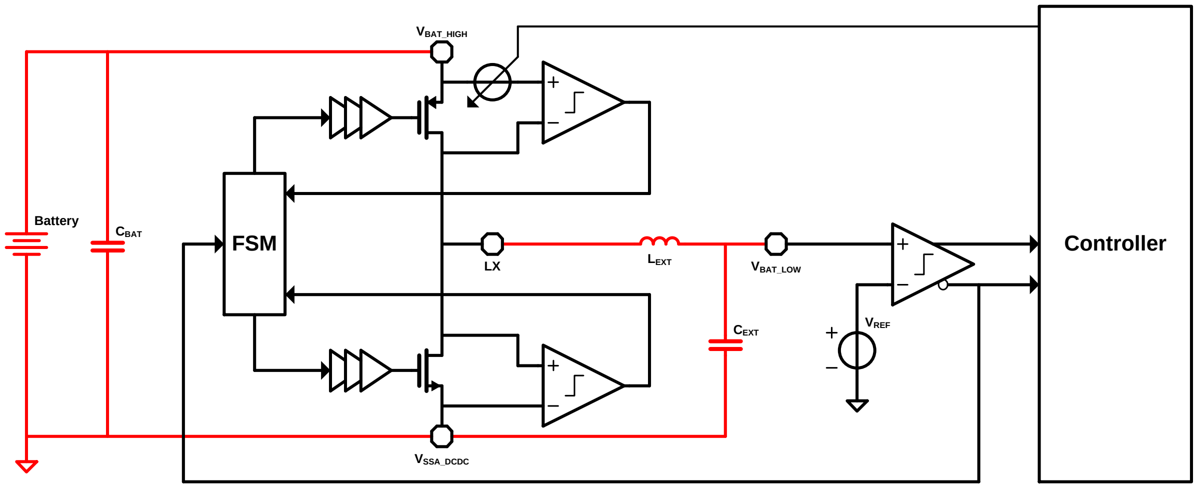

Figure 71. DCDC block diagram – buck configuration

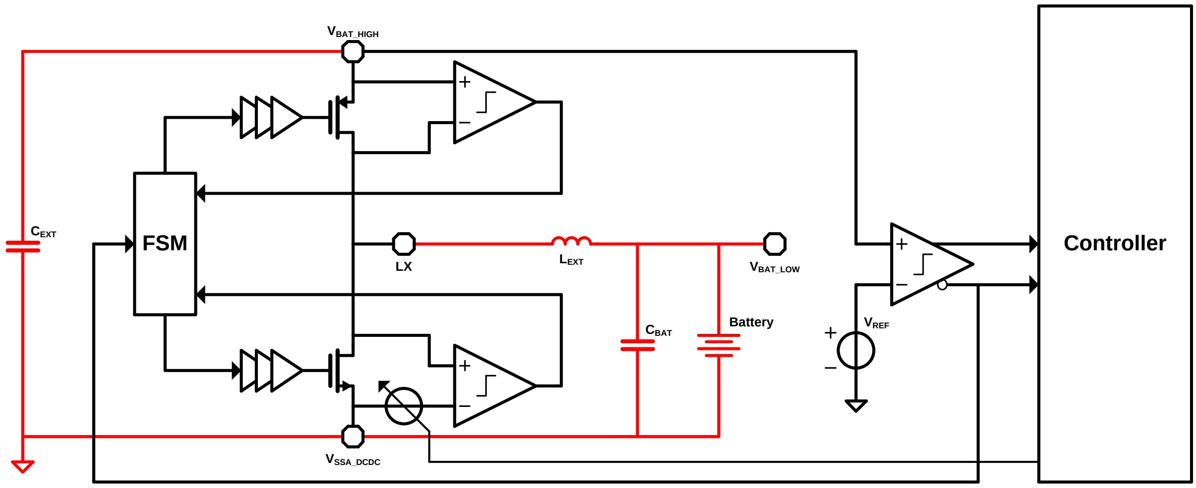

Figure 72. DCDC block diagram – boost configuration

- In Buck configuration the battery is connected to V_{BAT_HIGH}, and DCDC supplies power to V_{BAT_LOW} rail.
- In Boost configuration the battery is connected to V_{BAT_LOW}, and DCDC supplies power to V_{BAT_HIGH} rail.
- In DCDC bypass configuration V_{BAT_HIGH} is connected to V_{BAT_LOW} and the battery is connected to both rails.

The DCDC level can be programmed by POWER_LEVEL_REG[DCDC_LEVEL] and a fine trimming is available by POWER_LEVEL_REG[DCDC_TRIM].

In Buck configuration, to enable the DCDC converter in sleep mode, the POWER_CTRL_REG[DCDC_ENABLE_SLEEP] should be used. When this bit is set and the system goes to sleep, the DCDC converter is enabled instead of the retention LDOs. When in sleep mode, the V_{BAT_LOW} rail is monitored by the same comparator as in active mode, but the RC32K clock is used instead of the RC32M clock. When undervoltage is detected, the RC32M is enabled together with the DCDC-FSM and the V_{BAT_LOW} rail is charged. If the voltage on the rail is OK, the RC32M and DCDC-FSM are turned off again. This results in duty cycled operation of the DCDC converter, where the duty cycle depends on the load current.

For Buck configuration, a typical DCDC efficiency at 25 °C as a function of the load current for different battery voltages (V_{BAT} = V_{BAT_HIGH}) is shown in Figure 73.

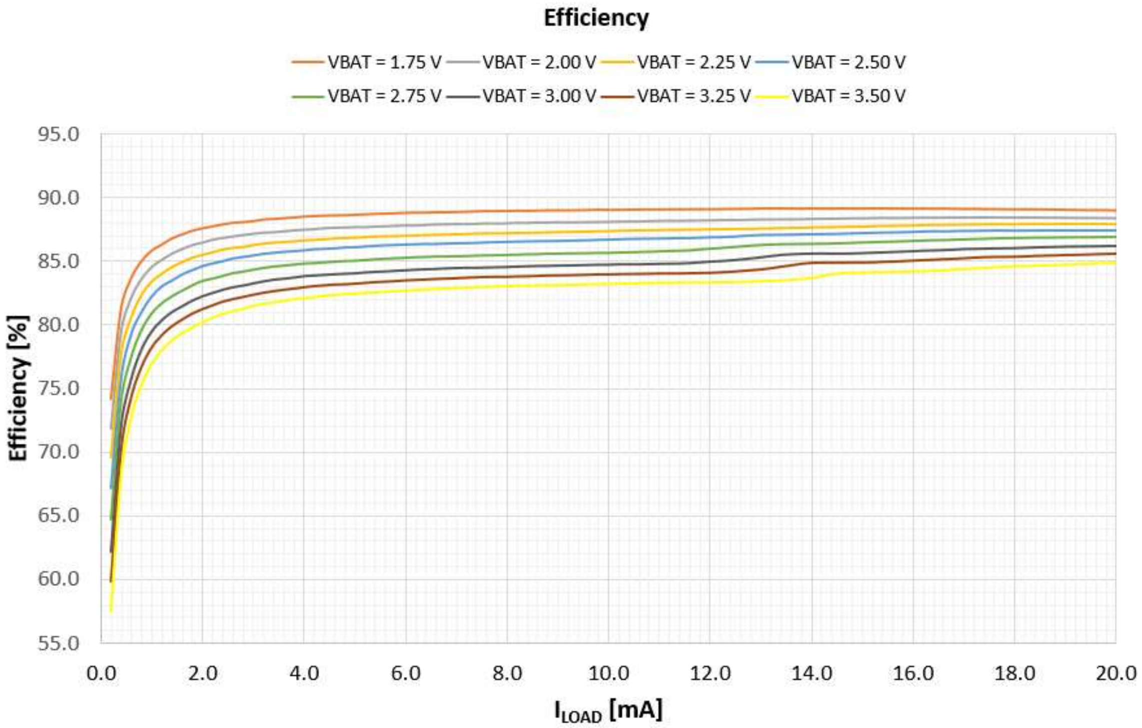

Figure 73. DCDC efficiency in buck configuration

For Boost configuration, a typical DCDC efficiency at 25 °C as a function of the load current for different battery voltages (V_{BAT} = V_{BAT_LOW}) is shown in Figure 74.

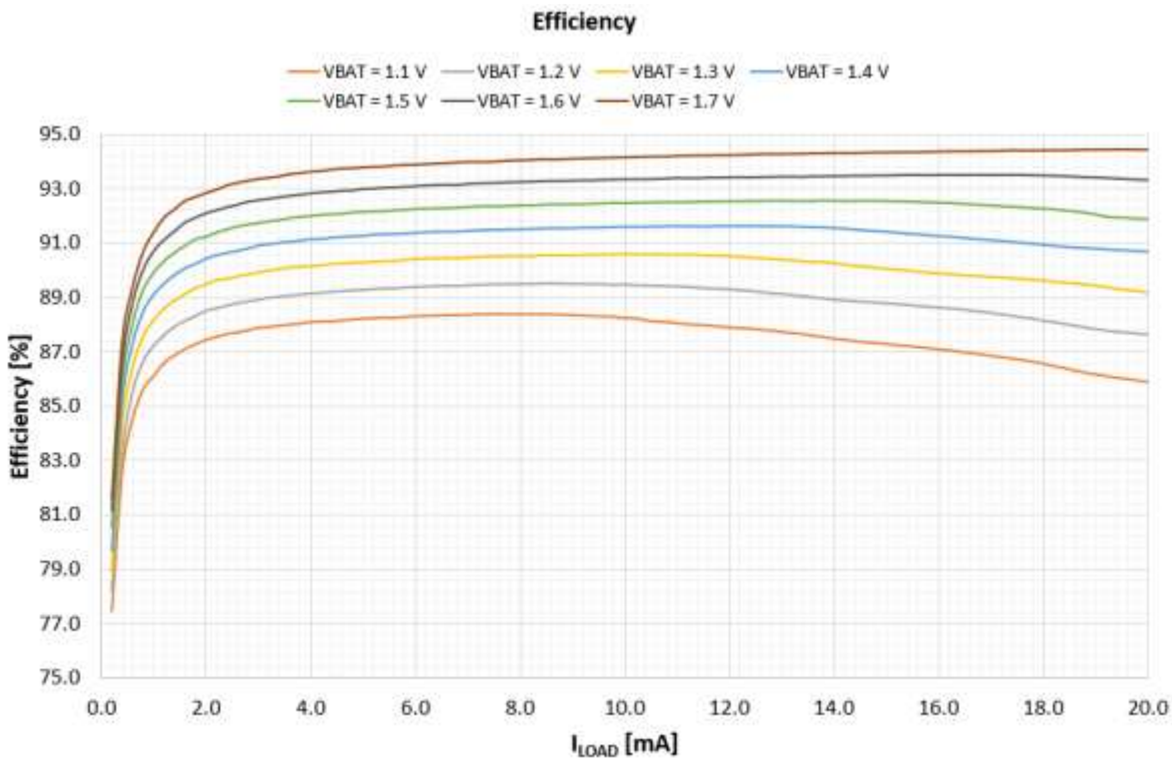

Figure 74. DCDC efficiency in boost configuration

28.2 LDOs

Several LDOs are used in DA14535 to provide a stable power supply to the rails and the building blocks.

- V_{DD}_Clamp generates a trimmable ~0.75 V V_{DD} supply voltage for the AON (always on) DCORE power domain from V_{BAT_HIGH} or V_{BAT_LOW} when the system is in hibernation mode.
- LDO_LOW provides power to the V_{BAT_LOW} rail in the buck configuration with a typical output voltage of 1.2 V. This LDO is used during start-up and can also be used after start-up. Alternatively, it can be disabled and the V_{BAT_LOW} rail can be supplied by the DCDC converter. The LDO has a low power setting which is used to maintain the V_{BAT_LOW} rail during sleep mode if the DCDC is disabled. See Section 4.2.3 for more details.
- LDO_CORE supplies the internal V_{DD} from V_{BAT_LOW}. In the active mode it generates 0.9 V and in the sleep mode 0.75 V.
- LDOs for the RF and the analog building blocks generate 0.9 V when the particular blocks are active. When the blocks are switched off, the LDOs are disabled.

28.3 POR Circuit

The POR_LOW circuit issues a POR when the VBAT_LOW voltage is below the threshold voltage V_{IL} for more than 50 µs. The POR is cleared when the battery voltage is above V_{IH} for at least 25 µs. The threshold levels of the POR circuit are summarized in Section 3.9.

The POR_HIGH circuit issues a POR when the VBAT_HIGH voltage is below the V_{IL} for more than 50 µs. The POR is cleared when the battery voltage is above V_{IH} for at least 25 µs. The threshold levels of the POR circuit are summarized in Section 3.9.

29. Bluetooth® LE Core

The Bluetooth® Low Energy core used in DA14535 is a qualified Bluetooth® 5.3 baseband controller compatible with the Bluetooth LE specification and it is in charge of packet encoding/decoding and frame scheduling.

The block diagram of Bluetooth LE core is shown in Figure 75.

Features

- Compliant with Bluetooth® Core Specification, v5.3, Bluetooth® SIG.
  - Dual topology
  - Low duty cycle advertising
  - L2CAP connection-oriented channels
- All device classes support (Broadcaster, Central, Observer, and Peripheral).
- All packet types (Advertising, Data, and Control).
- Dedicated Encryption (AES/CCM).
- Bit stream processing (CRC and Whitening).
- FDMA/TDMA/events formatting and synchronization.
- Frequency hopping calculation.
- Operating clock 16 MHz or 8 MHz.
- Low power modes supporting 32.0 kHz, 32.768 kHz, or 15 kHz.
- Supports powerdown of the baseband during the protocol's idle periods.

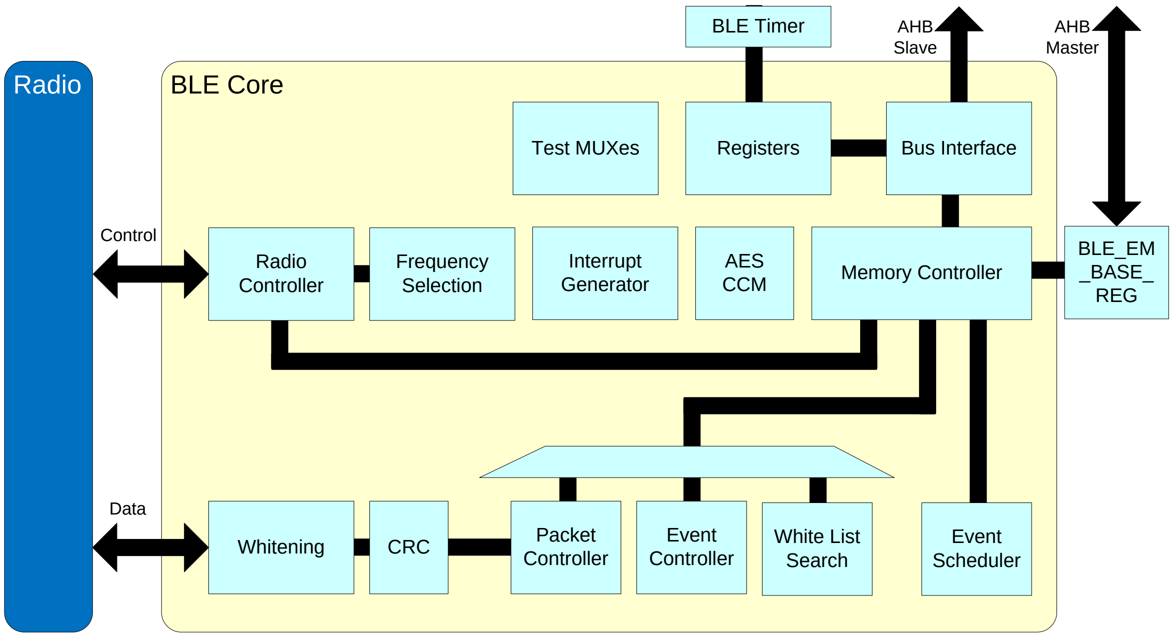

Figure 75. Bluetooth LE core block diagram

29.1 Architecture

29.1.1 Exchange Memory

The Bluetooth LE Core requires access to a memory space named "Exchange Memory" to store control structures and frame buffers. The access to Exchange Memory is performed through the AHB Master interface. The base address of the Exchange Memory is programmable by means of the BLE_EM_BASE register.

29.2 Programming

29.2.1 Wake-up IRQ

When the Bluetooth LE core switches to "Bluetooth LE Deep Sleep mode", the only way to correctly exit from this state is by generating initially the BLE_WAKEUP_LP_IRQ and consecutively the BLE_SLP_IRQ. This sequence must be followed regardless of the cause of the termination of the "BLE Deep Sleep mode", that is, regardless of whether the Bluetooth LE Timer has expired or Bluetooth LE Timer has been stopped due to the assertion of BLE_WAKEUP_REQ.

The assertion and de-assertion of BLE_WAKEUP_LP_IRQ is fully controlled through the BLE_ENBPRESET_REG bit fields. A detailed description is as follows:

- **TWIRQ_SET:** it defines the number of "ble_lp_clk" cycles before the expiration of the Bluetooth LE Timer when the BLE_WAKEUP_LP_IRQ must be asserted. It is recommended to select a TWIRQ_SET value larger than the amount of time that is required to finish trimming the XTAL 32 MHz (refer to XTAL32M_TRIM_READY) plus the execution time of the IRQ Handler. If the programmed value of TWIRQ_SET is less than the minimum recommended value, the system wakes up but the actual Bluetooth LE sleep duration (see BLE_DEEPSLSTAT_REG) is larger than the programmed sleep duration (see BLE_DEEPSLWKUP_REG).
- **TWIRQ_RESET:** it defines the number of "ble_lp_clk" cycles before the expiration of the sleep period when BLE_WAKEUP_LP_IRQ is de-asserted. It is recommended to always set its value to 1.
- **TWEXT:** it determines the high period of BLE_WAKEUP_LP_IRQ if an external wake-up event (refer to GP_CONTROL_REG[BLE_WAKEUP_REQ]) occurs. Its minimum value is "TWIRQ_RESET + X", where X is the number of "ble_lp_clk" clock cycles that BLE_WAKEUP_LP_IRQ is held high. The recommended value is "TWIRQ_RESET + 1". Note that as soon as GP_CONTROL_REG[BLE_WAKEUP_REQ] is set to 1, BLE_WAKEUP_LP_IRQ is asserted.
- **Minimum Bluetooth LE Sleep Duration:** The minimum value of BLE_DEEPSLWKUP_REG[DEEPSLTIME] bit, measured in "ble_lp_clk" cycles, is the higher value between (a) "TWIRQ_SET + 1" and (b) the software execution time from setting BLE_DEEPSLCNTL_REG[DEEP_SLEEP_ON] up to preparing CPU to accept the BLE_WAKEUP_LP_IRQ (for example, to call the Cortex instruction WFI). If the programmed DEEPSLTIME is less than the minimum value of BLE_DEEPSLWKUP_REG[DEEPSLTIME], the BLE_WAKEUP_LP_IRQ Handler may execute sooner than the call of the Cortex WFI instruction in the example and cause software instability.

29.2.2 Switch from Bluetooth LE Active Mode to Bluetooth LE Deep Sleep Mode

Software can set the Bluetooth LE core into the "Bluetooth LE Deep Sleep mode" by first programming the timing of BLE_WAKEUP_LP_IRQ generation, then programming the desired sleep duration at BLE_DEEPSLWKUP_REG, and finally set the register bit BLE_DEEPSLCNTL_REG[DEEP_SLEEP_ON].

During the "Bluetooth LE Deep Sleep mode", the Bluetooth LE Core switches to the "ble_lp_clk" (15 kHz, 32.0 kHz, or 32.768 kHz) to maintain its internal 625 μs timing reference. Software must poll the state of BLE_CNTL2_REG[RADIO_PWRDN_ALLOW] to detect the completion of this mode transition. When the "ble_lp_clk" is used for base time reference, software must disable the Bluetooth LE clocks ("ble_master1_clk", "ble_master2_clk", and "ble_crypt_clk") by setting the CLK_RADIO_REG[BLE_ENABLE] register bit to 0.

Finally, software can optionally power down the Radio Subsystem by using the PMU_CTRL_REG[RADIO_SLEEP] and the Peripheral and System power domains as well.

Figure 76 shows the waveforms when the Bluetooth LE Deep Sleep mode is entered. In this case, as soon as the software detects that RADIO_PWRDOWN_ALLOW is 1, it sets the PMU_CTRL_REG[RADIO_SLEEP] to power down the Radio Subsystem. In Figure 76, Figure 77, Figure 78, Figure 79, and Figure 80, the corresponding Bluetooth LE Core signals are marked with red while Radio Subsystem is in power-down state and they remain red-marked during the period when RADIO_SLEEP is set.

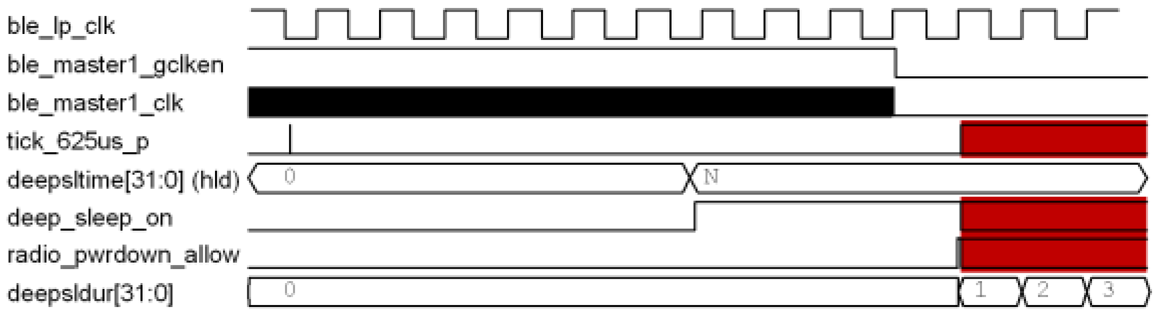

Figure 76. Entering Bluetooth® LE Deep Sleep mode

29.2.3 Switch from Bluetooth LE Deep Sleep Mode to Bluetooth LE Active Mode

There are two possibilities for the Bluetooth LE Core to terminate the Bluetooth LE Deep Sleep mode:

- Termination at the end of a predetermined time.
- Termination on software wake-up request due to an external event.

29.2.3.1 Switching at an Anchor Point

Figure 79 shows a typical Bluetooth LE deep sleep phase that is terminated at a predetermined time. After a configurable time before the scheduled wake-up time (configured through the BLE_ENBPRESET_REG register bit fields), the Bluetooth LE Timer asserts the BLE_WAKEUP_LP_IRQ to wake up the CPU (powering up the System Power Domain). The BLE_WAKEUP_LP_IRQ Interrupt Handler prepares the code environment and the XTAL32M oscillator stabilization (see SYS_STAT_REG[XTAL32_SETTLED]) and decides when the Bluetooth LE Core is ready to exit the Bluetooth LE Deep Sleep mode.

When the software decides that the Bluetooth LE Core can wake up, it must enable the Bluetooth LE clocks (through CLK_RADIO_REG[BLE_ENABLE]) and power up the Radio Power Domain (refer to PMU_CTRL_REG[RADIO_SLEEP] and SYS_STAT_REG[RAD_IS_UP]).

After the sleep period is expired (as specified in BLE_DEEPSLWKUP_REG[DEEPSLTIME]), the Bluetooth LE Timer does not exit the Bluetooth LE Deep Sleep mode until it detects that the Bluetooth LE Core is powered up. That means, if the software requires more time to power up the Bluetooth LE Core, the final sleep duration (provided by BLE_DEEPSLSTAT_REG) is longer than the preprogrammed value.

When the Bluetooth LE Timer expires, Bluetooth LE clocks are enabled, and the Bluetooth LE Core (Radio Subsystem) is powered up, the Bluetooth LE Core exists the "Bluetooth LE Core Deep Sleep mode" and asserts the BLE_SLP_IRQ.

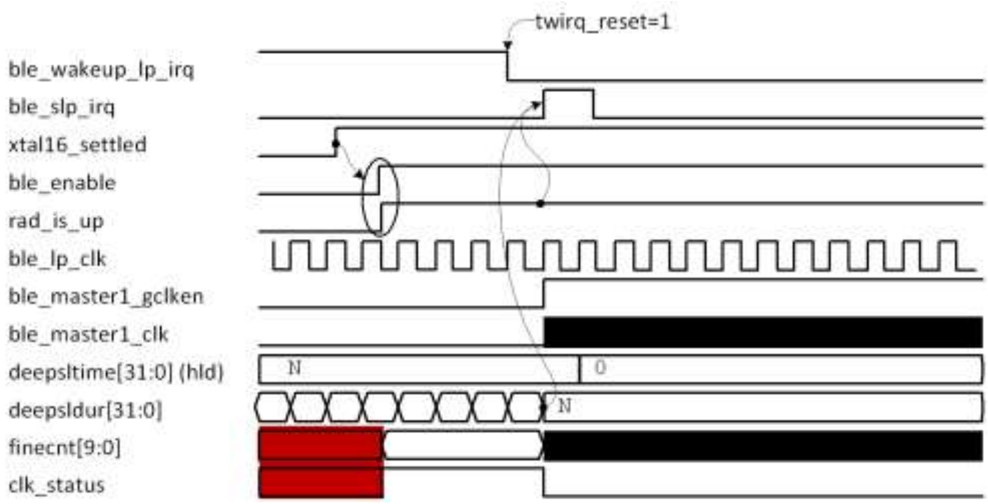

Figure 77. Exit Bluetooth LE Deep Sleep mode at predetermined time (zoom in)

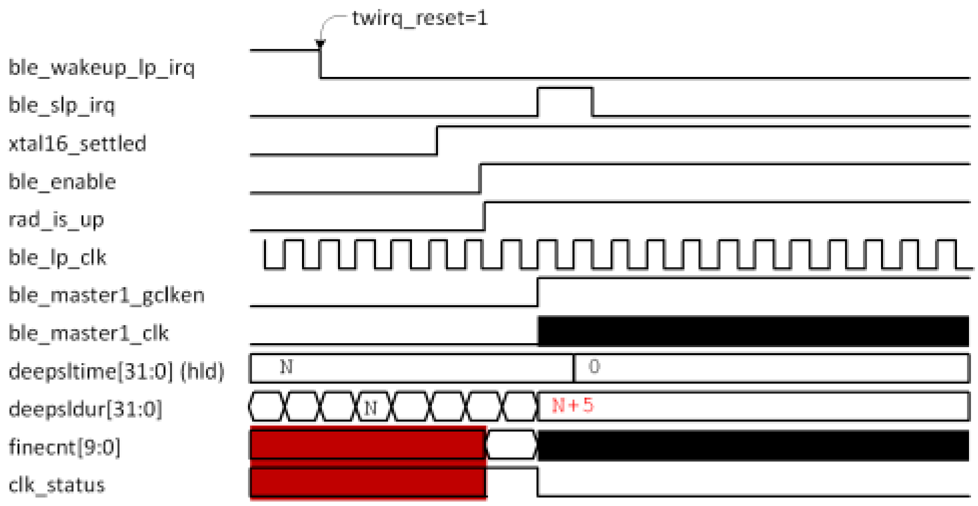

Figure 78. Exit Bluetooth LE Deep Sleep mode after predetermined time (zoom in)

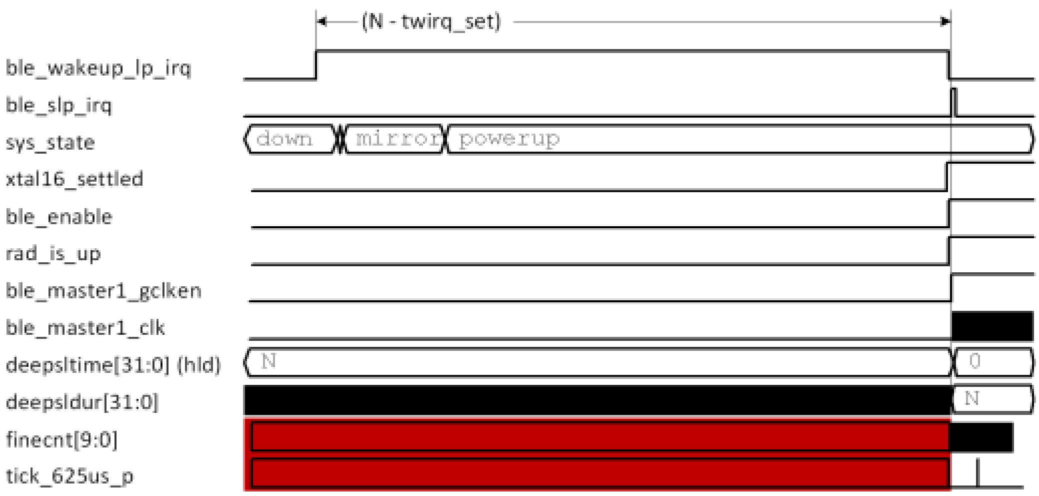

Figure 79. Exit Bluetooth LE Deep Sleep mode at predetermined time (zoom out)

29.2.3.2 Switching due to an External Event

Figure 80 shows a wake-up from a Bluetooth LE deep sleep period forced by the assertion of register bit GP_CONTROL_REG[BLE_WAKEUP_REQ].

Assume that the system is in Extended Sleep state with all power domains switched off and both the wake-up timer and wake-up controller programmed appropriately. Then assume that an event is detected at one of the GPIOs, causing the System Power Domain to wake up due to WKUP_QUADDEC_IRQ. In that case, the software decides to wake up the Bluetooth LE core, then it sets the GP_CONTROL_REG[BLE_WAKEUP_REQ] to 1 to force the wake-up sequence.

In Figure 80, BLE_WAKEUP_REQ is raised by the software as soon as possible, causing BLE_WAKEUP_LP_IRQ Handler to be executed as soon as possible. It is also possible to raise BLE_WAKEUP_REQ after the detection of XTAL32_TRIM_READY, causing both BLE_WAKEUP_LP_IRQ and BLE_SLP_IRQ Handlers to be executed sequentially. The decision depends on the software structure and the application.

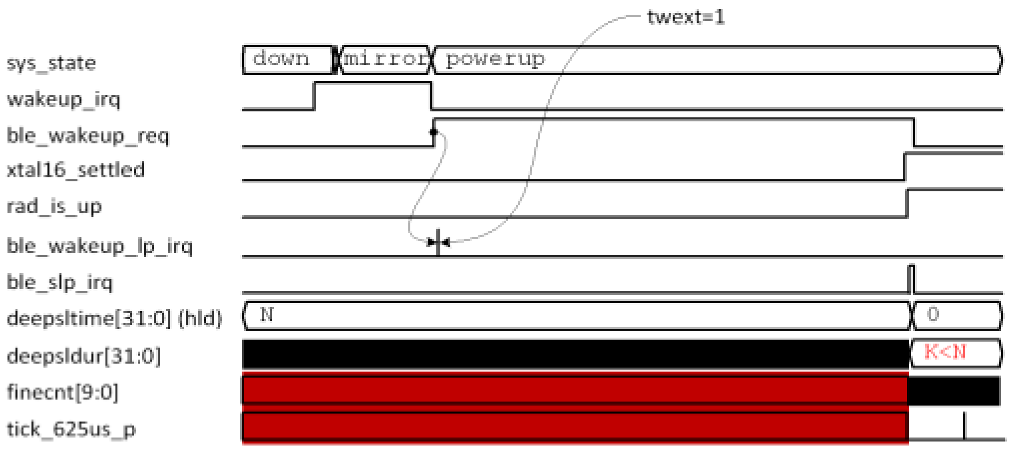

Figure 80. Exit Bluetooth LE Deep Sleep mode due to external event

As soon as the bit field BLE_WAKEUP_REQ is set to 1, BLE_WAKEUP_LP_IRQ is asserted. In that case, the high period of BLE_WAKEUP_LP_IRQ is controlled through TWEXT. The recommended value of TWEXT is "TWIRQ_RESET + 1", meaning that BLE_WAKEUP_LP_IRQ remains high for one "ble_lp_clk" period.

If the BLE_WAKEUP_REQ is high, entering the sleep mode is prohibited. Note that BLE_WAKEUP_REQ event can be disabled by setting BLE_DEEPSLCNTL_REG[EXTWKUPDSB].

30. Radio

30.1 Introduction

The Radio Transceiver implements the RF part of the Bluetooth® LE protocol. Together with the Bluetooth® 5.3 PHY layer, it provides up to 98.5 dB RF link budget for reliable wireless communication. All RF blocks are supplied by on-chip low-drop out-regulators (LDOs). The bias scheme is programmable per block and optimized for minimum power consumption. The radio block diagram is given in Figure 81. It comprises the Receiver, Transmitter, Synthesizer, RX/TX combiner block, and Biasing LDOs.

Features

- Single ended RFIO interface, 50 Ω matched
- Alignment free operation
- -94.5 dBm receiver sensitivity
- Configurable transmit output power from -19 dBm up to +4 dBm
- Ultra-low power consumption
- Fast frequency tuning minimizes overhead.

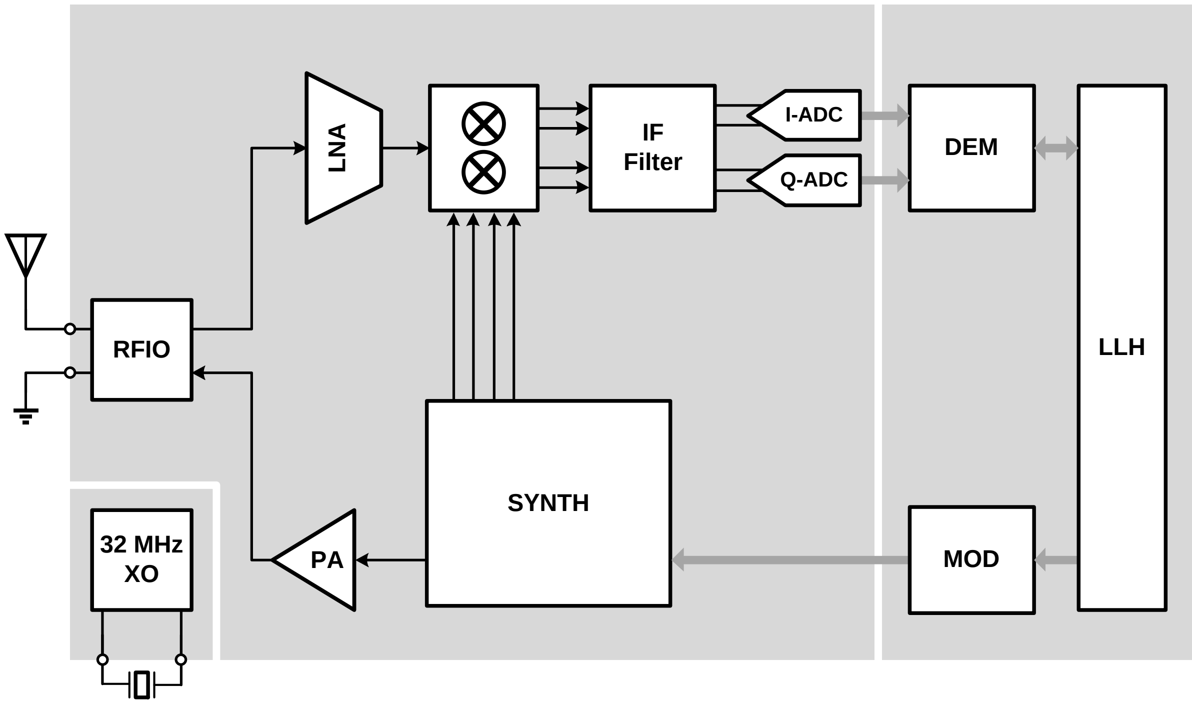

Figure 81. Bluetooth radio block diagram

30.2 Architecture

30.2.1 Receiver

The RX frontend consists of a selective matching network, a low noise amplifier (LNA), and an image rejection down conversion mixer. The intermediate frequency (IF) part of the receiver comprises a filter with a programmable gain. The LNA and IF Filter gains are controlled by the Automatic Gain Control (AGC). This provides the necessary signal conditioning prior to digitalization. The digital demodulator block (DEM) provides a synchronous bit stream.

30.2.2 Synthesizer

The RF Synthesizer generates the quadrature LO signal for the mixer, but also generates the modulated TX output signal. The Digitally Controlled Oscillator (DCO) runs at twice the required frequency and a dedicated divide-by-2 circuit generates the 2.4 GHz signals in the required phase relations. The reference frequency is the 32 MHz crystal clock. The modulation of the TX frequency is performed by two-point modulation.

30.2.3 Transmitter

The RF power amplifier (RFPA) is an extremely efficient Class-D structure, typically providing the power ranging from -19 dBm to +4 dBm to the antenna. It is fed by the DCO's divide-by-2 circuit and delivers its TX power to the antenna pin through the combined RX/TX matching circuit.

30.2.4 RFIO

The RX/TX combiner block is a unique feature of DA14535. It makes sure that the received power is applied to the LNA with minimum losses towards the RFPA. In TX mode, the LNA poses a minimal load for the RFPA and its input pins are protected from the RFPA. In both modes, the single-ended RFIO port is matched to a resistor of 50 Ω to provide the simplest possible interfacing to the antenna on the printed circuit board.

30.2.5 Biasing

All RF blocks are supplied by on-chip LDOs. The bias scheme is programmable and optimized for minimum power consumption.

30.2.6 RF Monitoring

The Radio is equipped with a monitoring block whose responsibility is to acquire the data provided by the RF Unit and other analog resources, to combine them in words of 32 bits (when necessary), and to store them in system's memory. Data can be the output of the Demodulator (I and Q) or be provided by the GPADC. With the monitoring block, production tests of the corresponding block can be achieved.

31. Registers

This section contains a detailed view of the DA14535 registers. It is organized as follows: an overview table is presented initially, which depicts all register names, addresses, and descriptions. A detailed bit level description of each register follows.

The register file of the Arm Cortex-M0+ can be found in the following documents available on the website:

Devices Generic User Guide:
https://developer.arm.com/documentation/dui0497/a/CHDBIBGJ

Technical Reference Manual:
https://developer.arm.com/docs/ddi0484/c/preface

These documents contain the register descriptions for the Nested Vectored Interrupt Controller (NVIC), the System Control Block (SCB), and the System Timer (SysTick).

31.1 Analog Miscellaneous Registers

Table 56: Register map ANAMISC

Address	Register	Description
0x50001600	CLK_REF_SEL_REG	Select clock for oscillator calibration
0x50001602	CLK_REF_CNT_REG	Count value for oscillator calibration
0x50001604	CLK_REF_VAL_L_REG	XTAL32M reference cycles, lower 16 bits
0x50001606	CLK_REF_VAL_H_REG	XTAL32M reference cycles, higher 16 bits

Table 57: CLK_REF_SEL_REG (0x50001600)

Bit	Mode	Symbol/Description	Reset
3	R/W	EXT_CNT_EN_SEL 0: Enable XTAL_CNT counter by the REF_CLK selected by REF_CLK_SEL. 1: Enable XTAL_CNT counter from an external input.	0x0
2	R/W	REF_CAL_START Writing 1 starts a calibration of the clock selected by CLK_REF_SEL_REG[REF_CLK_SEL]. This bit is cleared when calibration is finished, and CLK_REF_VAL is ready.	0x0
1:0	R/W	REF_CLK_SEL Select clock input for calibration: 0x0: RC32K 0x1: RC32M 0x2: XTAL32K 0x3: RCX	0x0

Table 58: CLK_REF_CNT_REG (0x50001602)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	REF_CNT_VAL Indicates the calibration time, with a decrement counter to 1.	0x0

Table 59: CLK_REF_VAL_L_REG (0x50001604)

Bit	Mode	Symbol/Description	Reset
15:0	R	XTAL_CNT_VAL	0x0

Bit	Mode	Symbol/Description	Reset
		Returns the number of the DIVN clock cycles counted during the calibration time, defined with REF_CNT_VAL	

Table 60: CLK_REF_VAL_H_REG (0x50001606)

Bit	Mode	Symbol/Description	Reset
15:0	R	XTAL_CNT_VAL Returns the number of the DIVN clock cycles counted during the calibration time, defined with REF_CNT_VAL	0x0

31.2 Bluetooth® LE Core Registers

Table 61: Register map BLE

Address	Register	Description
0x40000000	BLE_RWBLECNTL_REG	Bluetooth LE Control register
0x40000004	BLE_VERSION_REG	Version register
0x40000008	BLE_RWBLECONF_REG	Configuration register
0x4000000c	BLE_INTCNTL_REG	Interrupt controller register
0x40000010	BLE_INTSTAT_REG	Interrupt status register
0x40000014	BLE_INTRAWSTAT_REG	Interrupt raw status register
0x40000018	BLE_INTACK_REG	Interrupt acknowledge register
0x4000001c	BLE_BASETIMECNT_REG	Base time reference counter
0x40000020	BLE_FINETIMECNT_REG	Fine time reference counter
0x40000024	BLE_BDADDRL_REG	Bluetooth LE device address LSB register
0x40000028	BLE_BDADDRU_REG	Bluetooth LE device address MSB register
0x4000002c	BLE_CURRENTRXDESCPTR_REG	RX Descriptor Pointer for the Receive Buffer Chained List
0x40000030	BLE_DEEPSLCNTL_REG	Deep-Sleep control register
0x40000034	BLE_DEEPSLWKUP_REG	Time (measured in Low Power clock cycles) in Deep Sleep Mode before waking-up the device
0x40000038	BLE_DEEPSLSTAT_REG	Duration of the last deep sleep phase register
0x4000003c	BLE_ENBPRESET_REG	Time in low power oscillator cycles register
0x40000040	BLE_FINECNTCORR_REG	Phase correction value register
0x40000044	BLE_BASETIMECNTCORR_REG	Base Time Counter
0x40000050	BLE_DIAGCNTL_REG	Diagnostics Register
0x40000054	BLE_DIAGSTAT_REG	Debug use only
0x40000058	BLE_DEBUGADDMAX_REG	Upper limit for the memory zone
0x4000005c	BLE_DEBUGADDMIN_REG	Lower limit for the memory zone
0x40000060	BLE_ERRORTYPESTAT_REG	Error Type Status registers
0x40000064	BLE_SWPROFILING_REG	Software Profiling register
0x40000074	BLE_RADIOCNTL1_REG	Radio interface control register
0x40000080	BLE_RADIOPWRUPDN_REG	RX/TX power up/down phase register
0x40000090	BLE_ADVCHMAP_REG	Advertising Channel Map
0x400000a0	BLE_ADVTIM_REG	Advertising Packet Interval

Address	Register	Description
0x400000a4	BLE_ACTSCANSTAT_REG	Active scan register
0x400000b0	BLE_WLPUBADDPTR_REG	Start address of public devices list
0x400000b4	BLE_WLPRIVADDPTR_REG	Start address of private devices list
0x400000b8	BLE_WLNBDEV_REG	Devices in white list
0x400000c0	BLE_AESCNTL_REG	Start AES register
0x400000c4	BLE_AESKEY31_0_REG	AES encryption key
0x400000c8	BLE_AESKEY63_32_REG	AES encryption key
0x400000cc	BLE_AESKEY95_64_REG	AES encryption key
0x400000d0	BLE_AESKEY127_96_REG	AES encryption key
0x400000d4	BLE_AESPTR_REG	Pointer to the block to encrypt/decrypt
0x400000d8	BLE_TXMICVAL_REG	AES/CCM plain MIC value
0x400000dc	BLE_RXMICVAL_REG	AES/CCM plain MIC value
0x400000e0	BLE_RFTESTCNTL_REG	RF Testing Register
0x400000e4	BLE_RFTESTTXSTAT_REG	RF Testing Register
0x400000e8	BLE_RFTESTRXSTAT_REG	RF Testing Register
0x400000f0	BLE_TIMGENCNTL_REG	Timing Generator Register
0x400000f4	BLE_GROSSTIMTGT_REG	Gross Timer Target value
0x400000f8	BLE_FINETIMTGT_REG	Fine Timer Target value
0x400000fc	BLE_SAMPLECLK_REG	Samples the Base Time Counter
0x40000100	BLE_COEXIFCNTL0_REG	Coexistence interface Control 0 Register
0x40000104	BLE_COEXIFCNTL1_REG	Coexistence interface Control 1 Register
0x40000108	BLE_BLEMPRIO0_REG	Coexistence interface Priority 0 Register
0x4000010c	BLE_BLEMPRIO1_REG	Coexistence interface Priority 1 Register
0x40000200	BLE_CNTL2_REG	Bluetooth LE Control Register 2
0x40000208	BLE_EM_BASE_REG	Exchange Memory Base Register
0x4000020c	BLE_DIAGCNTL2_REG	Debug use only
0x40000210	BLE_DIAGCNTL3_REG	Debug use only

Table 62: BLE_RWBLECNTL_REG (0x40000000)

Bit	Mode	Symbol/Description	Reset
31	R0/W	MASTER_SOFT_RST	0x0

Bit	Mode	Symbol/Description	Reset
		Reset the complete Bluetooth LE Core except registers and timing generator, when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0.	
30	R0/W	MASTER_TGSOFT_RST Reset the timing generator, when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0.	0x0
29	R/W	REG_SOFT_RST Reset the complete register block, when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0. Note that INT STAT will not be cleared, so you should also write to BLE_INTACK_REG after the software reset.	0x0
28	R0/W	SWINT_REQ Forces the generation of ble_sw_irq when written with 1, and proper masking is set. Resets at 0 when action is performed. No action happens if it is written with 0.	0x0
26	R0/W	RFTEST_ABORT Abort the current RF Testing defined as per CS-FORMAT when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0. Note that when RFTEST_ABORT is requested: 1) In case of infinite TX, the Packet Controller FSM stops at the end of the current byte in process, and processes accordingly the packet CRC. 2) In case of Infinite RX, the Packet Controller FSM either stops as the end of the current Packet reception (if Access address has been detected), or simply stop the processing switching off the RF.	0x0
25	R0/W	ADVERT_ABORT Abort the current Advertising event when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0.	0x0
24	R0/W	SCAN_ABORT Abort the current scan window when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0.	0x0
22	R/W	MD_DSB 0: Normal operation of MD bits management 1: Allow a single TX/RX exchange whatever the MD bits are - value forced by software from TX Descriptor - value just saved in RX Descriptor during reception	0x0
21	R/W	SN_DSB 0: Normal operation of Sequence number 1: Sequence Number Management disabled - value forced by software from TX Descriptor - value ignored in RX, where no SN error reported	0x0
20	R/W	NESN_DSB 0: Normal operation of Acknowledge 1: Acknowledge scheme disabled - value forced by software from TX Descriptor - value ignored in RX, where no NESN error reported	0x0
19	R/W	CRYPT_DSB 0: Normal operation. Encryption/Decryption enabled 1: Encryption/Decryption disabled	0x0

Bit	Mode	Symbol/Description	Reset
		Note that if CS-CRYPT_EN is set, then MIC is generated, and only data encryption is disabled, meaning data sent are plain data.	
18	R/W	WHIT_DSB 0: Normal operation. Whitening enabled 1: Whitening disabled	0x0
17	R/W	CRC_DSB 0: Normal operation. CRC removed from data stream. 1: CRC stripping disabled on RX packets, CRC replaced by 0x000 in TX.	0x0
16	R/W	HOP_REMAP_DSB 0: Normal operation. Frequency Hopping Remapping algorithm enabled 1: Frequency Hopping Remapping algorithm disabled	0x0
13:12	R/W	- Reserved	0x0
9	R/W	ADVERTFILT_EN Advertising Channels Error Filtering Enable control 0: Bluetooth LE Core reports all errors to RW-Bluetooth LE Software 1: Bluetooth LE Core reports only correctly received packet, without error to RW-Bluetooth LE Software	0x0
8	R/W	RWBLE_EN 0: Disable Bluetooth LE Core Exchange Table pre-fetch mechanism 1: Enable Bluetooth LE Core Exchange table pre-fetch mechanism	0x0
7:4	R/W	RXWINSZDEF Default RX Window size in µs. Used when device: - is master connected - performs its second receipt. 0 is not a valid value. Recommended value is 10 (in decimal).	0x0
2:0	R/W	SYNCERR Indicates the maximum number of errors allowed to recognize the synchronization word	0x0

Table 63: BLE_VERSION_REG (0x40000004)

Bit	Mode	Symbol/Description	Reset
31:24	R	TYP Bluetooth LE Core Type	0x7
23:16	R	REL Bluetooth LE Core version Major release number	0x1
15:8	R	UPG Bluetooth LE Core Upgrade number	0x0
7:0	R	BUILD Bluetooth LE Core Build number	0x0

Table 64: BLE_RWBLECONF_REG (0x40000008)

Bit	Mode	Symbol/Description	Reset
29:24	R	ADD_WIDTH Value of the RW_BLE_ADDRESS_WIDTH parameter concerted into binary	0xF
22:16	R	RFIF Radio Interface ID	0x2
13:8	R	CLK_SEL Operating Frequency (in MHz)	0x0
6	R	DECIPHER 0: AES deciphering not present	0x0
5	R	DMMODE 0: Bluetooth LE Core is used as a standalone Bluetooth LE device	0x0
4	R	INTMODE 1: Interrupts are trigger level generated, stays active at 1 till acknowledgement	0x1
3	R	COEX 1: WLAN Coexistence mechanism present	0x1
2	R	USEDBG 1: Diagnostic port instantiated	0x1
1	R	USECRYPT 1: AES-CCM Encryption block present	0x1
0	R	BUSWIDTH Processor bus width: 1: 32 bits	0x1

Table 65: BLE_INTCNTL_REG (0x4000000C)

Bit	Mode	Symbol/Description	Reset
15	R/W	CSCNTDEVMSK CSCNT interrupt mask during event. This bit allows to enable CSCNT interrupt generation during events (advertising, scanning, initiating, and connection) 0: CSCNT Interrupt not generated during events 1: CSCNT Interrupt generated during events	0x1
14:10	R	- Reserved	0x0
9	R/W	SWINTMSK Software triggered interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x0
8	R/W	EVENTAPFAINTMSK End of event/anticipated pre-fetch abort interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x1
7	R/W	FINETGTIMINTMSK Fine Target Timer Mask	0x0

Bit	Mode	Symbol/Description	Reset
		0: Interrupt not generated 1: Interrupt generated	
6	R/W	GROSSTGTIMINTMSK Gross Target Timer Mask 0: Interrupt not generated 1: Interrupt generated	0x0
5	R/W	ERRORINTMSK Error Interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x0
4	R/W	CRYPTINTMSK Encryption engine Interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x1
3	R/W	EVENTINTMSK End of event Interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x1
2	R/W	SLPINTMSK Sleep Mode Interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x1
1	R/W	RXINTMSK Rx Interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x1
0	R/W	CSCNTINTMSK 625 µs Base Time Interrupt Mask 0: Interrupt not generated 1: Interrupt generated	0x1

Table 66: BLE_INTSTAT_REG (0x40000010)

Bit	Mode	Symbol/Description	Reset
9	R	SWINTSTAT Software triggered interrupt status 0: No software triggered interrupt 1: A software triggered interrupt is pending	0x0
8	R	EVENTAPFAINTSTAT End of event/Anticipated Pre-Fetch Abort interrupt status 0: No End of Event interrupt 1: An End of Event interrupt is pending	0x0
7	R	FINETGTIMINTSTAT Masked Fine Target Timer Error interrupt status 0: No Fine Target Timer interrupt	0x0

Bit	Mode	Symbol/Description	Reset
		1: A Fine Target Timer interrupt is pending	
6	R	GROSSTGTIMINTSTAT Masked Gross Target Timer interrupt status 0: No Gross Target Timer interrupt 1: A Gross Target Timer interrupt is pending	0x0
5	R	ERRORINTSTAT Masked Error interrupt status 0: No Error interrupt 1: An Error interrupt is pending	0x0
4	R	CRYPTINTSTAT Masked Encryption engine interrupt status 0: No Encryption/Decryption interrupt 1: An Encryption/Decryption interrupt is pending	0x0
3	R	EVENTINTSTAT Masked End of Event interrupt status 0: No End of Advertising/Scanning/Connection interrupt 1: An End of Advertising/Scanning/Connection interrupt is pending	0x0
2	R	SLPINTSTAT Masked Sleep interrupt status 0: No End of Sleep Mode interrupt 1: An End of Sleep Mode interrupt is pending	0x0
1	R	RXINTSTAT Masked Packet Reception interrupt status 0: No RX interrupt 1: An RX interrupt is pending	0x0
0	R	CSCNTINTSTAT Masked 625 μs base time reference interrupt status 0: No 625 μs Base Time interrupt 1: A 625 μs Base Time interrupt is pending	0x0

Table 67: BLE_INTRAWSTAT_REG (0x40000014)

Bit	Mode	Symbol/Description	Reset
9	R	SWINTRAWSTAT Software triggered interrupt raw status 0: No software triggered interrupt 1: A software triggered interrupt is pending	0x0
8	R	EVENTAPFAINTRAWSTAT End of event/Anticipated Pre-Fetch Abort interrupt raw status 0: No End of Event interrupt 1: An End of Event interrupt is pending	0x0
7	R	FINETGTIMINTRAWSTAT Fine Target Timer Error interrupt raw status 0: No Fine Target Timer interrupt 1: A Fine Target Timer interrupt is pending	0x0

Bit	Mode	Symbol/Description	Reset
6	R	GROSSTGTIMINTRAWSTAT Gross Target Timer interrupt raw status 0: No Gross Target Timer interrupt 1: A Gross Target Timer interrupt is pending	0x0
5	R	ERRORINTRAWSTAT Error interrupt raw status 0: No Error interrupt 1: An Error interrupt is pending	0x0
4	R	CRYPTINTRAWSTAT Encryption engine interrupt raw status 0: No Encryption/Decryption interrupt 1: An Encryption/Decryption interrupt is pending	0x0
3	R	EVENTINTRAWSTAT End of Event interrupt raw status 0: No End of Advertising/Scanning/Connection interrupt 1: An End of Advertising/Scanning/Connection interrupt is pending	0x0
2	R	SLPINTRAWSTAT Sleep interrupt raw status 0: No End of Sleep Mode interrupt 1: An End of Sleep Mode interrupt is pending	0x0
1	R	RXINTRAWSTAT Packet Reception interrupt raw status 0: No RX interrupt 1: An RX interrupt is pending	0x0
0	R	CSCNTINTRAWSTAT 625 µs base time reference interrupt raw status 0: No 625 µs Base Time interrupt 1: A 625 µs Base Time interrupt is pending	0x0

Table 68: BLE_INTACK_REG (0x40000018)

Bit	Mode	Symbol/Description	Reset
9	R0/W	SWINTACK Software triggered interrupt acknowledgement bit. Software writing 1 acknowledges the software triggered interrupt. This bit resets SWINTSTAT and SWINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
8	R0/W	EVENTAPFAINTACK End of event/Anticipated Pre-Fetch Abort interrupt acknowledgement bit. Software writing 1 acknowledges the End of event/Anticipated Pre-Fetch Abort interrupt. This bit resets EVENTAPFAINTSTAT and EVENTAPFAINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
7	R0/W	FINETGTIMINTACK Fine Target Timer interrupt acknowledgement bit.	0x0

Bit	Mode	Symbol/Description	Reset
		Software writing 1 acknowledges the Fine Timer interrupt. This bit resets FINETGTIMINTSTAT and FINETGTIMINTRAWSTAT flags. Resets at 0 when action is performed.	
6	R0/W	GROSSTGTIMINTACK Gross Target Timer interrupt acknowledgement bit. Software writing 1 acknowledges the Gross Timer interrupt. This bit resets GROSSTGTIMINTSTAT and GROSSTGTIMINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
5	R0/W	ERRORINTACK Error interrupt acknowledgement bit. Software writing 1 acknowledges the Error interrupt. This bit resets ERRORINTSTAT and ERRORINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
4	R0/W	CRYPTINTACK Encryption engine interrupt acknowledgement bit. Software writing 1 acknowledges the Encryption engine interrupt. This bit resets CRYPTINTSTAT and CRYPTINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
3	R0/W	EVENTINTACK End of Event interrupt acknowledgment bit. Software writing 1 acknowledges the End of Advertising/Scanning/Connection interrupt. This bit resets SLPINTSTAT and SLPINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
2	R0/W	SLPINTACK End of Deep Sleep interrupt acknowledgment bit. Software writing 1 acknowledges the End of Sleep Mode interrupt. This bit resets SLPINTSTAT and SLPINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
1	R0/W	RXINTACK Packet Reception interrupt acknowledgment bit. Software writing 1 acknowledges the RX interrupt. This bit resets RXINTSTAT and RXINTRAWSTAT flags. Resets at 0 when action is performed.	0x0
0	R0/W	CSCNTINTACK 625 µs base time reference interrupt acknowledgment bit. Software writing 1 acknowledges the CLKN interrupt. This bit resets CLKINTSTAT and CLKINTRAWSTAT flags. Resets at 0 when action is performed.	0x0

Table 69: BLE_BASETIMECNT_REG (0x4000001C)

Bit	Mode	Symbol/Description	Reset
26:0	R	BASETIMECNT Value of the 625 µs base time reference counter. Updated each time SAMPCLK is written. Used by the software to synchronize with the hardware.	0x0

Table 70: BLE_FINETIMECNT_REG (0x40000020)

Bit	Mode	Symbol/Description	Reset
9:0	R	FINECNT Value of the current µs fine time reference counter. Updated each time SAMPCLK is written. Used by the software to synchronize with the hardware, and obtain a more precise sleep duration.	0x0

Table 71: BLE_BDADDRL_REG (0x40000024)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	BDADDRL Bluetooth Low Energy Device Address. LSB part.	0x0

Table 72: BLE_BDADDRU_REG (0x40000028)

Bit	Mode	Symbol/Description	Reset
16	R/W	PRIV_NPUB Bluetooth Low Energy Device Address privacy indicator 0: Public Bluetooth Device Address 1: Private Bluetooth Device Address	0x0
15:0	R/W	BDADDRU Bluetooth Low Energy Device Address. MSB part.	0x0

Table 73: BLE_CURRENTRXDESCPTR_REG (0x4000002C)

Bit	Mode	Symbol/Description	Reset
31:16	R/W	ETPTR Exchange Table Pointer that determines the starting point of the Exchange Table.	0x0
14:0	R/W	CURRENTRXDESCPTR RX Descriptor Pointer that determines the starting point of the Receive Buffer Chained List.	0x0

Table 74: BLE_DEEPSLCNTL_REG (0x40000030)

Bit	Mode	Symbol/Description	Reset
31	R/W	EXTWKUPDSB External Wake-Up disable 0: RW-BLE Core can be woken by external wake-up 1: RW-BLE Core cannot be woken up by external wake-up	0x0
15	R	DEEP_SLEEP_STAT Indicator of current Deep Sleep clock mux status: 0: RW-Bluetooth LE Core is not yet in Deep Sleep mode 1: RW-Bluetooth LE Core is in Deep Sleep mode (only low_power_clk is running)	0x0
4	R/W	SOFT_WAKEUP_REQ Wake Up Request from Bluetooth LE Software. Applies when system is in Deep Sleep mode. It wakes up the Bluetooth LE Core when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0.	0x0
3	R0/W	DEEP_SLEEP_CORR_EN	0x0

Bit	Mode	Symbol/Description	Reset
		625 µs base time reference integer and fractional part correction. Applies when system has been woken-up from Deep Sleep mode. It enables Fine Counter and Base Time counter when written with 1. Resets at 0 when action is performed. No action happens if it is written with 0.	
2	R0/W	DEEP_SLEEP_ON 0: Bluetooth LE Core in normal Active mode 1: Request RW-Bluettoh LE Core to switch in Deep Sleep mode This bit is reset on DEEP_SLEEP_STAT falling edge	0x0
1:0	R/W	DEEP_SLEEP_IRQ_EN Always set to 3 when DEEP_SLEEP_ON is set to 1. It controls the generation of BLE_WAKEUP_LP_IRQ.	0x0

Table 75: BLE_DEEPSLWKUP_REG (0x40000034)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	DEEPSLTIME Determines the time in low_power_clk clock cycles to spend in Deep Sleep mode before waking up the device. This ensures a maximum of 37 hours and 16 mn sleep mode capabilities at 32 kHz. This ensures a maximum of 36 hours and 16 mn sleep mode capabilities at 32.768 kHz.	0x0

Table 76: BLE_DEEPSLSTAT_REG (0x40000038)

Bit	Mode	Symbol/Description	Reset
31:0	R	DEEPSLDUR Actual duration of the last deep sleep phase measured in low_power_clk clock cycle. DEEPSLDUR is set to zero at the beginning of the deep sleep phase, and is incremented at each low_power_clk clock cycle until the end of the deep sleep phase.	0x0

Table 77: BLE_ENBPRESET_REG (0x4000003C)

Bit	Mode	Symbol/Description	Reset
31:21	R/W	TWEXT Minimum and recommended value is "TWIRQ_RESET + 1". In the case of wake-up due to an external wake-up request, TWEXT specifies the time delay in low power oscillator cycles to deassert BLE_WAKEUP_LP_IRQ. See also GP_CONTROL_REG[BLE_WAKEUP_REQ]. Range is [0...64 ms] for 32 kHz; [0...62.5 ms] for 32.768 kHz	0x0
20:10	R/W	TWIRQ_SET Minimum value is "TWIRQ_RESET + 1". Time in low power oscillator cycles to set BLE_WAKEUP_LP_IRQ before the Bluetooth LE sleep timer expiration. See also BLE_DEEPSLWKUP_REG[DEEPSLTIME]. Range is [0...64 ms] for 32 kHz; [0...62.5 ms] for 32.768 kHz	0x0
9:0	R/W	TWIRQ_RESET Recommended value is 1. Time in low power oscillator cycles to reset BLE_WAKEUP_LP_IRQ before the Bluetooth LE sleep timer expiration.	0x0

Bit	Mode	Symbol/Description	Reset
		See also BLE_DEEPSLWKUP_REG[DEEPSLTIME]. Range is [0...32 ms] for 32 kHz; [0...31.25 ms] for 32.768 kHz.	

Table 78: BLE_FINECNTCORR_REG (0x40000040)

Bit	Mode	Symbol/Description	Reset
9:0	R/W	FINECNTCORR Phase correction value for the 625 µs reference counter (Fine Counter) in µs	0x0

Table 79: BLE_BASETIMECNTCORR_REG (0x40000044)

Bit	Mode	Symbol/Description	Reset
26:0	R/W	BASETIMECNTCORR Base Time Counter correction value	0x0

Table 80: BLE_DIAGCNTL_REG (0x40000050)

Bit	Mode	Symbol/Description	Reset
31	R/W	DIAG3_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
29:24	R/W	DIAG3 Only relevant when DIAG3_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG3.	0x0
23	R/W	DIAG2_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
21:16	R/W	DIAG2 Only relevant when DIAG2_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG2.	0x0
15	R/W	DIAG1_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
13:8	R/W	DIAG1 Only relevant when DIAG1_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG1.	0x0
7	R/W	DIAG0_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
5:0	R/W	DIAG0 Only relevant when DIAG0_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG0.	0x0

Table 81: BLE_DIAGSTAT_REG (0x40000054)

Bit	Mode	Symbol/Description	Reset
31:24	R	DIAG3STAT Directly connected to ble_dbg3[7:0] output. Debug use only.	0x0
23:16	R	DIAG2STAT Directly connected to ble_dbg2[7:0] output. Debug use only.	0x0
15:8	R	DIAG1STAT Directly connected to ble_dbg1[7:0] output. Debug use only.	0x0
7:0	R	DIAG0STAT Directly connected to ble_dbg0[7:0] output. Debug use only.	0x0

Table 82: BLE_DEBUGADDMAX_REG (0x40000058)

Bit	Mode	Symbol/Description	Reset
31:16	R/W	REG_ADDMAX Upper limit for the Register zone indicated by the reg_inzone flag.	0x0
15:0	R/W	EM_ADDMAX Upper limit for the Exchange Memory zone indicated by the em_inzone flag.	0x0

Table 83: BLE_DEBUGADDMIN_REG (0x4000005C)

Bit	Mode	Symbol/Description	Reset
31:16	R/W	REG_ADDMIN Lower limit for the Register zone indicated by the reg_inzone flag.	0x0
15:0	R/W	EM_ADDMIN Lower limit for the Exchange Memory zone indicated by the em_inzone flag.	0x0

Table 84: BLE_ERRORTYPESTAT_REG (0x40000060)

Bit	Mode	Symbol/Description	Reset
17	R	CONCEVTIRQ_ERROR Indicates whether two consecutive and concurrent ble_event_irq have been generated, and not acknowledged in time by the Bluetooth LE Software. 0: No error 1: Error occurred	0x0
16	R	RXDATA_PTR_ERROR Indicates whether RX data buffer pointer value programmed is null: this is a major programming failure. 0: No error 1: Error occurred	0x0
15	R	TXDATA_PTR_ERROR Indicates whether TX data buffer pointer value programmed is null during Advertising/Scanning/Initiating events, or during Master/Slave connections with non-null packet length: this is a major programming failure. 0: No error 1: Error occurred	0x0

Bit	Mode	Symbol/Description	Reset
14	R	RXDESC_EMPTY_ERROR Indicates whether RX Descriptor pointer value programmed in register is null: this is a major programming failure. 0: No error 1: Error occurred	0x0
13	R	TXDESC_EMPTY_ERROR Indicates whether TX Descriptor pointer value programmed in Control Structure is null during Advertising/Scanning/Initiating events: this is a major programming failure. 0: No error 1: Error occurred	0x0
12	R	CSFORMAT_ERROR Indicates whether CS-FORMAT has been programmed with an invalid value: this is a major software programming failure. 0: No error 1: Error occurred	0x0
11	R	LLCHMAP_ERROR Indicates Link Layer Channel Map error, happens when actual number of CS-LLCHMAP bit set to one is different from CS-NBCHGOOD at the beginning of Frequency Hopping process 0: No error 1: Error occurred	0x0
10	R	ADV_UNDERRUN Indicates Advertising Interval Under run, occurs if time between two consecutive Advertising packet (in Advertising mode) is lower than the expected value. 0: No error 1: Error occurred	0x0
9	R	IFS_UNDERRUN Indicates Inter Frame Space Under run, occurs if IFS time is not enough to update and read Control Structure/Descriptors, and/or White List parsing is not finished and/or Decryption time is too long to be finished on time 0: No error 1: Error occurred	0x0
8	R	WHITELIST_ERROR Indicates White List Timeout error, occurs if White List parsing is not finished on time 0: No error 1: Error occurred	0x0
7	R	EVT_CNTL_APFM_ERROR Indicates Anticipated Pre-Fetch Mechanism error: happens when two consecutive events are programmed, and when the first event is not completely finished while second pre-fetch instant is reached. 0: No error 1: Error occurred	0x0
6	R	EVT_SCHDL_APFM_ERROR Indicates Anticipated Pre-Fetch Mechanism error: happens when two consecutive events are programmed, and when the first event is not completely finished while second pre-fetch instant is reached.	0x0

Bit	Mode	Symbol/Description	Reset
		0: No error 1: Error occurred	
5	R	EVT_SCHDL_ENTRY_ERROR Indicates Event Scheduler faced Invalid timing programing on two consecutive ET entries (for example, first one with 624 s offset and second one with no offset) 0: No error 1: Error occurred	0x0
4	R	EVT_SCHDL_EMACC_ERROR Indicates Event Scheduler Exchange Memory access error, happens when Exchange Memory accesses are not served in time, and blocks the Exchange Table entry read 0: No error 1: Error occurred	0x0
3	R	RADIO_EMACC_ERROR Indicates Radio Controller Exchange Memory access error, happens when Exchange Memory accesses are not served in time and data are corrupted. 0: No error 1: Error occurred	0x0
2	R	PKTCNTL_EMACC_ERROR Indicates Packet Controller Exchange Memory access error, happens when Exchange Memory accesses are not served in time and TX/RX data are corrupted 0: No error 1: Error occurred	0x0
1	R	RXCRYPT_ERROR Indicates real time decryption error, happens when AES-CCM decryption is too slow compared to Packet Controller requests. A 16-bytes block has to be decrypted prior the next block is received by the Packet Controller 0: No error 1: Error occurred	0x0
0	R	TXCRYPT_ERROR Indicates Real Time encryption error, happens when AES-CCM encryption is too slow compared to Packet Controller requests. A 16-bytes block has to be encrypted and prepared on Packet Controller request, and needs to be ready before the Packet Controller has to send ti 0: No error 1: Error occurred	0x0

Table 85: BLE_SWPROFILING_REG (0x40000064)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	SWPROFVAL Software Profiling register: used by Bluetooth LE Software for profiling purpose: this value is copied on Diagnostic port.	0x0

Table 86: BLE_RADIOCNTL1_REG (0x40000074)

Bit	Mode	Symbol/Description	Reset
31:21	-	-	0x0

Bit	Mode	Symbol/Description	Reset
		Reserved	
20:16	R/W	XRFSEL Extended radio selection field, Must be set to "2".	0x0

Table 87: BLE_RADIOPWRUPDN_REG (0x40000080)

Bit	Mode	Symbol/Description	Reset
30:24	R/W	RTRIP_DELAY Defines round trip delay value. This value corresponds to the addition of data latency in TX and data latency in RX. The value is in μs.	0x0
23:16	R/W	RXPWRUP This register holds the length in μs of the RX powerup phase for the current radio device. The default value is 210 μs (reset value). Operating range depends on the selected radio.	0xD2
11:8	R/W	TXPWRDN This register extends the length in μs of the TX powerdown phase for the current radio device. The default value is 3 μs (reset value). Operating range depends on the selected radio.	0x3
7:0	R/W	TXPWRUP This register holds the length in μs of the TX powerup phase for the current radio device. The default value is 210 μs (reset value). Operating range depends on the selected radio.	0xD2

Table 88: BLE_ADVCHMAP_REG (0x40000090)

Bit	Mode	Symbol/Description	Reset
2:0	R/W	ADVCHMAP Advertising Channel Map, defined as per the advertising connection settings. Contains advertising channels index 37 to 39. If ADVCHMAP[i] equals: 0: Do not use data channel i+37 1: Use data channel i+37	0x7

Table 89: BLE_ADVTIM_REG (0x400000A0)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	ADVINT Advertising Packet Interval defines the time interval in between two ADV_xxx packet sent. Value is in μs. Value to program depends on the used Advertising Packet type and the device filtering policy.	0x0

Table 90: BLE_ACTSCANSTAT_REG (0x400000A4)

Bit	Mode	Symbol/Description	Reset
24:16	R	BACKOFF Active scan mode back-off counter initialization value.	0x1
8:0	R	UPPERLIMIT Active scan mode upper limit counter value.	0x1

Table 91: BLE_WLPUBADDPTR_REG (0x400000B0)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	WLPUBADDPTR Start address pointer of the public devices whitelist.	0x0

Table 92: BLE_WLPRIVADDPTR_REG (0x400000B4)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	WLPRIVADDPTR Start address pointer of the private devices whitelist.	0x0

Table 93: BLE_WLNBDEV_REG (0x400000B8)

Bit	Mode	Symbol/Description	Reset
15:8	R/W	NBPRIVDEV Number of private devices in the whitelist.	0x0
7:0	R/W	NBPUBDEV Number of public devices in the whitelist.	0x0

Table 94: BLE_AESCNTL_REG (0x400000C0)

Bit	Mode	Symbol/Description	Reset
1	R/W	AES_MODE 0: Cipher mode 1: Decipher mode	0x0
0	R0/W	AES_START Writing 1 starts AES-128 ciphering/deciphering process. This bit is reset when the process is finished (ble_crypt_irq interrupt occurs, even masked).	0x0

Table 95: BLE_AESKEY31_0_REG (0x400000C4)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	AESKEY31_0 AES encryption 128-bit key. Bit 31 down to 0	0x0

Table 96: BLE_AESKEY63_32_REG (0x400000C8)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	AESKEY63_32 AES encryption 128-bit key. Bit 63 down to 32	0x0

Table 97: BLE_AESKEY95_64_REG (0x400000CC)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	AESKEY95_64	0x0

Bit	Mode	Symbol/Description	Reset
		AES encryption 128-bit key. Bit 95 down to 64	

Table 98: BLE_AESKEY127_96_REG (0x400000D0)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	AESKEY127_96 AES encryption 128-bit key. Bit 127 down to 96	0x0

Table 99: BLE_AESPTR_REG (0x400000D4)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	AESPTR Pointer to the memory zone where the block to cipher/decipher using AES-128 is stored.	0x0

Table 100: BLE_TXMICVAL_REG (0x400000D8)

Bit	Mode	Symbol/Description	Reset
31:0	R	TXMICVAL AES-CCM plain MIC value. Valid on when MIC has been calculated (in TX).	0x0

Table 101: BLE_RXMICVAL_REG (0x400000DC)

Bit	Mode	Symbol/Description	Reset
31:0	R	RXMICVAL AES-CCM plain MIC value. Valid on when MIC has been extracted from RX packet.	0x0

Table 102: BLE_RFTESTCNTL_REG (0x400000E0)

Bit	Mode	Symbol/Description	Reset
31	R/W	INFINITERX Applicable in RF Test Mode only 0: Normal mode of operation 1: Infinite RX window	0x0
27	R/W	RXPKTCNTEN Applicable in RF Test Mode only 0: RX packet count disabled 1: RX packet count enabled, and reported in CS-RXCCMPKTCNT and BLE_RFTESTRXSTAT_REG[RXPKTCNT] on RF abort command	0x0
15	R/W	INFINITETX Applicable in RF Test Mode only 0: Normal mode of operation 1: Infinite TX packet/Normal start of a packet but endless payload	0x0
14	R/W	TXLENGTHSRC Applicable only in TX/RX RF Test mode	0x0

Bit	Mode	Symbol/Description	Reset
		0: Normal mode of operation: TxDESC-TXADVLEN controls the TX packet payload size 1: Uses BLE_RFTESTCNTL_REG[TXLENGTH] packet length (can support up to 512 bytes transmit)	
13	R/W	PRBSTYPE Applicable only in TX/RX RF Test mode 0: TX Packet Payload are PRBS9 type 1: TX Packet Payload are PRBS15 type	0x0
12	R/W	TXPLDSRC Applicable only in TX/RX RF Test mode 0: TX Packet Payload source is the Control Structure 1: TX Packet Payload are PRBS generator	0x0
11	R/W	TXPKTCNTEN Applicable in RF Test mode only 0: TX packet count disabled 1: TX packet count enabled, and reported in CS-TXCCMPKTCNT and BLE_RFTESTTXSTAT_REG[TXPKTCNT] on RF abort command	0x0
8:0	R/W	TXLENGTH Applicable only for TX/RX RF Test mode, and valid when BLE_RFTESTCNTL_REG[TXLENGTHSRC] = 1 TX packet length in number of byte.	0x0

Table 103: BLE_RFTESTTXSTAT_REG (0x400000E4)

Bit	Mode	Symbol/Description	Reset
31:0	R	TXPKTCNT Reports number of transmitted packet during Test modes. Value is valid if BLE_RFTESTCNTL_REG[TXPKTCNTEN] is set.	0x0

Table 104: BLE_RFTESTRXSTAT_REG (0x400000E8)

Bit	Mode	Symbol/Description	Reset
31:0	R	RXPKTCNT Reports number of correctly received packet during Test Modes (no sync error, no CRC error). Value is valid if BLE_RFTESTCNTL_REG[RXPKTCNTEN] is set	0x0

Table 105: BLE_TIMGENCNTL_REG (0x400000F0)

Bit	Mode	Symbol/Description	Reset
31	R/W	APFM_EN Controls the Anticipated pre-Fetch Abort mechanism 0: Disabled 1: Enabled	0x1
25:16	R/W	PREFETCHABORT_TIME Defines the instant in μs at which immediate abort is required after anticipated pre-fetch abort.	0x1FE

Bit	Mode	Symbol/Description	Reset
8:0	R/W	PREFETCH_TIME Defines Exchange Table pre-fetch instant in µs	0x96

Table 106: BLE_GROSSTIMTGT_REG (0x400000F4)

Bit	Mode	Symbol/Description	Reset
22:0	R/W	GROSSTARGET Gross Timer Target value on which a ble_grosstgtim_irq must be generated. This timer has a precision of 10 ms: interrupt is generated only when GROSSTARGET[22:0] = BASETIMECNT[26:4] and BASETIMECNT[3:0] = 0.	0x0

Table 107: BLE_FINETIMTGT_REG (0x400000F8)

Bit	Mode	Symbol/Description	Reset
26:0	R/W	FINETARGET Fine Timer Target value on which a ble_finetgtim_irq must be generated. This timer has a precision of 625 µs: interrupt is generated only when FINETARGET = BASETIMECNT	0x0

Table 108: BLE_SAMPLECLK_REG (0x400000FC)

Bit	Mode	Symbol/Description	Reset
31:1	-	- Reserved	0x0
0	R0/W	SAMP Writing a 1 samples the Base Time Counter value in BASETIMECNT register. Resets at 0 when action is performed.	0x0

Table 109: BLE_COEXIFCNTL0_REG (0x40000100)

Bit	Mode	Symbol/Description	Reset
21:20	R/W	WLCRXPRIOMODE Defines Bluetooth Low Energy packet ble_rx mode behavior. 00: RX indication excluding RX Power up delay (starts when correlator is enabled) 01: RX indication including RX Power up delay 10: RX High priority indicator 11: n/a	0x0
17:16	R/W	WLCTXPRIOMODE Defines Bluetooth Low Energy packet ble_tx mode behavior 00: TX indication excluding TX Power up delay 01: TX indication including TX Power up delay 10: TX High priority indicator 11: n/a	0x0
7:6	R/W	WLANTXMSK Determines how wlan_tx impact Bluetooth LE TX and RX 00: wlan_tx has no impact (default mode) 01: wlan_tx can stop Bluetooth LE TX, no impact on Bluetooth LE RX	0x0

Bit	Mode	Symbol/Description	Reset
		10: wlan_tx can stop Bluetooth LE RX, no impact on Bluetooth LE TX 11: wlan_tx can stop both Bluetooth LE TX and Bluetooth LE RX	
5:4	R/W	WLANRXMSK Determines how wlan_rx impact Bluetooth LE TX and RX 00: wlan_rx has no impact 01: wlan_rx can stop Bluetooth LE TX, no impact on Bluetooth LE RX (default mode) 10: wlan_rx can stop Bluetooth LE RX, no impact on Bluetooth LE TX 11: wlan_rx can stop both Bluetooth LE TX and Bluetooth LE RX	0x1
1	R/W	SYNCGEN_EN Determines whether ble_sync is generated or not. 0: ble_sync pulse not generated 1: ble_sync pulse generated	0x0
0	R/W	COEX_EN Enable/Disable control of the MWS/WLAN Coexistence control 0: Coexistence interface disabled 1: Coexistence interface enabled	0x0

Table 110: BLE_COEXIFCNTL1_REG (0x40000104)

Bit	Mode	Symbol/Description	Reset
28:24	R/W	WLCPRXTHR Applies on ble_rx if WLCRXPRIOMODE equals 10 Determines the threshold for RX priority setting. If ble_pti[3:0] output value is greater than WLCPRXTHR, then RX Bluetooth Low Energy priority is considered as high, and must be provided to the WLAN coexistence interface.	0x0
20:16	R/W	WLCPTXTHR Applies on ble_tx if WLCTXPRIOMODE equals 10 Determines the threshold for priority setting. If ble_pti[3:0] output value is greater than WLCPTXTHR, then TX Bluetooth Low Energy priority is considered as high, and must be provided to the WLAN coexistence interface.	0x0
14:8	R/W	WLCPDURATION Applies on ble_tx if WLCTXPRIOMODE equals 10. Applies on ble_rx if WLCRXPRIOMODE equals 10. Determines how many s the priority information must be maintained. Note that if WLCPDURATION = 0x00, then TX/RX priority levels are maintained till TX/RX EN are de-asserted.	0x0
6:0	R/W	WLCPDELAY Applies on ble_tx if WLCTXPRIOMODE equals 10. Applies on ble_rx if WLCRXPRIOMODE equals 10. Determines the delay (in μs) in TX/RX enables rises the time Bluetooth Low energy TX/RX priority has to be provided.	0x0

Table 111: BLE_BLEMPRIO0_REG (0x40000108)

Bit	Mode	Symbol/Description	Reset
31:28	R/W	BLEM7 Set Priority value for Passive Scanning	0x3
27:24	R/W	BLEM6 Set Priority value for Non-Connectable Advertising	0x4
23:20	R/W	BLEM5 Set Priority value for Connectable Advertising BLE message	0x8
19:16	R/W	BLEM4 Set Priority value for Active Scanning Bluetooth LE message	0x9
15:12	R/W	BLEM3 Set Priority value for Initiating (Scanning) Bluetooth LE message	0xA
11:8	R/W	BLEM2 Set Priority value for Data Channel transmission Bluetooth LE message	0xD
7:4	R/W	BLEM1 Set Priority value for LLCP Bluetooth LE message	0xE
3:0	R/W	BLEM0 Set Priority value for Initiating (Connection Request Response) Bluetooth LE message	0xF

Table 112: BLE_BLEMPRIO1_REG (0x4000010C)

Bit	Mode	Symbol/Description	Reset
31:28	R/W	BLEMDEFAULT Set the default priority value for other Bluetooth LE messages than those defined above.	0x3

Table 113: BLE_CNTL2_REG (0x40000200)

Bit	Mode	Symbol/Description	Reset
31:25	R	- Reserved	0x0
24	R/W	BLE_PHY_ERR_MSK_N	0x0
23	R/W	BLE_ARP_ERR_MSK_N When cleared to "0", then it masks the BLE_ARP_ERR_STAT to not trigger the BLE_ERROR_IRQ.	0x0
22	RW1C	BLE_ARP_PHY_ERR_STAT When set to "1", then an error occurred in Bluetooth LE ARP subblock and the BLE_GEN_IRQ is asserted. It is set if ARP_ERROR or PHY_ERROR is asserted and if BLE_ARP_ERR_MSK is set to "1". Writing the value "1" acknowledges and clears this field.	0x0
21	R/W	BLE_RSSI_SEL 0: (default) Select Peak-hold RSSI value during the SYNC_FOUND event:	0x0

Bit	Mode	Symbol/Description	Reset
		CS->RXRSSI[7:0] = RF_RSSI_RESULT_REG->RSSI_LATCHED_RD[9:2]. 1: Select the Average RSSI value during the SYNC_FOUND event: CS->RXRSSI[7:0] = RF_RSSI_RESULT_REG->RSSI_AVG_RD[9:2].	
20	R	WAKEUPLPSTAT The status of the BLE_WAKEUP_LP_IRQ. The Interrupt Service Routine of BLE_WAKEUP_LP_IRQ should return only when the WAKEUPLPSTAT is cleared. Note that BLE_WAKEUP_LP_IRQ is automatically acknowledged after the power up of the Radio Subsystem, plus one Low Power Clock period.	0x0
19	R/W	SW_RPL_SPI Keep to 0.	0x0
18	R/W	BB_ONLY Keep to 0.	0x0
17	R/W	BLE_PTI_SOURCE_SEL 0: Provide to COEX block the PTI value indicated by the Control Structure. Recommended value is "0". 1: Provide to COEX block the PTI value generated dynamically by the Bluetooth LE core, which is based on the PTI of the Control Structure.	0x0
16:15	R	- Reserved	0x0
14:9	R/W	BLE_CLK_SEL Bluetooth LE Clock Select. Specifies the Bluetooth LE master clock absolute frequency in MHz. Typical values are 16 and 8. Value depends on the selected XTAL frequency and the value of CLK_RADIO_REG[BLE_DIV] bitfield. For example, if XTAL oscillates at 16 MHz and CLK_RADIO_REG[BLE_DIV] = 1 (divide by 2), then Bluetooth LE master clock frequency is 8 MHz and BLE_CLK_SEL should be set to value 8. The selected Bluetooth LE master clock frequency (affected by BLE_DIV and BLE_CLK_SEL) must be modified and set only during the initialization time, that is before setting BLE_RWBLECNTL_REG[RWBLE_EN] to 1. Also see BLE_RWBLECONF_REG[CLK_SEL].	0x0
8	R	RADIO_PWRDN_ALLOW This active high signal indicates when it is allowed for the Bluetooth LE core (embedded in the Radio sub-system power domain) to be powered down. After the assertion of the BLE_DEEPSLCNTL_REG[DEEP_SLEEP_ON], a hardware sequence based on the Low Power clock causes the assertion of RADIO_PWRDN_ALLOW. The RADIO_PWRDN_ALLOW is cleared to "0" when the Bluetooth LE core exits from the SLEEP state, when the BLE_SLP_IRQ is asserted.	0x0
7	R	MON_LP_CLK The software can only write a "0" to this bit. Whenever a positive edge of the low power clock used by the Bluetooth LE Timers is detected, then the hardware automatically sets this bit to "1". This functionality does not work if Bluetooth LE Timer is in RESET state (see CLK_RADIO_REG[BLE_LP_RESET]). This bit can be used for software synchronization, to debug the low power clock, and so on.	0x0
6	R	BLE_CLK_STAT 0: Bluetooth LE uses low power clock	0x0

Bit	Mode	Symbol/Description	Reset
		1: Bluetooth LE uses master clock	
5:4	R/W	- Reserved	0x0
3	R/W	BLE_DIAG_OVR 1: Overrule BLE_DIAG. 0: BLE_DIAG is not overruled.	0x0
2	R/W	EMACCERRMSK Exchange Memory Access Error Mask: When cleared to "0", the EM_ACC_ERR does not cause the BLE_ERROR_IRQ interrupt. When set to "1", BLE_ERROR_IRQ is generated as long as EM_ACC_ERR is "1".	0x1
1	R0/W	EMACCERRACK Exchange Memory Access Error Acknowledge. When the software writes a "1" to this bit, then the EMACCERRSTAT bit is cleared. When the software writes "0", it has no effect. The read value is always "0".	0x0
0	R	EMACCERRSTAT Exchange Memory Access Error Status: The bit is read-only and can be cleared only by writing a "1" at EMACCERRACK bitfield. This bit is set to "1" by the hardware when the controller accesses an EM page that is not mapped according to the EM_MAPPING value. When this bit is "1", then BLE_ERROR_IRQ is asserted as long as EMACCERRMSK is "1".	0x0

Table 114: BLE_EM_BASE_REG (0x40000208)

Bit	Mode	Symbol/Description	Reset
31:17	R	- Reserved	0x0
16:10	R/W	BLE_EM_BASE_16_10 The physical address on the system memory map of the base of the Exchange Memory.	0x0
9:0	R	- Reserved	0x0

Table 115: BLE_DIAGCNTL2_REG (0x4000020C)

Bit	Mode	Symbol/Description	Reset
31	R/W	DIAG7_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
30	R	- Reserved	0x0
29:24	R/W	DIAG7	0x0

Bit	Mode	Symbol/Description	Reset
		Only relevant when DIAG7_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG7.	
23	R/W	DIAG6_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
22	R	- Reserved	0x0
21:16	R/W	DIAG6 Only relevant when DIAG6_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG6.	0x0
15	R/W	DIAG5_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
14	R	- Reserved	0x0
13:8	R/W	DIAG5 Only relevant when DIAG5_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG5.	0x0
7	R/W	DIAG4_EN 0: Disable diagnostic port 0 output. All outputs are set to 0x0. 1: Enable diagnostic port 0 output.	0x0
6	R	- Reserved	0x0
5:0	R/W	DIAG4 Only relevant when DIAG4_EN = 1. Selection of the outputs that must be driven to the diagnostic port BLE_DIAG4.	0x0

Table 116: BLE_DIAGCNTL3_REG (0x40000210)

Bit	Mode	Symbol/Description	Reset
31	R/W	DIAG7_INV If set, then the specific diagnostic bit is inverted.	0x0
30:28	R/W	DIAG7_BIT Selects which bit from the DIAG7 word is forwarded to bit 7 of the Bluetooth LE DIagnostic Port.	0x0
27	R/W	DIAG6_INV If set, then the specific diagnostic bit is inverted.	0x0
26:24	R/W	DIAG6_BIT Selects which bit from the DIAG6 word is forwarded to bit 6 of the Bluetooth LE DIagnostic Port.	0x0
23	R/W	DIAG5_INV If set, then the specific diagnostic bit is inverted.	0x0
22:20	R/W	DIAG5_BIT	0x0

Bit	Mode	Symbol/Description	Reset
		Selects which bit from the DIAG5 word is forwarded to bit 5 of the Bluetooth LE DIagnostic Port.	
19	R/W	DIAG4_INV If set, then the specific diagnostic bit is inverted.	0x0
18:16	R/W	DIAG4_BIT Selects which bit from the DIAG4 word is forwarded to bit 4 of the Bluetooth LE DIagnostic Port.	0x0
15	R/W	DIAG3_INV If set, then the specific diagnostic bit is inverted.	0x0
14:12	R/W	DIAG3_BIT Selects which bit from the DIAG3 word is forwarded to bit 3 of the Bluetooth LE DIagnostic Port.	0x0
11	R/W	DIAG2_INV If set, then the specific diagnostic bit is inverted.	0x0
10:8	R/W	DIAG2_BIT Selects which bit from the DIAG2 word is forwarded to bit 2 of the Bluetooth LE DIagnostic Port.	0x0
7	R/W	DIAG1_INV If set, then the specific diagnostic bit is inverted.	0x0
6:4	R/W	DIAG1_BIT Selects which bit from the DIAG1 word is forwarded to bit 1 of the Bluetooth LE DIagnostic Port.	0x0
3	R/W	DIAG0_INV If set, then the specific diagnostic bit is inverted.	0x0
2:0	R/W	DIAG0_BIT Selects which bit from the DIAG0 word is forwarded to bit 0 of the Bluetooth LE DIagnostic Port.	0x0

31.3 Clock Generation and Reset Registers

Table 117: Register map CRG

Address	Register	Description
0x50000000	CLK_AMBA_REG	HCLK, PCLK, divider and clock gates
0x50000002	CLK_FREQ_TRIM_REG	Xtal frequency trimming register
0x50000004	CLK_PER_REG	Peripheral divider register
0x50000008	CLK_RADIO_REG	Radio PLL control register
0x5000000a	CLK_CTRL_REG	Clock control register
0x50000010	PMU_CTRL_REG	Power Management Unit control register
0x50000012	SYS_CTRL_REG	System Control register
0x50000014	SYS_STAT_REG	System status register
0x50000016	TRIM_CTRL_REG	Control trimming of the XTAL32M
0x50000018	RAM_PWR_CTRL_REG	Control power state of System RAMS
0x50000020	CLK_RC32K_REG	32 kHz RC oscillator register
0x50000022	CLK_XTAL32K_REG	32 kHz XTAL oscillator register
0x50000024	CLK_RC32M_REG	Fast RC control register
0x50000026	CLK_RCX_REG	RCX-oscillator control register
0x50000028	BANDGAP_REG	Bandgap trimming
0x5000002a	ANA_STATUS_REG	Status bit of analog (power management) circuits
0x50000030	XTAL32M_START_REG	Trim values for XTAL32M
0x50000032	XTAL32M_TRSTAT_REG	Read back value of current XTAL trimming
0x50000034	XTALRDY_CTRL_REG	Control register for XTALRDY IRQ
0x50000038	XTAL32M_CTRL0_REG	Control bits for XTAL32M
0x50000040	POR_PIN_REG	Selects a GPIO pin for POR generation
0x50000042	POR_TIMER_REG	Time for POR to happen
0x50000050	PMU_SLEEP_REG	Bandgap refresh interval during sleep
0x50000052	POWER_CTRL_REG	Power management control
0x50000054	POWER_LEVEL_REG	Power management level and trim settings
0x50000056	DCDC_SLP_CTRL_REG	Control of DCDC in sleep
0x50000058	LDO_CORE_LEVEL_REG	Control the LDO CORE voltage output

Table 118: CLK_AMBA_REG (0x50000000)

Bit	Mode	Symbol/Description	Reset
7	R/W	OTP_ENABLE Clock enable for OTP controller	0x0
6	R/W	- Reserved	0x0
5:4	R/W	PCLK_DIV APB interface clock (PCLK). Divider is cascaded with HCLK_DIV. PCLK is HCLK divided by: 0x0: Divide by 1	0x0

Bit	Mode	Symbol/Description	Reset
		0x1: Divide by 2 0x2: Divide by 4 0x3: Divide by 8	
3:2	R/W	- Reserved	0x0
1:0	R/W	HCLK_DIV AHB interface and microprocessor clock (HCLK). HCLK is source clock divided by: 0x0: Divide by 1 0x1: Divide by 2 0x2: Divide by 4 0x3: Divide by 8	0x0

Table 119: CLK_FREQ_TRIM_REG (0x50000002)

Bit	Mode	Symbol/Description	Reset
7:0	R/W	XTAL32M_TRIM XTAL frequency fine trimming register. 0x00: Highest frequency 0xFF: Lowest frequency	0x80

Table 120: CLK_PER_REG (0x50000004)

Bit	Mode	Symbol/Description	Reset
11	R/W	QUAD_ENABLE Enable the Quadrature clock	0x1
10	R/W	SPI_ENABLE Enable SPI clock	0x0
9:8	R/W	- Reserved	0x0
7	R/W	UART1_ENABLE Enable UART1 clock	0x0
6	R/W	UART2_ENABLE Enable UART2 clock	0x0
5	R/W	I2C_ENABLE Enable I2C clock	0x0
4	R/W	WAKEUPCT_ENABLE Enable Wake-up CaptureTimer clock	0x0
3	R/W	TMR_ENABLE Enable TIMER0 and TIMER2 clock	0x0
2	R/W	- Reserved	0x0
1:0	R/W	TMR_DIV Division factor for TIMER0 0x0: divide by 1	0x0

Bit	Mode	Symbol/Description	Reset
		0x1: divide by 2 0x2: divide by 4 0x3: divide by 8	

Table 121: CLK_RADIO_REG (0x50000008)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7	R/W	BLE_ENABLE Enable the Bluetooth LE core clocks	0x0
6	R/W	BLE_LP_RESET Reset for the Bluetooth LE LP timer	0x1
5:4	R/W	BLE_DIV Division factor for Bluetooth LE core blocks 0x0: divide by 1 0x1: divide by 2 0x2: divide by 4 0x3: divide by 8 The programmed frequency should not be lower than 8 MHz and not faster than the programmed CPU clock frequency. Also, see BLE_CNTL2_REG[BLE_CLK_SEL].	0x0
3	R/W	RFCU_ENABLE Enable the RF control Unit clock	0x0
2	R/W	- Reserved	0x0
1:0	R/W	- Reserved	0x0

Table 122: CLK_CTRL_REG (0x5000000A)

Bit	Mode	Symbol/Description	Reset
7	R	RUNNING_AT_XTAL32M Indicates that the XTAL32M clock is used as clock, and may not be switched off	0x0
6	R	RUNNING_AT_RC32M Indicates that the RC32M clock is used as clock	0x1
5	R	RUNNING_AT_LP_CLK Indicates that either the LP_CLK is being used as system clock	0x0
4:3	R/W	LP_CLK_SEL Sets the clock source of the LowerPower clock 0x0: RC32K 0x1: RCX 0x2: XTAL32K through the oscillator with an external Crystal. 0x3: XTAL32K through an external square wave generator (set PID of P0[3] to FUNC_GPIO)	0x0

Bit	Mode	Symbol/Description	Reset
		Change this setting before using this clock, and while RUNNING_AT_LP_CLK == 0.	
2	R/W	XTAL32M_DISABLE Setting this bit instantaneously disables the 32-MHz crystal oscillator. Also, after sleep/wake-up cycle, the oscillator will not be enabled. This bit may not be set to 1 when RUNNING_AT_XTAL32M is 1 to prevent deadlock. After resetting this bit, wait for XTAL32M_SETTLED or XTAL32M_TRIM_READY to become 1 before switching to XTAL32M clock source.	0x0
1:0	R/W	SYS_CLK_SEL Selects the clock source. 0x0: XTAL32M (check the XTAL32M_SETTLED and XTAL32M_TRIM_READY bits) 0x1: RC32M 0x2/0x3: LP_CLK	0x1

Table 123: PMU_CTRL_REG (0x50000010)

Bit	Mode	Symbol/Description	Reset
6	R/W	MAP_BANDGAP_EN Enable the wake-up diagnostics mapping. When set, these functions are mapped (set direction to output) P0[2]: BANDGAP_ENABLE P0[1]: Power WOKENUP Note: P0[2] assigned also to SWD_CLK, thus the debugger must be detached before entering into Sleep mode with MAP_BANDGAP_EN = 1. Also, see SYS_STAT_REG->DBG_IS_UP.	0x0
5:4	R/W	OTP_COPY_DIV Sets the HCLK division during OTP mirroring.	0x0
3	R/W	- Reserved	0x0
2	R/W	RADIO_SLEEP Put the digital part of the radio in powerdown	0x1
1	R/W	TIM_SLEEP Put PD_TIM in powerdown	0x1
0	R/W	RESET_ON_WAKEUP Perform a Hardware Reset after waking up. Booter will be started.	0x0

Table 124: SYS_CTRL_REG (0x50000012)

Bit	Mode	Symbol/Description	Reset
15	W	SW_RESET Writing 1 to this bit resets the device, except for: SYS_CTRL_REG CLK_FREQ_TRIM_REG ...	0x0
10	R/W	TIMEOUT_DISABLE	0x0

Bit	Mode	Symbol/Description	Reset
		Disables timeout in Power statemachine. By default, the statemachine continues if after 2 ms the blocks are not started up. This can be read back from ANA_STATUS_REG.	
9	R/W	- Reserved	0x0
8:7	R/W	DEBUGGER_ENABLE Enable the debugger. This bit is set by the booter according to the OTP header. If not set, the SWDIO and SW_CLK can be used as GPIO ports. 0x0: no debugger enabled. 0x1: SW_CLK = P0[2], SW_DIO=P0[5] 0x2: SW_CLK = P0[2], SW_DIO=P0[1] 0x3: SW_CLK = P0[2], SW_DIO=P0[10]	0x0
6	R/W	OTPC_RESET_REQ Reset request for the OTP controller.	0x0
5	R/W	- Reserved	0x1
4	R/W	OTP_COPY Enables OTP to SysRAM copy action after waking up PD_SYS.	0x0
3	R/W	- Reserved	0x0
2	R/W	DEV_PHASE Sets the development phase mode. If this bit is set, in combination with the OTP_COPY bit, the OTP DMA emulates the OTP mirroring to System RAM. No actual writing to RAM is done, but the exact same amount of time is spent as if the mirroring would take place. This is to mimic the behavior as if the System Code is already in OTP, and the mirroring takes place after waking up, but the (development) code still resides in an external source. If this bit is set to 0 and OTP_COPY = 1, then the OTP DMA actually does the OTP mirroring at wake-up.	0x0
1:0	R/W	REMAP_ADR0 Controls which memory is located at address 0x0000 for execution. 0x0: ROM 0x1: OTP 0x2: RAM (SysRAM1) 0x3: RAM (SysRAM2, 32 kB offset) This bitfield only takes affect after software reset.	0x0

Table 125: SYS_STAT_REG (0x50000014)

Bit	Mode	Symbol/Description	Reset
7	R	XTAL32M_SETTLED Indicates that XTAL32M has had its settle time, as defined by TRIM_CTRL_REG[XTAL_SETTLE_N].	0x0
6	R	XTAL32M_TRIM_READY Indicates that XTAL trimming mechanism is ready, which means the trimming equals CLK_FREQ_TRIM_REG.	0x1

Bit	Mode	Symbol/Description	Reset
5	R	- Reserved	0x0
4	R	DBG_IS_UP Indicates that the software debugger is attached and in connection with the Cortex.	0x0
3	R	TIM_IS_UP Indicates that PD_TIM is functional.	0x0
2	R	TIM_IS_DOWN Indicates that PD_TIM is in powerdown.	0x1
1	R	RAD_IS_UP Indicates that PD_RAD is functional.	0x0
0	R	RAD_IS_DOWN Indicates that PD_RAD is in powerdown.	0x1

Table 126: TRIM_CTRL_REG (0x50000016)

Bit	Mode	Symbol/Description	Reset
13:8	R/W	XTAL_SETTLE_N Designates that the XTAL can be safely used as the CPU clock. When XTAL_CLK_CNT reases this value, the signal XTAL32M_SETTLED bit in the SYS_STAT_REG will be set. Counts in steps of 64 XTAL clock-cycles.	0x3F
7:6	R/W	XTAL_TRIM_SELECT Select which source controls the XTAL trimming 0b00: XTAL counter. Starts XTAL32M_START_REG[XTAL32M_START] after COUNT_N * 32 xtal pulses trim is changed to CLK_FREQ_TRIM_REG[XTAL32M_TRIM]. 0b01: XTAL OK filter. Starts with CLK_FREQ_TRIM_REG[XTAL32M_START], when XTAL amplitude is ramping is changed to CLK_FREQ_TRIM_REG[XTAL32M_TRIM]. 0b10: statically forced off. Only uses CLK_FREQ_TRIM_REG[XTAL32M_TRIM]. 0b11: XTAL OK filter, 2 stage. Starts with CLK_FREQ_TRIM_REG[XTAL32M_START] switches to CLK_FREQ_TRIM_REG[XTAL32M_RAMP] after timeout (32 μs), and switches to CLK_FREQ_TRIM_REG[XTAL32M_TRIM] when XTAL amplitude is ramping up.	0x0
5:0	R/W	XTAL_COUNT_N Defines the number of XTAL cycles to be counted, before the XTAL trimming is applied, in steps of 64 cycles. 0x01: 64 0x02: 128 0x3f: 4032	0x22

Table 127: RAM_PWR_CTRL_REG (0x50000018)

Bit	Mode	Symbol/Description	Reset
5:4	R/W	- Reserved	0x0
3:2	R/W	RAM2_PWR_CTRL See description of RAM1_PWR_CTRL.	0x0

Bit	Mode	Symbol/Description	Reset
1:0	R/W	RAM1_PWR_CTRL Power state control of the individual RAMs. May only change when the memory is not accessed. When in Active or Sleep mode: 0x0: Normal operation 0x1: Normal operation 0x2: Retained (no access possible) 0x3: Off (memory content corrupted) When in Extended Sleep, Deep Sleep or Hibernation mode 0x0: Retained 0x1: Off (memory content corrupted) 0x2: Retained 0x3: Off (memory content corrupted)	0x0

Table 128: CLK_RC32K_REG (0x50000020)

Bit	Mode	Symbol/Description	Reset
4:1	R/W	RC32K_TRIM 0000 = Lowest frequency 0111 = Default 1111 = Highest frequency	0x7
0	R/W	RC32K_DISABLE Instantly disables the 32-kHz RC oscillator. Sleep cycles cannot happen with this clock disabled.	0x0

Table 129: CLK_XTAL32K_REG (0x50000022)

Bit	Mode	Symbol/Description	Reset
8	R/W	- Reserved	0x0
7	R/W	XTAL32K_DISABLE_AMPREG Setting this bit disables the amplitude regulation of the XTAL32kHz oscillator. Set this bit to 1 for an external clock to XTAL32Kp. Keep this bit 0 with a crystal between XTAL32Kp and XTAL32Km.	0x0
6:3	R/W	XTAL32K_CUR Bias current for the 32-kHz XTAL oscillator. 0000 is minimum, 1111 is maximum, 0011 is default. For each application, there is an optimal setting for which the start-up behavior is optimal.	0x5
2:1	R/W	XTAL32K_RBIAS Setting for the bias resistor. 00 is maximum, 11 is minimum. Prefered setting will be provided by Renesas Electronics.	0x3
0	R/W	XTAL32K_ENABLE Enables the 32-kHz XTAL oscillator. Also, set GP_DATA_REG[P03_P04_FILT_DIS] = 1 for lowest current consumption.	0x0

Table 130: CLK_RC32M_REG (0x50000024)

Bit	Mode	Symbol/Description	Reset
10:7	R/W	RC32M_COSC C-adjust of RC-oscillator. A higher value of COSC results in a lower frequency.	0xF
6:5	R/W	RC32M_RANGE Coarse adjust. A higher value of RANGE results in a higher frequency, values 2 and 3 are equal.	0x0
4:1	R/W	RC32M_BIAS Bias adjustment.	0x8
0	R/W	RC32M_DISABLE Instantly disables the 32-MHz RC oscillator. Disabling of the oscillator during sleep happens automatically.	0x0

Table 131: CLK_RCX_REG (0x50000026)

Bit	Mode	Symbol/Description	Reset
11:8	R/W	RCX_BIAS LDO bias current. 0x0: Minimum 0xF: Maximum	0xA
7	R/W	RCX_C0 Add unit capacitance to RC-time delay.	0x1
6:2	R/W	RCX_CADJUST Adjust capacitance part of RC-time delay. 0x00: Minimum capacitance 0x1F: Maximum capacitance	0x1F
1	R/W	RCX_RADJUST Adjust resistance part of RC-time delay. Lower resistance increases power consumption. 0x0: Maximum resistance 0x1: Minimum resistance	0x0
0	R/W	RCX_ENABLE Enable the RCX oscillator.	0x0

Table 132: BANDGAP_REG (0x50000028)

Bit	Mode	Symbol/Description	Reset
9:5	R/W	BGR_ITRIM Trim setting for bandgap bias current 10000 -> -25% .... 11111 -> ~0% 00000 -> ~0% (typ) ... 01111 -> +32%	0x0

Bit	Mode	Symbol/Description	Reset
4:0	R/W	BGR_TRIM Trim setting for bandgap voltage 10000 -> -6.4% .... 11111 -> ~0% 00000 -> ~0% (typ) ... 01111 -> +5.8%	0x0

Table 133: ANA_STATUS_REG (0x5000002A)

Bit	Mode	Symbol/Description	Reset
13	R	DCDC_SLP_INTERVAL_STAT Indicates that there were more than 16 vdcdc sampling cycles (SLP) where the voltage was OK.	0x0
12	R	CLKLESS_WAKEUP_STAT Indicates the output of the Clockless wake-up XOR tree. If this signal is 0, the chip will wake up. Use the HIBERN_WKUP_POLARITY bit to set the value to 1 before going into Hibernation mode.	0x0
11	R	- Reserved	0x0
10	R	LDO_GPADC_OK Indicates that LDO_GPADC output is OK	0x0
9	R	LDO_XTAL_OK Indicates that LDO_XTAL output is OK	0x0
8	R	BOOST_SELECTED 0: Buck mode detected 1: Boost mode detected	0x0
7	R	POR_VBAT_HIGH Output of V_{BAT_HIGH} supply rail voltage monitoring circuit. 0: Voltage level on V_{BAT_HIGH} is lower than POR VBAT_HIGH threshold V_{TH_L} (rail not ok, will result in reset if not masked) 1: Voltage level on V_{BAT_HIGH} is higher than POR VBAT_HIGH threshold V_{TH_H} (rail ok, reset released)	0x0
6	R	POR_VBAT_LOW Output of V_{BAT_LOW} supply rail voltage monitoring circuit. 0: Voltage level on V_{BAT_LOW} is lower than POR VBAT_LOW threshold V_{TH_L} (rail not ok, will result in reset if not masked) 1: Voltage level on V_{BAT_LOW} is higher than POR VBAT_LOW threshold V_{TH_H} (rail ok, reset released)	0x0
5	R	BANDGAP_OK Indicates that BANDGAP is OK	0x0
4	R	COMP_VBAT_HIGH_NOK Indicates that $V_{BAT\_HIGH} < V_{BAT\_LOW}$ -50 mV	0x0
3	R	COMP_VBAT_HIGH_OK	0x0

Bit	Mode	Symbol/Description	Reset
		Indicates that V_{BAT_HIGH} > V_{BAT_LOW} +50 mV	
2	R	DCDC_OK Indicates that V_{BAT_LOW} (Buck mode) or V_{BAT_HIGH} (Boost mode) is OK	0x0
1	R	LDO_LOW_OK Indicates that LDO_LOW output is OK (only valid for high current mode)	0x0
0	R	LDO_CORE_OK Indicates that LDO_CORE output is OK	0x0

Table 134: XTAL32M_START_REG (0x50000030)

Bit	Mode	Symbol/Description	Reset
15:8	R/W	XTAL32M_RAMP XTAL frequency trimming register. 0x00: Highest frequency 0xFF: Lowest frequency	0x0
7:0	R/W	XTAL32M_START XTAL frequency trimming register. 0x0: Highest frequency 0xF: Lowest frequency	0x32

Table 135: XTAL32M_TRSTAT_REG (0x50000032)

Bit	Mode	Symbol/Description	Reset
7:0	R	XTAL32M_TRSTAT Reads value of the current XTAL trimming	0x0

Table 136: XTALRDY_CTRL_REG (0x50000034)

Bit	Mode	Symbol/Description	Reset
7:0	R/W	XTALRDY_CNT Number of 32 kHz cycles between the crystal is enabled, and the XTALRDY_IRQ is fired. 0x00: no interrupt	0x0

Table 137: XTAL32M_CTRL0_REG (0x50000038)

Bit	Mode	Symbol/Description	Reset
9	R/W	- Reserved	0x0
8	R/W	CMP_BIAS_LVL Comparator bias current setting: 0: 3 µA 1: 1 µA	0x0
7:5	R/W	CORE_AMPL_TRIM Core amplitude trimming	0x3

Bit	Mode	Symbol/Description	Reset
4:2	R/W	CORE_CUR_SET Core current trim setting	0x2
1	R/W	CORE_AMPL_REG_NULLBIAS Keep bias in ampl detector alive, even when there is a large drive	0x0
0	R/W	DCBLOCK_ENABLE Enable dcblock/high pass filter circuit	0x1

Table 138: POR_PIN_REG (0x50000040)

Bit	Mode	Symbol/Description	Reset
7	R/W	POR_PIN_POLARITY 0: Active Low 1: Active High Note: This applies only for the GPIO pin. Reset pad has a fixed polarity	0x0
6:4	R/W	- Reserved	0x0
3:0	R/W	POR_PIN_SELECT Selects the GPIO which is used for POR generation. 0x0: GPIO pin POReset disabled 0x1: P0_0 0x2: P0_1 ... 0xB: P0_10 0xC: P0_11 0xD - 0xF: reserved	0x0

Table 139: POR_TIMER_REG (0x50000042)

Bit	Mode	Symbol/Description	Reset
6:0	R/W	POR_TIME Time for the POReset to happen. Formula: Time = POR_TIME x 4096 x RC32k clock period Default value: ~3 seconds When set to 0x00, the POR TIMER is disabled.	0x18

Table 140: PMU_SLEEP_REG (0x50000050)

Bit	Mode	Symbol/Description	Reset
11:0	R/W	BG_REFRESH_INTERVAL Defines the refresh interval of reference voltages (bandgap activation and sampling), in units of 2 ms.	0x80

Table 141: POWER_CTRL_REG (0x50000052)

Bit	Mode	Symbol/Description	Reset
15	R/W	VBAT_HL_CONNECT_MODE Sets the control mode fo the switch between VBAT_HIGH and VBAT_LOW 0: Manual (default) 1: Automatic (Boost mode only)	0x0
14	R/W	POR_VBAT_HIGH_HYST_DIS 0: Hysteresis enabled 1: Hysteresis disabled	0x1
13	R/W	POR_VBAT_HIGH_HYST_SEL 0: Low level selected 1: High level selected	0x0
12	R/W	POR_VBAT_HIGH_DISABLE Disable por_vbat_high circuit	0x0
11	R/W	POR_VBAT_LOW_HYST_DIS 0: Hysteresis enabled 1: Hysteresis disabled	0x0
10	R/W	POR_VBAT_LOW_HYST_SEL 0: Low level selected 1: High level selected	0x0
9	R/W	POR_VBAT_LOW_DISABLE Disable por_vbat_low circuit	0x0
8	R/W	CP_DISABLE Disables LDO_CORE charge-pump circuit	0x0
7	R/W	LDO_VREF_HOLD_FORCE Forces LDO references in HOLD mode	0x0
6:5	R/W	LDO_LOW_CTRL_REG 00: High-current mode in active, LDO_LOW OFF in sleep 01: LDO_LOW OFF 10: Low-current mode in active, Low-current mode in sleep 11: High-current mode in active, Low-current mode in sleep	0x0
4	R/W	LDO_CORE_DISABLE Disables LDO_CORE	0x0
3	R/W	LDO_CORE_RET_ENABLE LDO_CORE_RETENTION 0: Disabled 1: Enabled	0x0
2	R/W	VBAT_HL_CONNECT Switch between V_{BAT_HIGH} and V_{BAT_LOW} 0: Open 1: Closed	0x0
1	R/W	CMP_VBAT_HIGH_OK_ENABLE Enable cmp_vbat_high_ok	0x0
0	R/W	CMP_VBAT_HIGH_NOK_ENABLE	0x0

Bit	Mode	Symbol/Description	Reset
		Enable cmp_vbat_high_nok	

Table 142: POWER_LEVEL_REG (0x50000054)

Bit	Mode	Symbol/Description	Reset
13:11	R/W	DCDC_TRIM Delta from DCDC_LEVEL nominal value 000: -75 mV 001: -50 mV 010: -25 mV 011: 0 (default) 100: +25 mV 101: +50 mV 110: +75 mV 111: +100 mV	0x5
10:9	R/W	DCDC_LEVEL 00: 1.1 V when DCDC_LEVEL1V1_BUMP = 0, 1.2 V when DCDC_LEVEL1V1_BUMP = 1 01: 1.8 V (default) 10: 2.5 V 11: 3.0 V	0x1
8	R/W	- Reserved	0x0
7	R/W	- Reserved	0x0
6:4	R/W	LDO_XTAL_TRIM Delta from 0.9 V nominal value 000: -75 mV 001: -50 mV 010: -25 mV 011: 0 (default) 100: +25 mV 101: +50 mV 110: +75 mV 111: +100 mV	0x3
3:1	R/W	LDO_LOW_TRIM Delta from 1.2 V nominal value 000: -75 mV 001: -50 mV 010: -25 mV 011: 0 (default) 100: +25 mV 101: +50 mV 110: +75 mV 111: +100 mV (coldboot)	0x7
0	R/W	-	0x0

Bit	Mode	Symbol/Description	Reset
		Reserved	

Table 143: DCDC_SLP_CTRL_REG (0x50000056)

Bit	Mode	Symbol/Description	Reset
7:5	R/W	DCDC_SLP_OFFSET DCDC offset voltage in sleep 0: 0 mV 1: 25 mV 2: 50 mV 3: 75 mV 4: 100 mV (default) 5: 150 mV 6: 200 mV 7: 250 mV	0x4
4	R/W	DCDC_SLP_UPDATE_INTERVAL Update sampling interval dynamically	0x1
3:1	R/W	DCDC_SLP_INTERVAL DCDC sampling interval in sleep. DCDC_SLP_UPDATE_INTERVAL needs to be 0x0 for this setting to take effect. 0: 62.5 µs 1: 125 µs 2: 250 µs 3: 500 µs 4: 1 ms 5: 2 ms 6: 4 ms 7: 8 ms	0x0
0	R/W	DCDC_SLP_ENABLE Enable DCDC in sleep	0x0

Table 144: LDO_CORE_LEVEL_REG (0x50000058)

Bit	Mode	Symbol/Description	Reset
2:0	R/W	LDO_CORE_LEVEL 0: 900 mV 1: 910 mV 2: 920 mV 3: 930 mV 4: 940 mV 5: 960 mV 6: 980 mV 7: 1 V	0x0

31.4 DCDC Converter Registers

Table 145: Register map DCDC

Address	Register	Description
0x50000080	DCDC_CTRL_REG	
0x50000082	DCDC_CTRL1_REG	

Table 146: DCDC_CTRL_REG (0x50000080)

Bit	Mode	Symbol/Description	Reset
15:12	R/W	DCDC_ILIM_MAX Maximum value for automatic inductor peak current limit control. 0x0: 6 mA 0x1: 12 mA 0x2: 18 mA 0x3: 24 mA 0x4: 30 mA 0x5: 36 mA 0x6: 42 mA 0x7: 48 mA 0x8: 54 mA (default, limits inrush current) 0x9: 60 mA 0xA: 66 mA 0xB: 72 mA 0xC: 78 mA 0xD: 84 mA 0xE: 90 mA 0xF: 96 mA (set as default for low-ohmic batteries)	0x8
11:8	R/W	DCDC_ILIM_MIN Minimum value for automatic inductor peak current limit control. 0x0: 6 mA 0x1: 12 mA 0x2: 18 mA 0x3: 24 mA 0x4: 30 mA (default) 0x5: 36 mA 0x6: 42 mA 0x7: 48 mA 0x8: 54 mA 0x9: 60 mA 0xA: 66 mA 0xB: 72 mA 0xC: 78 mA 0xD: 84 mA 0xE: 90 mA 0xF: 96 mA	0x4
7:6	R/W	DCDC_OK_CLR_CNT Number of subsequent V_NOK events needed to reset VDCD_OK.	0x2

Bit	Mode	Symbol/Description	Reset
		0x0: 2 0x1: 4 0x2: 8 (default) 0x3: 15	
5:3	R/W	DCDC_TIMEOUT Switch timeout, go to next state if either switch is active for longer than this setting. 0x0: Disabled 0x1: 0.25 µs 0x2: 0.50 µs 0x3: 0.75 µs 0x4: 1.00 µs (default) 0x5: 1.25 µs 0x6: 1.50 µs 0x7: 1.75 µs	0x4
2:1	R/W	DCDC_CLK_DIV Idle clock divider, sets rate at which the output is monitored when the converter is idle. 0x0: Divide by 4 0x1: Divide by 8 0x2: Divide by 16 0x3: Divide by 32	0x1
0	R/W	DCDC_ENABLE Enables hardware control of the DCDC converter. 0: DCDC converter disabled 1: DCDC converter under hardware control	0x0

Table 147: DCDC_CTRL1_REG (0x50000082)

Bit	Mode	Symbol/Description	Reset
5	R/W	DCDC_LEVEL1V1_BUMP Increases the DCDC output voltage in Buck mode (DCDC_LEVEL = 00) from 1.1 V to 1.2 V.	0x1
4	R/W	DCDC_FIX_ILIM_SLP Sets a fixed inductor current limit in sleep to maximize amount of charge added per cycle and minimize the duty-cycle. 0x0: Automatic current limit setting remains active in Sleep mode 0x1: Current limit fixed in Sleep mode, value set with field DCDC_ILIM_SLP	0x1
3:0	R/W	DCDC_ILIM_SLP Maximum value for the fixed inductor current in Sleep mode: 0x0: 6 mA 0x1: 12 mA 0x2: 18 mA 0x3: 24 mA 0x4: 30 mA 0x5: 36 mA 0x6: 42 mA 0x7: 48 mA	0xF

Bit	Mode	Symbol/Description	Reset
		0x8: 54 mA 0x9: 60 mA 0xA: 66 mA 0xB: 72 mA 0xC: 78 mA 0xD: 84 mA 0xE: 90 mA 0xF: 96 mA (default, lowest duty-cycle)	

31.5 DMA Controller Registers

Table 148: Register map DMA

Address	Register	Description
0x50003600	DMA0_A_STARTL_REG	Start address Low A of DMA channel 0
0x50003602	DMA0_A_STARTH_REG	Start address High A of DMA channel 0
0x50003604	DMA0_B_STARTL_REG	Start address Low B of DMA channel 0
0x50003606	DMA0_B_STARTH_REG	Start address High B of DMA channel 0
0x50003608	DMA0_INT_REG	DMA receive interrupt register channel 0
0x5000360a	DMA0_LEN_REG	DMA receive length register channel 0
0x5000360c	DMA0_CTRL_REG	Control register for the DMA channel 0
0x5000360e	DMA0_IDX_REG	Index value of DMA channel 0
0x50003610	DMA1_A_STARTL_REG	Start address Low A of DMA channel 1
0x50003612	DMA1_A_STARTH_REG	Start address High A of DMA channel 1
0x50003614	DMA1_B_STARTL_REG	Start address Low B of DMA channel 1
0x50003616	DMA1_B_STARTH_REG	Start address High B of DMA channel 1
0x50003618	DMA1_INT_REG	DMA receive interrupt register channel 1
0x5000361a	DMA1_LEN_REG	DMA receive length register channel 1
0x5000361c	DMA1_CTRL_REG	Control register for the DMA channel 1
0x5000361e	DMA1_IDX_REG	Index value of DMA channel 1
0x50003620	DMA2_A_STARTL_REG	Start address Low A of DMA channel 2
0x50003622	DMA2_A_STARTH_REG	Start address High A of DMA channel 2
0x50003624	DMA2_B_STARTL_REG	Start address Low B of DMA channel 2
0x50003626	DMA2_B_STARTH_REG	Start address High B of DMA channel 2
0x50003628	DMA2_INT_REG	DMA receive interrupt register channel 2
0x5000362a	DMA2_LEN_REG	DMA receive length register channel 2
0x5000362c	DMA2_CTRL_REG	Control register for the DMA channel 2
0x5000362e	DMA2_IDX_REG	Index value of DMA channel 2
0x50003630	DMA3_A_STARTL_REG	Start address Low A of DMA channel 3
0x50003632	DMA3_A_STARTH_REG	Start address High A of DMA channel 3
0x50003634	DMA3_B_STARTL_REG	Start address Low B of DMA channel 3
0x50003636	DMA3_B_STARTH_REG	Start address High B of DMA channel 3
0x50003638	DMA3_INT_REG	DMA receive interrupt register channel 3
0x5000363a	DMA3_LEN_REG	DMA receive length register channel 3
0x5000363c	DMA3_CTRL_REG	Control register for the DMA channel 3
0x5000363e	DMA3_IDX_REG	Index value of DMA channel 3
0x50003680	DMA_REQ_MUX_REG	DMA channel assignments
0x50003682	DMA_INT_STATUS_REG	DMA interrupt status register
0x50003684	DMA_CLEAR_INT_REG	DMA clear interrupt register

Table 149: DMA0_A_STARTL_REG (0x50003600)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA0_A_STARTL Source start address, lower 16 bits	0x0

Table 150: DMA0_A_STARTH_REG (0x50003602)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA0_A_STARTH Source start address, upper 16 bits	0x0

Table 151: DMA0_B_STARTL_REG (0x50003604)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA0_B_STARTL Destination start address, lower 16 bits	0x0

Table 152: DMA0_B_STARTH_REG (0x50003606)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA0_B_STARTH Destination start address, upper 16 bits	0x0

Table 153: DMA0_INT_REG (0x50003608)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA0_INT Number of transfers until an interrupt is generated. The interrupt is generated after a transfer, if DMAx_INT_REG is equal to DMAx_IDX_REG and before DMAx_IDX_REG is incremented. The bit field IRQ_ENABLE of DMAx_CTRL_REG must be set to 1 to let the controller generate the interrupt.	0x0

Table 154: DMA0_LEN_REG (0x5000360A)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA0_LEN DMA channel's transfer length. DMAx_LEN of value 0, 1, 2, ... results into an actual transfer length of 1, 2, 3, ...	0x0

Table 155: DMA0_CTRL_REG (0x5000360C)

Bit	Mode	Symbol/Description	Reset
15:14	R	- Reserved	0x0
13	R/W	REQ_SENSE 0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests	0x0
12	R/W	DMA_INIT	0x0

Bit	Mode	Symbol/Description	Reset
		0 = DMA performs copy A1 to B1, A2 to B2, and so on ... 1 = DMA performs copy of A1 to B1, B2, and so on ... This feature is useful for memory initialization to any value. Thus, BINC must be set to 1, while AINC is don't care, as only one fetch from A is done. This process cannot be interrupted by other DMA channels. It is also noted that DMA_INIT should not be used when DREQ_MODE = 1.	
11	R/W	DMA_IDLE 0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority. 1 = Interrupting mode, the DMA inserts a wait cycle after each store allowing the CPU to steal cycles or cache to perform a burst read. If DREQ_MODE = 1, DMA_IDLE is don't care.	0x0
10:8	R/W	DMA_PRIO The priority level determines which DMA channel is granted access for transferring data, in case more than one channel is active and request the bus at the same time. The greater the value, the higher the priority. In specific: 000 = lowest priority 111 = highest priority If different channels with equal priority level values request the bus at the same time, an inherent priority mechanism is applied. According to this mechanism, if, for example, both the DMA0 and DMA1 channels have the same priority level, then DMA0 is first granted access to the bus.	0x0
7	R/W	CIRCULAR 0 = Normal mode. The DMA channel stops after having completed the transfer of length determined by DMAx_LEN_REG. DMA_ON automatically deasserts when the transfer is completed. 1 = Circular mode (applicable only if DREQ_MODE = 1). In this mode, DMA_ON never deasserts, as the DMA channel automatically resets DMAx_IDX_REG and starts a new transfer.	0x0
6	R/W	AINC Enable increment of source address. 0 = Do not increment (source address stays the same during the transfer) 1 = Increment according to the value of BW bit field (by 1, when BW = 00; by 2, when BW = 01; by 4, when BW = 10)	0x0
5	R/W	BINC Enable increment of destination address. 0 = Do not increment (destination address stays the same during the transfer) 1 = Increment according to the value of BW bit field (by 1, when BW = 00; by 2, when BW = 01; by 4, when BW = 10)	0x0
4	R/W	DREQ_MODE 0 = DMA channel starts immediately 1 = DMA channel must be triggered by peripheral DMA request (see also the description of DMA_REQ_MUX_REG)	0x0
3	R/W	IRQ_ENABLE 0 = Disable interrupt on this channel 1 = Enable interrupt on this channel	0x0
2:1	R/W	BW Bus transfer width: 00 = 1 Byte (suggested for peripherals like UART and 8-bit SPI)	0x0

Bit	Mode	Symbol/Description	Reset
		01 = 2 Bytes (suggested for peripherals like I2C and 16-bit SPI) 10 = 4 Bytes (suggested for Memory-to-Memory transfers) 11 = Reserved	
0	R/W	DMA_ON 0 = DMA channel is off, clocks are disabled 1 = DMA channel is enabled. This bit is automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set.	0x0

Table 156: DMA0_IDX_REG (0x5000360E)

Bit	Mode	Symbol/Description	Reset
15:0	R	DMA0_IDX This (read-only) register determines the data items currently fetched by the DMA channel, during an on-going transfer. When the transfer is completed, the register is automatically reset to 0. The DMA channel uses this register to form the source/destination address of the next DMA cycle, considering also AINC/BINC and BW.	0x0

Table 157: DMA1_A_STARTL_REG (0x50003610)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA1_A_STARTL Source start address, lower 16 bits	0x0

Table 158: DMA1_A_STARTH_REG (0x50003612)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA1_A_STARTH Source start address, upper 16 bits	0x0

Table 159: DMA1_B_STARTL_REG (0x50003614)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA1_B_STARTL Destination start address, lower 16 bits	0x0

Table 160: DMA1_B_STARTH_REG (0x50003616)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA1_B_STARTH Destination start address, upper 16 bits	0x0

Table 161: DMA1_INT_REG (0x50003618)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA1_INT Number of transfers until an interrupt is generated. The interrupt is generated after a transfer, if DMAx_INT_REG is equal to DMAx_IDX_REG and before	0x0

Bit	Mode	Symbol/Description	Reset
		DMAx_IDX_REG is incremented. The bit field IRQ_ENABLE of DMAx_CTRL_REG must be set to 1 to let the controller generate the interrupt.	

Table 162: DMA1_LEN_REG (0x5000361A)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA1_LEN DMA channel's transfer length. DMAx_LEN of value 0, 1, 2, ... results into an actual transfer length of 1, 2, 3, ...	0x0

Table 163: DMA1_CTRL_REG (0x5000361C)

Bit	Mode	Symbol/Description	Reset
15:14	R	- Reserved	0x0
13	R/W	REQ_SENSE 0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests	0x0
12	R/W	DMA_INIT 0 = DMA performs copy A1 to B1, A2 to B2, and so on ... 1 = DMA performs copy of A1 to B1, B2, and so on ... This feature is useful for memory initialization to any value. Thus, BINC must be set to 1, while AINC is don't care, as only one fetch from A is done. This process cannot be interrupted by other DMA channels. It is also noted that DMA_INIT should not be used when DREQ_MODE = 1.	0x0
11	R/W	DMA_IDLE 0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority. 1 = Interrupting mode, the DMA inserts a wait cycle after each store allowing the CPU to steal cycles or cache to perform a burst read. If DREQ_MODE= 1, DMA_IDLE is don't care.	0x0
10:8	R/W	DMA_PRIO The priority level determines which DMA channel is granted access for transferring data, in case more than one channel is active and request the bus at the same time. The greater the value, the higher the priority. In specific: 000 = lowest priority 111 = highest priority If different channels with equal priority level values request the bus at the same time, an inherent priority mechanism is applied. According to this mechanism, if, for example, both the DMA0 and DMA1 channels have the same priority level, then DMA0 is first granted access to the bus.	0x0
7	R/W	CIRCULAR 0 = Normal mode. The DMA channel stops after having completed the transfer of length determined by DMAx_LEN_REG. DMA_ON automatically deasserts when the transfer is completed. 1 = Circular mode (applicable only if DREQ_MODE = 1). In this mode, DMA_ON never deasserts, as the DMA channel automatically resets DMAx_IDX_REG and starts a new transfer.	0x0
6	R/W	AINC	0x0

Bit	Mode	Symbol/Description	Reset
		Enable increment of source address. 0 = Do not increment (source address stays the same during the transfer) 1 = Increment according to the value of BW bit field (by 1, when BW = 00; by 2, when BW = 01; by 4, when BW = 10)	
5	R/W	BINC Enable increment of destination address. 0 = Do not increment (destination address stays the same during the transfer) 1 = Increment according to the value of BW bit field (by 1, when BW = 00; by 2, when BW = 01; by 4, when BW = 10)	0x0
4	R/W	DREQ_MODE 0 = DMA channel starts immediately 1 = DMA channel must be triggered by peripheral DMA request (see also the description of DMA_REQ_MUX_REG)	0x0
3	R/W	IRQ_ENABLE 0 = disable interrupt on this channel 1 = enable interrupt on this channel	0x0
2:1	R/W	BW Bus transfer width: 00 = 1 Byte (suggested for peripherals like UART and 8-bit SPI) 01 = 2 Bytes (suggested for peripherals like I2C and 16-bit SPI) 10 = 4 Bytes (suggested for Memory-to-Memory transfers) 11 = Reserved	0x0
0	R/W	DMA_ON 0 = DMA channel is off, clocks are disabled 1 = DMA channel is enabled. This bit is automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set.	0x0

Table 164: DMA1_IDX_REG (0x5000361E)

Bit	Mode	Symbol/Description	Reset
15:0	R	DMA1_IDX This (read-only) register determines the data items currently fetched by the DMA channel, during an on-going transfer. When the transfer is completed, the register is automatically reset to 0. The DMA channel uses this register to form the source/destination address of the next DMA cycle, considering also AINC/BINC and BW.	0x0

Table 165: DMA2_A_STARTL_REG (0x50003620)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA2_A_STARTL Source start address, lower 16 bits	0x0

Table 166: DMA2_A_STARTH_REG (0x50003622)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA2_A_STARTH	0x0

Bit	Mode	Symbol/Description	Reset
		Source start address, upper 16 bits	

Table 167: DMA2_B_STARTL_REG (0x50003624)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA2_B_STARTL Destination start address, lower 16 bits	0x0

Table 168: DMA2_B_STARTH_REG (0x50003626)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA2_B_STARTH Destination start address, upper 16 bits	0x0

Table 169: DMA2_INT_REG (0x50003628)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA2_INT Number of transfers until an interrupt is generated. The interrupt is generated after a transfer, if DMAx_INT_REG is equal to DMAx_IDX_REG and before DMAx_IDX_REG is incremented. The bit field IRQ_ENABLE of DMAx_CTRL_REG must be set to 1 to let the controller generate the interrupt.	0x0

Table 170: DMA2_LEN_REG (0x5000362A)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA2_LEN DMA channel's transfer length. DMAx_LEN of value 0, 1, 2, ... results into an actual transfer length of 1, 2, 3, ...	0x0

Table 171: DMA2_CTRL_REG (0x5000362C)

Bit	Mode	Symbol/Description	Reset
15:14	R	- Reserved	0x0
13	R/W	REQ_SENSE 0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests	0x0
12	R/W	DMA_INIT 0 = DMA performs copy A1 to B1, A2 to B2, and so on ... 1 = DMA performs copy of A1 to B1, B2, and so on ... This feature is useful for memory initialization to any value. Thus, BINC must be set to 1, while AINC is don't care, as only one fetch from A is done. This process cannot be interrupted by other DMA channels. It is also noted that DMA_INIT should not be used when DREQ_MODE = 1.	0x0
11	R/W	DMA_IDLE 0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority.	0x0

Bit	Mode	Symbol/Description	Reset
		1 = Interrupting mode, the DMA inserts a wait cycle after each store allowing the CPU to steal cycles or cache to perform a burst read. If DREQ_MODE = 1, DMA_IDLE is don't care.	
10:8	R/W	DMA_PRIO The priority level determines which DMA channel is granted access for transferring data, in case more than one channel is active and request the bus at the same time. The greater the value, the higher the priority. In specific: 000 = lowest priority 111 = highest priority If different channels with equal priority level values request the bus at the same time, an inherent priority mechanism is applied. According to this mechanism, if, for example, both the DMA0 and DMA1 channels have the same priority level, then DMA0 is first granted access to the bus.	0x0
7	R/W	CIRCULAR 0 = Normal mode. The DMA channel stops after having completed the transfer of length determined by DMAx_LEN_REG. DMA_ON automatically deasserts when the transfer is completed. 1 = Circular mode (applicable only if DREQ_MODE = 1). In this mode, DMA_ON never deasserts, as the DMA channel automatically resets DMAx_IDX_REG and starts a new transfer.	0x0
6	R/W	AINC Enable increment of destination address. 0 = Do not increment (destination address stays the same during the transfer) 1 = Increment according to the value of BW bit field (by 1, when BW = 00; by 2, when BW = 01; by 4, when BW = 10)	0x0
5	R/W	BINC Enable increment of destination address 0 = Do not increment 1 = Increment according to the value of BW	0x0
4	R/W	DREQ_MODE 0 = DMA channel starts immediately 1 = DMA channel must be triggered by peripheral DMA request (see also the description of DMA_REQ_MUX_REG)	0x0
3	R/W	IRQ_ENABLE 0 = Disable interrupt on this channel 1 = Enable interrupt on this channel	0x0
2:1	R/W	BW Bus transfer width: 00 = 1 Byte (suggested for peripherals like UART and 8-bit SPI) 01 = 2 Bytes (suggested for peripherals like I2C and 16-bit SPI) 10 = 4 Bytes (suggested for Memory-to-Memory transfers) 11 = Reserved	0x0
0	R/W	DMA_ON 0 = DMA channel is off, clocks are disabled 1 = DMA channel is enabled. This bit is automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set.	0x0

Table 172: DMA2_IDX_REG (0x5000362E)

Bit	Mode	Symbol/Description	Reset
15:0	R	DMA2_IDX This (read-only) register determines the data items currently fetched by the DMA channel, during an on-going transfer. When the transfer is completed, the register is automatically reset to 0. The DMA channel uses this register to form the source/destination address of the next DMA cycle, considering also AINC/BINC and BW.	0x0

Table 173: DMA3_A_STARTL_REG (0x50003630)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA3_A_STARTL Source start address, lower 16 bits	0x0

Table 174: DMA3_A_STARTH_REG (0x50003632)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA3_A_STARTH Source start address, upper 16 bits	0x0

Table 175: DMA3_B_STARTL_REG (0x50003634)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA3_B_STARTL Destination start address, lower 16 bits	0x0

Table 176: DMA3_B_STARTH_REG (0x50003636)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA3_B_STARTH Destination start address, upper 16 bits	0x0

Table 177: DMA3_INT_REG (0x50003638)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA3_INT Number of transfers until an interrupt is generated. The interrupt is generated after a transfer, if DMAx_INT_REG is equal to DMAx_IDX_REG and before DMAx_IDX_REG is incremented. The bit field IRQ_ENABLE of DMAx_CTRL_REG must be set to 1 to let the controller generate the interrupt.	0x0

Table 178: DMA3_LEN_REG (0x5000363A)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	DMA3_LEN DMA channel's transfer length. DMAx_LEN of value 0, 1, 2, ... results into an actual transfer length of 1, 2, 3, ...	0x0

Table 179: DMA3_CTRL_REG (0x5000363C)

Bit	Mode	Symbol/Description	Reset
15:14	R	- Reserved	0x0
13	R/W	REQ_SENSE 0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests	0x0
12	R/W	DMA_INIT 0 = DMA performs copy A1 to B1, A2 to B2, and so on ... 1 = DMA performs copy of A1 to B1, B2, and so on ... This feature is useful for memory initialization to any value. Thus, BINC must be set to 1, while AINC is don't care, as only one fetch from A is done. This process cannot be interrupted by other DMA channels. It is also noted that DMA_INIT should not be used when DREQ_MODE = 1.	0x0
11	R/W	DMA_IDLE 0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority. 1 = Interrupting mode, the DMA inserts a wait cycle after each store allowing the CPU to steal cycles or cache to perform a burst read. If DREQ_MODE = 1, DMA_IDLE is don't care.	0x0
10:8	R/W	DMA_PRIO The priority level determines which DMA channel is granted access for transferring data, in case more than one channel is active and request the bus at the same time. The greater the value, the higher the priority. In specific: 000 = lowest priority 111 = highest priority If different channels with equal priority level values request the bus at the same time, an inherent priority mechanism is applied. According to this mechanism, if, for example, both the DMA0 and DMA1 channels have the same priority level, then DMA0 is first granted access to the bus.	0x0
7	R/W	CIRCULAR 0 = Normal mode. The DMA channel stops after having completed the transfer of length determined by DMAx_LEN_REG. DMA_ON automatically deasserts when the transfer is completed. 1 = Circular mode (applicable only if DREQ_MODE = 1). In this mode, DMA_ON never deasserts, as the DMA channel automatically resets DMAx_IDX_REG and starts a new transfer.	0x0
6	R/W	AINC Enable increment of source address. 0 = Do not increment (source address stays the same during the transfer) 1 = Increment according to the value of BW bit field (by 1, when BW = 00; by 2, when BW = 01; by 4, when BW = 10)	0x0
5	R/W	BINC Enable increment of destination address. 0 = Do not increment (destination address stays the same during the transfer) 1 = Increment according to the value of BW bit field (by 1, when BW = 00; by 2, when BW = 01; by 4, when BW = 10)	0x0
4	R/W	DREQ_MODE 0 = DMA channel starts immediately	0x0

Bit	Mode	Symbol/Description	Reset
		1 = DMA channel must be triggered by peripheral DMA request (see also the description of DMA_REQ_MUX_REG)	
3	R/W	IRQ_ENABLE 0 = Disable interrupt on this channel 1 = Enable interrupt on this channel	0x0
2:1	R/W	BW Bus transfer width: 00 = 1 Byte (suggested for peripherals like UART and 8-bit SPI) 01 = 2 Bytes (suggested for peripherals like I2C and 16-bit SPI) 10 = 4 Bytes (suggested for Memory-to-Memory transfers) 11 = Reserved	0x0
0	R/W	DMA_ON 0 = DMA channel is off, clocks are disabled 1 = DMA channel is enabled. This bit is automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set.	0x0

Table 180: DMA3_IDX_REG (0x5000363E)

Bit	Mode	Symbol/Description	Reset
15:0	R	DMA3_IDX This (read-only) register determines the data items currently fetched by the DMA channel, during an on-going transfer. When the transfer is completed, the register is automatically reset to 0. The DMA channel uses this register to form the source/destination address of the next DMA cycle, considering also AINC/BINC and BW.	0x0

Table 181: DMA_REQ_MUX_REG (0x50003680)

Bit	Mode	Symbol/Description	Reset
15:12	R/W	- Reserved	0xF
11:8	R/W	- Reserved	0xF
7:4	R/W	DMA23_SEL Select which combination of peripherals are mapped on the DMA channels. The peripherals are mapped as pairs on two channels. Hence, the first DMA request (peripheral-to-memory) is mapped on channel 2 and the second (memory-to-peripheral) on channel 3. See also the description of DMA01_SEL bit field of this register for the supported peripherals.	0xF
3:0	R/W	DMA01_SEL Select which combination of peripherals are mapped on the DMA channels. The peripherals are mapped as pairs on two channels. Hence, the first DMA request (peripheral-to-memory) is mapped on channel 0 and the second (memory-to-peripheral) on channel 1. 0x0: SPI_rx/SPI_tx 0x1: Reserved 0x2: UART_rx/UART_tx	0xF

Bit	Mode	Symbol/Description	Reset
		0x3: UART2_rx/UART2_tx 0x4: I2C_rx/I2C_tx 0x5: GP_ADC (RX only) 0x6-0xE: Reserved 0xF: None Note: If any of the two available peripheral selector fields (DMA01_SEL, DMA23_SEL) have the same value, the lesser significant selector has higher priority and controls the DMA acknowledge. Hence, if DMA01_SEL = DMA23_SEL, the channels 0 and 1 generate the DMA acknowledge signals for the selected peripheral. Consequently, it is suggested to assign the intended peripheral value to a unique selector field.	

Table 182: DMA_INT_STATUS_REG (0x50003682)

Bit	Mode	Symbol/Description	Reset
15:8	R	- Reserved	0x0
7	R	- Reserved	0x0
6	R	- Reserved	0x0
5	R	- Reserved	0x0
4	R	- Reserved	0x0
3	R	DMA_IRQ_CH3 0: IRQ on channel 3 is not set 1: IRQ on channel 3 is set	0x0
2	R	DMA_IRQ_CH2 0: IRQ on channel 2 is not set 1: IRQ on channel 2 is set	0x0
1	R	DMA_IRQ_CH1 0: IRQ on channel 1 is not set 1: IRQ on channel 1 is set	0x0
0	R	DMA_IRQ_CH0 0: IRQ on channel 0 is not set 1: IRQ on channel 0 is set	0x0

Table 183: DMA_CLEAR_INT_REG (0x50003684)

Bit	Mode	Symbol/Description	Reset
15:8	R	- Reserved	0x0
7	R	- Reserved	0x0

Bit	Mode	Symbol/Description	Reset
6	R	- Reserved	0x0
5	R	- Reserved	0x0
4	R	- Reserved	0x0
3	R0/W	DMA_RST_IRQ_CH3 Writing 1 resets the status bit of DMA_INT_STATUS_REG for channel 3; writing 0 has no effect.	0x0
2	R0/W	DMA_RST_IRQ_CH2 Writing 1 resets the status bit of DMA_INT_STATUS_REG for channel 2; writing 0 has no effect.	0x0
1	R0/W	DMA_RST_IRQ_CH1 Writing 1 resets the status bit of DMA_INT_STATUS_REG for channel 1; writing 0 has no effect.	0x0
0	R0/W	DMA_RST_IRQ_CH0 Writing 1 resets the status bit of DMA_INT_STATUS_REG for channel 0; writing 0 has no effect.	0x0

31.6 General-Purpose ADC Registers

Table 184: Register map GPADC

Address	Register	Description
0x50001500	GP_ADC_CTRL_REG	General Purpose ADC Control Register
0x50001502	GP_ADC_CTRL2_REG	General Purpose ADC Second Control Register
0x50001504	GP_ADC_CTRL3_REG	General Purpose ADC Third Control Register
0x50001506	GP_ADC_SEL_REG	General Purpose ADC Input Selection Register
0x50001508	GP_ADC_OFFP_REG	General Purpose ADC Positive Offset Register
0x5000150a	GP_ADC_OFFN_REG	General Purpose ADC Negative Offset Register
0x5000150e	GP_ADC_CLEAR_INT_REG	General Purpose ADC Clear Interrupt Register
0x50001510	GP_ADC_RESULT_REG	General Purpose ADC Result Register

Table 185: GP_ADC_CTRL_REG (0x50001500)

Bit	Mode	Symbol/Description	Reset
12	R/W	DIE_TEMP_EN Enables the die-temperature sensor. Output can be measured on GPADC input 4.	0x0
11	R/W	- Reserved	0x0
10	R/W	GP_ADC_LDO_HOLD 0: GPADC LDO tracking bandgap reference 1: GPADC LDO hold sampled bandgap reference	0x0
9	R/W	GP_ADC_CHOP 0: Chopper mode off 1: Chopper mode enabled. Takes two samples with opposite GP_ADC_SIGN to cancel the internal offset voltage of the ADC; Highly recommended for DC-measurements.	0x0
8	R/W	GP_ADC_SIGN 0: Default 1: Conversion with opposite sign at input and output to cancel out the internal offset of the ADC and low-frequency	0x0
7	R/W	GP_ADC_MUTE 0: Normal operation 1: Mute ADC input. Takes sample at mid-scale (to dertermine the internal offset and/or noise of the ADC with regards to VDD_REF which is also sampled by the ADC).	0x0
6	R/W	GP_ADC_SE 0: Differential mode 1: Single ended mode	0x0
5	R/W	GP_ADC_MINT 0: Disable (mask) GP_ADC_INT. 1: Enable GP_ADC_INT to ICU.	0x0
4	R	GP_ADC_INT	0x0

Bit	Mode	Symbol/Description	Reset
		1: AD conversion ready and has generated an interrupt. Must be cleared by writing any value to GP_ADC_CLEAR_INT_REG.	
3	R/W	GP_ADC_DMA_EN 0: DMA functionality disabled 1: DMA functionality enabled	0x0
2	R/W	GP_ADC_CONT 0: Manual ADC mode, a single result will be generated after setting the GP_ADC_START bit. 1: Continuous ADC mode, new ADC results will be constantly stored in GP_ADC_RESULT_REG. Still GP_ADC_START has to be set to start the execution. The time between conversions is configurable with GP_ADC_INTERVAL.	0x0
1	R/W	GP_ADC_START 0: ADC conversion ready. 1: If a 1 is written, the ADC starts a conversion. After the conversion this bit will be set to 0 and the GP_ADC_INT bit will be set. It is not allowed to write this bit while it is not (yet) zero.	0x0
0	R/W	GP_ADC_EN 0: LDO is off and ADC is disabled.. 1: LDO is turned on and afterwards the ADC is enabled.	0x0

Table 186: GP_ADC_CTRL2_REG (0x50001502)

Bit	Mode	Symbol/Description	Reset
15:13	R/W	GP_ADC_STORE_DEL 0: Data is stored after handshake synchronization 1: Data is stored 2 ADC_CLK cycles after internal start trigger 7: Data is stored 8 ADC_CLK cycles after internal start trigger	0x0
12:9	R/W	GP_ADC_SMPL_TIME 0: The sample time (switch is closed) is two ADC_CLK cycles 1: The sample time is 1*8 ADC_CLK cycles 2: The sample time is 2*8 ADC_CLK cycles 15: The sample time is 15*8 ADC_CLK cycles	0x1
8:6	R/W	GP_ADC_CONV_NRS 0: 1 sample is taken or 2 in case ADC_CHOP is active. 1: 2 samples are taken. 2: 4 samples are taken. 7: 128 samples are taken.	0x0
5:4	R/W	- Reserved	0x1
3	R/W	- Reserved	0x0
2	R/W	GP_ADC_I20U 1: Adds 20 μA constant load current at the ADC LDO to minimize ripple on the reference voltage of the ADC.	0x0
1:0	R/W	GP_ADC_ATTN	0x0

Bit	Mode	Symbol/Description	Reset
		0: No attenuator (input voltages up to 0.9 V allowed) 1: Enabling 2x attenuator (input voltages up to 1.8 V allowed) 2: Enabling 3x attenuator (input voltages up to 2.7 V allowed) 3: Enabling 4x attenuator (input voltages up to 3.6 V allowed) Enabling the attenuator requires a longer sampling time.	

Table 187: GP_ADC_CTRL3_REG (0x50001504)

Bit	Mode	Symbol/Description	Reset
15:8	R/W	GP_ADC_INTERVAL Defines the interval between two ADC conversions in case GP_ADC_CONT is set. 0: No extra delay between two conversions. 1: 1.024 ms interval between two conversions. 2: 2.048 ms interval between two conversions. 255: 261.12 ms interval between two conversions.	0x0
7:0	R/W	GP_ADC_EN_DEL Defines the delay for enabling the ADC after enabling the LDO. 0: Not allowed 1: 4x ADC_CLK period. n: n*4x ADC_CLK period.	0x40

Table 188: GP_ADC_SEL_REG (0x50001506)

Bit	Mode	Symbol/Description	Reset
7	R/W	- Reserved	0x0
6:4	R/W	GP_ADC_SEL_P ADC positive input selection. 0: ADC0 (P0[1]) 1: ADC1 (P0[2]) 2: ADC2 (P0[6]) 3: ADC3 (P0[7]) 4: Temperature Sensor 5: VBAT_HIGH 6: VBAT_LOW 7: VDDD	0x0
3	R/W	- Reserved	0x0
2:0	R/W	GP_ADC_SEL_N ADC negative input selection. Differential only (GP_ADC_SE=0). 0: ADC0 (P0[1]) 1: ADC1 (P0[2]) 2: ADC2 (P0[6]) 3: ADC3 (P0[7]) All other combinations are reserved.	0x0

Table 189: GP_ADC_OFFP_REG (0x50001508)

Bit	Mode	Symbol/Description	Reset
9:0	R/W	GP_ADC_OFFP Offset adjust of the "positive" array of ADC-network (effective if "GP_ADC_SE=0", or "GP_ADC_SE=1 AND GP_ADC_SIGN=0 OR GP_ADC_CHOP=1")	0x200

Table 190: GP_ADC_OFFN_REG (0x5000150A)

Bit	Mode	Symbol/Description	Reset
9:0	R/W	GP_ADC_OFFN Offset adjust of the "negative" array of ADC-network (effective if "GP_ADC_SE=0", or "GP_ADC_SE=1 AND GP_ADC_SIGN=1 OR GP_ADC_CHOP=1")	0x200

Table 191: GP_ADC_CLEAR_INT_REG (0x5000150E)

Bit	Mode	Symbol/Description	Reset
15:0	R0/W	GP_ADC_CLR_INT Writing any value to this register clears the ADC_INT interrupt. Reading returns 0.	0x0

Table 192: GP_ADC_RESULT_REG (0x50001510)

Bit	Mode	Symbol/Description	Reset
15:0	R	GP_ADC_VAL Returns the 10 up to 16 bits linear value of the last AD conversion. The upper 10 bits are always valid, the lower 6 bits are only valid in case oversampling has been applied. Two samples result in one extra bit and 64 samples result in six extra bits.	0x0

31.7 General Purpose I/O Registers

Table 193: Register map GPIO

Address	Register	Description
0x50003000	P0_DATA_REG	P0 Data input/output Register
0x50003002	P0_SET_DATA_REG	P0 Set port pins Register
0x50003004	P0_RESET_DATA_REG	P0 Reset port pins Register
0x50003006	P00_MODE_REG	P00 Mode Register
0x50003008	P01_MODE_REG	P01 Mode Register
0x5000300a	P02_MODE_REG	P02 Mode Register
0x5000300c	P03_MODE_REG	P03 Mode Register
0x5000300e	P04_MODE_REG	P04 Mode Register
0x50003010	P05_MODE_REG	P05 Mode Register
0x50003012	P06_MODE_REG	P06 Mode Register
0x50003014	P07_MODE_REG	P07 Mode Register
0x50003016	P08_MODE_REG	P08 Mode Register
0x50003018	P09_MODE_REG	P09 Mode Register
0x5000301a	P010_MODE_REG	P010 Mode Register
0x5000301c	P011_MODE_REG	P011 Mode Register
0x5000301e	PAD_WEAK_CTRL_REG	Pad driving strength control Register
0x50003042	TEST_CTRL5_REG	Test Control Register 5

Table 194: P0_DATA_REG (0x50003000)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11:0	R/W	P0_DATA Sets P0 output register when written; Returns the value of P0 port when read	0x0

Table 195: P0_SET_DATA_REG (0x50003002)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11:0	R0/W	P0_SET Writing a 1 to P0[x] sets P0[x] to 1. Writing 0 is discarded, reading returns 0	0x0

Table 196: P0_RESET_DATA_REG (0x50003004)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11:0	R0/W	P0_RESET	0x0

Bit	Mode	Symbol/Description	Reset
		Writing a 1 to P0[x] sets P0[x] to 0. Writing 0 is discarded, reading returns 0.	

Table 197: P00_MODE_REG (0x50003006)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected In ADC mode, these bits are don't care.	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID Function of port 0 = GPIO (pin direction determined by "PUPD" field) 1 = UART1_RX 2 = UART1_TX 3 = UART2_RX 4 = UART2_TX 5 = SYS_CLK 6 = LP_CLK 7 = SPI_DATA_READY (new, pin direction automatically set to INPUT) 8 = Reserved 9 = I2C_SCL 10 = I2C_SDA 11 = PWM5 12 = PWM6 13 = PWM7 14 = Reserved 15 = ADC (only for P0_1, P0_2, P0_6 and P0_7) 16 = PWM0 17 = PWM1 18 = BLE_DIAG (signals mapped to P0[3:0] are also mapped to P0[11:8]) 19 = UART1_CTSN 20 = UART1_RTSN 21 = Reserved 22 = Reserved 23 = PWM2 24 = PWM3 25 = PWM4 26 = SPI_DI 27 = SPI_DO 28 = SPI_CLK	0x0

Bit	Mode	Symbol/Description	Reset
		29 = SPI_CSN0 30 = SPI_CSN1 31 = Reserved Note: When a certain input function (like SPI_DI) is selected on more than 1 pins, the pin of the lowest index has the highest priority.	

Table 198: P01_MODE_REG (0x50003008)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected In ADC mode, these bits are don't care.	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 199: P02_MODE_REG (0x5000300A)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected In ADC mode, these bits are don't care.	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 200: P03_MODE_REG (0x5000300C)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected	0x2

Bit	Mode	Symbol/Description	Reset
		01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected In ADC mode, these bits are don't care.	
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 201: P04_MODE_REG (0x5000300E)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 202: P05_MODE_REG (0x50003010)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 203: P06_MODE_REG (0x50003012)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD	0x2

Bit	Mode	Symbol/Description	Reset
		00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 204: P07_MODE_REG (0x50003014)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 205: P08_MODE_REG (0x50003016)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 206: P09_MODE_REG (0x50003018)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD	0x2

Bit	Mode	Symbol/Description	Reset
		00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 207: P010_MODE_REG (0x5000301A)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 208: P011_MODE_REG (0x5000301C)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:8	R/W	PUPD 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected 11 = Output, no resistors selected	0x2
7:5	-	- Reserved	0x0
4:0	R/W	PID See P00_MODE_REG[PID]	0x0

Table 209: PAD_WEAK_CTRL_REG (0x5000301E)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11:0	R/W	PAD_LOW_DRV	0x0

Bit	Mode	Symbol/Description	Reset
		0 = Normal operation 1 = Reduces the driving strength of P0_x pad. Bit x controls the driving strength of P0_x, x=0, 1,..., 11.	

Table 210: TEST_CTRL5_REG (0x50003042)

Bit	Mode	Symbol/Description	Reset
3:2	R/W	- Reserved	0x0
1	R/W	- Reserved	0x0
0	R/W	- Reserved	0x0

31.8 General Purpose Registers

Table 211: Register map GPREG

Address	Register	Description
0x50003300	SET_FREEZE_REG	Controls freezing of various timers/counters.
0x50003302	RESET_FREEZE_REG	Controls unfreezing of various timers/counters.
0x50003304	DEBUG_REG	Various debug information register.
0x50003306	GP_STATUS_REG	General purpose system status register.
0x50003308	GP_CONTROL_REG	General purpose system control register.
0x5000330a	BLE_TIMER_REG	BLE FINECNT sampled value while in deep sleep state.

Table 212: SET_FREEZE_REG (0x50003300)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4	R/W	FRZ_DMA If 1, the DMA is frozen, 0 is discarded.	0x0
3	R/W	FRZ_WDOG If 1, the watchdog timer is frozen, 0 is discarded. WATCHDOG_CTRL_REG[NMI_RST] must be 0 to allow the freeze function.	0x0
2	R/W	FRZ_BLETIM If 1, the Bluetooth LE master clock is frozen, 0 is discarded.	0x0
1	R/W	FRZ_SWTIM If 1, the Software Timer (TIMER0) is frozen, 0 is discarded.	0x0
0	R/W	FRZ_WKUPTIM If 1, the Wake-Up Timer is frozen, 0 is discarded.	0x0

Table 213: RESET_FREEZE_REG (0x50003302)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4	R/W	FRZ_DMA If 1, the DMA continues, 0 is discarded.	0x0
3	R/W	FRZ_WDOG If 1, the watchdog timer continues, 0 is discarded.	0x0
2	R/W	FRZ_BLETIM If 1, the the Bluetooth LE master clock continues, 0 is discarded.	0x0
1	R/W	FRZ_SWTIM If 1, the Software Timer (TIMER0) continues, 0 is discarded.	0x0
0	R/W	FRZ_WKUPTIM If 1, the Wake-Up Timer continues, 0 is discarded.	0x0

Table 214: DEBUG_REG (0x50003304)

Bit	Mode	Symbol/Description	Reset
15:1	R/W	- Reserved	0x0
0	R/W	DEBUGS_FREEZE_EN Default 1, freezing of the on-chip timers is enabled when the Cortex is halted in DEBUG state. If 0, freezing of the on-chip timers is depending on FREEZE_REG when the Cortex is halted in DEBUG state except the watchdog timer. The watchdog timer is always frozen when the Cortex is halted in DEBUG state.	0x1

Table 215: GP_STATUS_REG (0x50003306)

Bit	Mode	Symbol/Description	Reset
15:2	-	- Reserved	0x0
1	R/W	- Reserved	0x0
0	R/W	CAL_PHASE If 1, it designates that the chip is in Calibration Phase, that means the OTP has been initially programmed but no Calibration has occurred.	0x0

Table 216: GP_CONTROL_REG (0x50003308)

Bit	Mode	Symbol/Description	Reset
15:7	-	- Reserved	0x0
6:5	R/W	BLE_TIMER_DATA_CTRL See BLE_TIMER_REG.	0x0
4	R/W	CPU_DMA_BUS_PRIO Controls the CPU DMA system bus priority: If 0, the CPU has highest priority. If 1, the DMA has highest priority.	0x0
3	-	- Reserved	0x0
2	R	BLE_WAKEUP_LP_IRQ The current value of the BLE_WAKEUP_LP_IRQ interrupt request.	0x0
1	-	- Reserved	0x0
0	R/W	BLE_WAKEUP_REQ If 1, the Bluetooth LE wakes up. Must be kept high at least for 1 low power clock period. If the Bluetooth LE is in DEEP SLEEP state, then by setting this bit it causes the wake-up LP IRQ to be asserted with a delay of 3 to 4 low power cycles.	0x0

Table 217: BLE_TIMER_REG (0x5000330A)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:0	R/W	BLE_TIMER_DATA Operation depends on GP_CONTROL_REG->BLE_TIMER_DATA_CTRL. If BLE_TIMER_DATA_CTRL = 0, then: This register is located at the Always On Power Domain and it holds the automatically sampled value of the BLE FINECNT timer. The hardware automatically samples the value into this register during the sequence of "BLE Sleep On" and restores automatically the value during the Bluetooth LE Wake-up sequence. The software may read and modify the value while the Bluetooth LE is in SLEEP state. While the Bluetooth LE is awake, the value of the register has no meaning, while changing the value by writing another one has no effect in the operation of the Bluetooth LE core. There is a constraint when the software performs a write-read sequence where it has to inject a one cycle delay in between (for example, write-NOP-read) to read back the correct value. If BLE_TIMER_DATA_CTRL is non 0, then write operations have the same effect as when BLE_TIMER_DATA_CTRL = 0, while for read operations: BLE_TIMER_DATA_CTRL = 1: then reading BLE_TIMER_REG returns "deepsldur[9:0]". BLE_TIMER_DATA_CTRL = 2: then reading BLE_TIMER_REG returns "deepsltime_samp[9:0]". BLE_TIMER_DATA_CTRL = 3: then reading BLE_TIMER_REG returns "{deep_sleep_stat_monitor, deepsltime_samp[18:10]}.	0x0

31.9 I2C Interface Registers

Table 218: Register map I2C

Address	Register	Description
0x50001300	I2C_CON_REG	I2C Control Register
0x50001304	I2C_TAR_REG	I2C Target Address Register
0x50001308	I2C_SAR_REG	I2C Slave Address Register
0x50001310	I2C_DATA_CMD_REG	I2C Rx/Tx Data Buffer and Command Register
0x50001314	I2C_SS_SCL_HCNT_REG	Standard Speed I2C Clock SCL High Count Register
0x50001318	I2C_SS_SCL_LCNT_REG	Standard Speed I2C Clock SCL Low Count Register
0x5000131c	I2C_FS_SCL_HCNT_REG	Fast Speed I2C Clock SCL High Count Register
0x50001320	I2C_FS_SCL_LCNT_REG	Fast Speed I2C Clock SCL Low Count Register
0x5000132c	I2C_INTR_STAT_REG	I2C Interrupt Status Register
0x50001330	I2C_INTR_MASK_REG	I2C Interrupt Mask Register
0x50001334	I2C_RAW_INTR_STAT_REG	I2C Raw Interrupt Status Register
0x50001338	I2C_RX_TL_REG	I2C Receive FIFO Threshold Register
0x5000133c	I2C_TX_TL_REG	I2C Transmit FIFO Threshold Register
0x50001340	I2C_CLR_INTR_REG	Clear Combined and Individual Interrupt Register
0x50001344	I2C_CLR_RX_UNDER_REG	Clear RX_UNDER Interrupt Register
0x50001348	I2C_CLR_RX_OVER_REG	Clear RX_OVER Interrupt Register
0x5000134c	I2C_CLR_TX_OVER_REG	Clear TX_OVER Interrupt Register
0x50001350	I2C_CLR_RD_REQ_REG	Clear RD_REQ Interrupt Register
0x50001354	I2C_CLR_TX_ABRT_REG	Clear TX_ABRT Interrupt Register
0x50001358	I2C_CLR_RX_DONE_REG	Clear RX_DONE Interrupt Register
0x5000135c	I2C_CLR_ACTIVITY_REG	Clear ACTIVITY Interrupt Register
0x50001360	I2C_CLR_STOP_DET_REG	Clear STOP_DET Interrupt Register
0x50001364	I2C_CLR_START_DET_REG	Clear START_DET Interrupt Register
0x50001368	I2C_CLR_GEN_CALL_REG	Clear GEN_CALL Interrupt Register
0x5000136c	I2C_ENABLE_REG	I2C Enable Register
0x50001370	I2C_STATUS_REG	I2C Status Register
0x50001374	I2C_TXFLR_REG	I2C Transmit FIFO Level Register
0x50001378	I2C_RXFLR_REG	I2C Receive FIFO Level Register
0x5000137c	I2C_SDA_HOLD_REG	I2C SDA Hold Time Length Register

Address	Register	Description
0x50001380	I2C_TX_ABRT_SOURCE_REG	I2C Transmit Abort Source Register
0x50001388	I2C_DMA_CR_REG	DMA Control Register
0x5000138c	I2C_DMA_TDLR_REG	DMA Transmit Data Level Register
0x50001390	I2C_DMA_RDLR_REG	I2C Receive Data Level Register
0x50001394	I2C_SDA_SETUP_REG	I2C SDA Setup Register
0x50001398	I2C_ACK_GENERAL_CALL_REG	I2C ACK General Call Register
0x5000139c	I2C_ENABLE_STATUS_REG	I2C Enable Status Register
0x500013a0	I2C_IC_FS_SPKLEN_REG	I2C SS and FS spike suppression limit Size

Table 219: I2C_CON_REG (0x50001300)

Bit	Mode	Symbol/Description	Reset
15:7	-	- Reserved	0x0
6	R/W	I2C_SLAVE_DISABLE Slave enabled or disabled after reset is applied, which means software does not have to configure the slave. 0 = Slave is enabled 1 = Slave is disabled Software should ensure that if this bit is written with "0", then bit 0 should also be written with a "0".	0x1
5	R/W	I2C_RESTART_EN Determines whether RESTART conditions may be sent when acting as a master 0 = Disable 1 = Enable	0x1
4	R/W	I2C_10BITADDR_MASTER Controls whether the controller starts its transfers in 7- or 10-bit addressing mode when acting as a master. 0 = 7-bit addressing 1 = 10-bit addressing	0x1
3	R/W	I2C_10BITADDR_SLAVE When acting as a slave, this bit controls whether the controller responds to 7- or 10-bit addresses. 0 = 7-bit addressing 1 = 10-bit addressing	0x1
2:1	R/W	I2C_SPEED These bits control at which speed the controller operates. 1 = Standard mode (100 kbit/s) 2 = Fast mode (400 kbit/s) Note: The actuall speed depends on the pcb traces capacitance as well as on the values of the external pull-up resistorts. For an exact speed match, trimming might be required, by adjusting the values of I2C_SS_SCL_HCNT_REG, I2C_SS_SCL_LCNT_REG, I2C_FS_SCL_HCNT_REG, I2C_FS_SCL_LCNT_REG	0x2

Bit	Mode	Symbol/Description	Reset
		registers. The reset values of those registers were calculated with the assumption of 4.3 kΩ external pull-up resistors.	
0	R/W	I2C_MASTER_MODE This bit controls whether the controller master is enabled. 0 = Master disabled 1 = Master enabled Software should ensure that if this bit is written with "1", then bit 6 should also be written with a "1".	0x1

Table 220: I2C_TAR_REG (0x50001304)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11	R/W	SPECIAL This bit indicates whether software performs a General Call or START BYTE command. 0: Ignore bit 10 GC_OR_START and use IC_TAR normally 1: Perform special I2C command as specified in GC_OR_START bit	0x0
10	R/W	GC_OR_START If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the controller. 0: General Call Address - after issuing a General Call, only writes may be performed. Attempting to issue a read command results in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. The controller remains in General Call mode until the SPECIAL bit value (bit 11) is cleared. 1: START BYTE	0x0
9:0	R/W	IC_TAR This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE, the CPU needs to write only once into these bits. Note: If the IC_TAR and IC_SAR are the same, loopback exists but the FIFOs are shared between master and slave, so full loopback is not feasible. Only one direction loopback mode is supported (simplex), not duplex. A master cannot transmit to itself; it can transmit to only a slave.	0x55

Table 221: I2C_SAR_REG (0x50001308)

Bit	Mode	Symbol/Description	Reset
15:10	-	- Reserved	0x0
9:0	R/W	IC_SAR The IC_SAR holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only IC_SAR[6:0] is used. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.	0x55

Table 222: I2C_DATA_CMD_REG (0x50001310)

Bit	Mode	Symbol/Description	Reset
15:11	-	- Reserved	0x0
10	R/W	I2C_RESTART This bit controls whether a RESTART is issued before the byte is sent or received. If IC_RESTART_EN is 1, a RESTART is issued before the data is sent/received (according to the value of CMD), regardless of whether or not the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead. If IC_RESTART_EN is 1, a RESTART is issued only if the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead. Reset value: 0x0	0x0
9	R/W	I2C_STOP This bit controls whether a STOP is issued after the byte is sent or received. STOP is issued after this byte, regardless of whether or not the TX FIFO is empty. If the TX FIFO is not empty, the master immediately tries to start a new transfer by issuing a START and arbitrating for the bus. STOP is not issued after this byte, regardless of whether or not the TX FIFO is empty. If the TX FIFO is not empty, the master continues the current transfer by sending/receiving data bytes according to the value of the CMD bit. If the TX FIFO is empty, the master holds the SCL line low and stalls the bus until a new command is available in the TX FIFO. Reset value: 0x0	0x0
8	R/W	I2C_CMD This bit controls whether a read or a write is performed. This bit does not control the direction when the I2C Ctrl acts as a slave. It controls only the direction when it acts as a master. 1 = Read 0 = Write When a command is entered in the TX FIFO, this bit distinguishes the write and read commands. In slave-receiver mode, this bit is a "don't care" because writes to this register are not required. In slave-transmitter mode, a "0" indicates that CPU data is to be transmitted and as DAT or IC_DATA_CMD[7:0]. When programming this bit, you should remember the following: attempting to perform a read operation after a General Call command has been sent results in a TX_ABRT interrupt (bit 6 of the I2C_RAW_INTR_STAT_REG), unless bit 11 (SPECIAL) in the I2C_TAR register has been cleared. If a "1" is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs. NOTE: It is possible that while attempting a master I2C read transfer on the controller, a RD_REQ interrupt may have occurred simultaneously due to a remote I2C master addressing the controller. In this type of scenario, it ignores the I2C_DATA_CMD write, generates a TX_ABRT interrupt, and waits to service the RD_REQ interrupt	0x0
7:0	R/W	DAT This register contains the data to be transmitted or received on the I2C bus. If you are writing to this register and want to perform a read, bits 7:0 (DAT) are ignored by the controller. However, when you read this register, these bits return the value of data received on the controller's interface.	0x0

Table 223: I2C_SS_SCL_HCNT_REG (0x50001314)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	IC_SS_SCL_HCNT	0x48

Bit	Mode	Symbol/Description	Reset
		This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for standard speed. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. NOTE: This register must not be programmed to a value higher than 65525, because the controller uses a 16-bit counter to flag an I2C bus idle condition when this counter reaches a value of IC_SS_SCL_HCNT + 10.	

Table 224: I2C_SS_SCL_LCNT_REG (0x50001318)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	IC_SS_SCL_LCNT	0x4F
		This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for standard speed. This register can be written only when the I2C interface is disabled which corresponds to the I2C_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, results in 8 being set.	

Table 225: I2C_FS_SCL_HCNT_REG (0x5000131C)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	IC_FS_SCL_HCNT	0x8
		This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This register can be written only when the I2C interface is disabled, which corresponds to the I2C_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set.	

Table 226: I2C_FS_SCL_LCNT_REG (0x50001320)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	IC_FS_SCL_LCNT	0x17
		This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low-period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This register can be written only when the I2C interface is disabled, which corresponds to the I2C_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set. For designs with APB_DATA_WIDTH = 8 the order of programming is important to ensure the correct operation of the controller. The lower byte must be programmed first. Then the upper byte is programmed.	

Table 227: I2C_INTR_STAT_REG (0x5000132C)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11	R	R_GEN_CALL Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling controller or when the CPU reads bit 0 of the I2C_CLR_GEN_CALL register. The controller stores the received data in the RX buffer.	0x0
10	R	R_START_DET Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether controller is operating in Slave or Master mode.	0x0
9	R	R_STOP_DET Indicates whether a STOP condition has occurred on the I2C interface regardless of whether controller is operating in Slave or Master mode.	0x0
8	R	R_ACTIVITY This bit captures I2C Ctrl activity and stays set until it is cleared. There are four ways to clear it: => Disabling the I2C Ctrl => Reading the IC_CLR_ACTIVITY register => Reading the IC_CLR_INTR register => System reset When this bit is set, it stays set unless one of the four methods is used to clear it. Even if the controller module is idle, this bit remains set until cleared, indicating that there was activity on the bus.	0x0
7	R	R_RX_DONE When the controller is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.	0x0
6	R	R_TX_ABRT This bit indicates if the controller, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a "transmit abort". When this bit is set to 1, the I2C_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes place. NOTE: The controller flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register I2C_CLR_TX_ABRT is read. When this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.	0x0
5	R	R_RD_REQ This bit is set to 1 when the controller is acting as a slave and another I2C master is attempting to read data from the controller. The controller holds the I2C bus in a wait state (SCL = 0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the I2C_DATA_CMD register. This bit is set to 0 just after the processor reads the I2C_CLR_RD_REQ register.	0x0
4	R	R_TX_EMPTY This bit is set to 1 when the transmit buffer is at or below the threshold value set in the I2C_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold. When the IC_ENABLE bit 0 is 0, the TX FIFO is	0x0

Bit	Mode	Symbol/Description	Reset
		flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer activity, then with ic_en = 0, this bit is set to 0.	
3	R	R_TX_OVER Set during transmit if the transmit buffer is filled to 32 and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.	0x0
2	R	R_RX_FULL Set when the receive buffer reaches or goes above the RX_TL threshold in the I2C_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (I2C_ENABLE[0] = 0), the RX FIFO is flushed and held in reset; therefore, the RX FIFO is not full. So this bit is cleared when the I2C_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues.	0x0
1	R	R_RX_OVER Set if the receive buffer is completely filled to 32 and an additional byte is received from an external I2C device. The controller acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (I2C_ENABLE[0] = 0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.	0x0
0	R	R_RX_UNDER Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (I2C_ENABLE[0] = 0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.	0x0

Table 228: I2C_INTR_MASK_REG (0x50001330)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11	R/W	M_GEN_CALL These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1
10	R/W	M_START_DET These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x0
9	R/W	M_STOP_DET These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x0
8	R/W	M_ACTIVITY These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x0
7	R/W	M_RX_DONE These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1
6	R/W	M_TX_ABRT	0x1

Bit	Mode	Symbol/Description	Reset
		These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	
5	R/W	M_RD_REQ These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1
4	R/W	M_TX_EMPTY These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1
3	R/W	M_TX_OVER These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1
2	R/W	M_RX_FULL These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1
1	R/W	M_RX_OVER These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1
0	R/W	M_RX_UNDER These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register.	0x1

Table 229: I2C_RAW_INTR_STAT_REG (0x50001334)

Bit	Mode	Symbol/Description	Reset
15:12	-	- Reserved	0x0
11	R	GEN_CALL Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling controller or when the CPU reads bit 0 of the I2C_CLR_GEN_CALL register. I2C Ctrl stores the received data in the RX buffer.	0x0
10	R	START_DET Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether controller is operating in Slave or Master mode.	0x0
9	R	STOP_DET Indicates whether a STOP condition has occurred on the I2C interface regardless of whether controller is operating in slave or master mode.	0x0
8	R	ACTIVITY This bit captures I2C Ctrl activity and stays set until it is cleared. There are four ways to clear it: => Disabling the I2C Ctrl => Reading the IC_CLR_ACTIVITY register => Reading the IC_CLR_INTR register => System reset When this bit is set, it stays set unless one of the four methods is used to clear it. Even if the controller module is idle, this bit remains set until cleared, indicating that there was activity on the bus.	0x0

Bit	Mode	Symbol/Description	Reset
7	R	RX_DONE When the controller is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.	0x0
6	R	TX_ABRT This bit indicates if the controller, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a "transmit abort". When this bit is set to 1, the I2C_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes place. NOTE: The controller flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register I2C_CLR_TX_ABRT is read. When this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.	0x0
5	R	RD_REQ This bit is set to 1 when I2C Ctrl is acting as a slave and another I2C master is attempting to read data from the controller. The controller holds the I2C bus in a wait state (SCL = 0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the I2C_DATA_CMD register. This bit is set to 0 just after the processor reads the I2C_CLR_RD_REQ register	0x0
4	R	TX_EMPTY This bit is set to 1 when the transmit buffer is at or below the threshold value set in the I2C_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold. When the IC_ENABLE bit 0 is 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer activity, then with ic_en=0, this bit is set to 0.	0x0
3	R	TX_OVER Set during transmit if the transmit buffer is filled to 32 and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared	0x0
2	R	RX_FULL Set when the receive buffer reaches or goes above the RX_TL threshold in the I2C_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (I2C_ENABLE[0] = 0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the I2C_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues.	0x0
1	R	RX_OVER Set if the receive buffer is completely filled to 32 and an additional byte is received from an external I2C device. The controller acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (I2C_ENABLE[0] = 0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.	0x0
0	R	RX_UNDER Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (I2C_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.	0x0

Table 230: I2C_RX_TL_REG (0x50001338)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4:0	R/W	RX_TL Receive FIFO Threshold Level Controls the level of entries (or above) that triggers the RX_FULL interrupt (bit 2 in I2C_RAW_INTR_STAT register). The valid range is 0-31, with the additional restriction that hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to do that, the actual value set is the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 31 sets the threshold for 32 entries.	0x0

Table 231: I2C_TX_TL_REG (0x5000133C)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4:0	R/W	RX_TL Transmit FIFO Threshold Level Controls the level of entries (or below) that trigger the TX_EMPTY interrupt (bit 4 in I2C_RAW_INTR_STAT register). The valid range is 0-31, with the additional restriction that it may not be set to the value larger than the depth of the buffer. If an attempt is made to do that, the actual value set is the maximum depth of the buffer. A value of 0 sets the threshold for 0 entries, and a value of 31 sets the threshold for 32 entries.	0x0

Table 232: I2C_CLR_INTR_REG (0x50001340)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_INTR Read this register to clear the combined interrupt, all individual interrupts, and the I2C_TX_ABRT_SOURCE register. This bit does not clear hardware clearable interrupts but software clearable interrupts. Refer to Bit 9 of the I2C_TX_ABRT_SOURCE register for an exception to clearing I2C_TX_ABRT_SOURCE	0x0

Table 233: I2C_CLR_RX_UNDER_REG (0x50001344)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_RX_UNDER Read this register to clear the RX_UNDER interrupt (bit 0) of the I2C_RAW_INTR_STAT register.	0x0

Table 234: I2C_CLR_RX_OVER_REG (0x50001348)

Bit	Mode	Symbol/Description	Reset
15:1	-	-	0x0

Bit	Mode	Symbol/Description	Reset
		Reserved	
0	R	CLR_RX_OVER Read this register to clear the RX_OVER interrupt (bit 1) of the I2C_RAW_INTR_STAT register.	0x0

Table 235: I2C_CLR_TX_OVER_REG (0x5000134C)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_TX_OVER Read this register to clear the TX_OVER interrupt (bit 3) of the I2C_RAW_INTR_STAT register.	0x0

Table 236: I2C_CLR_RD_REQ_REG (0x50001350)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_RD_REQ Read this register to clear the RD_REQ interrupt (bit 5) of the I2C_RAW_INTR_STAT register.	0x0

Table 237: I2C_CLR_TX_ABRT_REG (0x50001354)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_TX_ABRT Read this register to clear the TX_ABRT interrupt (bit 6) of the IC_RAW_INTR_STAT register, and the I2C_TX_ABRT_SOURCE register. This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO. See Bit 9 of the I2C_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE.	0x0

Table 238: I2C_CLR_RX_DONE_REG (0x50001358)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_RX_DONE Read this register to clear the RX_DONE interrupt (bit 7) of the I2C_RAW_INTR_STAT register.	0x0

Table 239: I2C_CLR_ACTIVITY_REG (0x5000135C)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_ACTIVITY Reading this register clears the ACTIVITY interrupt if the I2C is not active anymore. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by hardware if the module is disabled and if there is no further activity on the bus. The value read from this register to get status of the ACTIVITY interrupt (bit 8) of the IC_RAW_INTR_STAT register.	0x0

Table 240: I2C_CLR_STOP_DET_REG (0x50001360)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_STOP_DET Read this register to clear the STOP_DET interrupt (bit 9) of the IC_RAW_INTR_STAT register. Reset value: 0x0	0x0

Table 241: I2C_CLR_START_DET_REG (0x50001364)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_START_DET Read this register to clear the START_DET interrupt (bit 10) of the IC_RAW_INTR_STAT register.	0x0

Table 242: I2C_CLR_GEN_CALL_REG (0x50001368)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R	CLR_GEN_CALL Read this register to clear the GEN_CALL interrupt (bit 11) of the I2C_RAW_INTR_STAT register.	0x0

Table 243: I2C_ENABLE_REG (0x5000136C)

Bit	Mode	Symbol/Description	Reset
15:2	-	- Reserved	0x0
1	R/W	I2C_ABORT 0 = ABORT not initiated or ABORT done 1 = ABORT operation in progress	0x0

Bit	Mode	Symbol/Description	Reset
		The software can abort the I2C transfer in Master mode by setting this bit. The software can set this bit only when ENABLE is already set; otherwise, the controller ignores any write to ABORT bit. The software cannot clear the ABORT bit when set. In response to an ABORT, the controller issues a STOP and flushes the TX FIFO after completing the current transfer, then sets the TX_ABORT interrupt after the abort operation. The ABORT bit is cleared automatically after the abort operation.	
0	R/W	CTRL_ENABLE Controls whether the controller is enabled. 0: Disables the controller (TX and RX FIFOs are held in an erased state) 1: Enables the controller Software can disable the controller while it is active. However, it is important that care be taken to ensure that the controller is disabled properly. When the controller is disabled, the following occurs: * The TX FIFO and RX FIFO get flushed. * Status bits in the IC_INTR_STAT register are still active until the controller goes into IDLE state. If the module is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete. If the module is receiving, the controller stops the current transfer at the end of the current byte and does not acknowledge the transfer. There is a two ic_clk delay when enabling or disabling the controller.	0x0

Table 244: I2C_STATUS_REG (0x50001370)

Bit	Mode	Symbol/Description	Reset
15:7	-	- Reserved	0x0
6	R	SLV_ACTIVITY Slave FSM Activity Status. When the Slave Finite State Machine (FSM) is not in the IDLE state, this bit is set. 0: Slave FSM is in IDLE state so the Slave part of the controller is not Active 1: Slave FSM is not in IDLE state so the Slave part of the controller is Active	0x0
5	R	MST_ACTIVITY Master FSM Activity Status. When the Master Finite State Machine (FSM) is not in the IDLE state, this bit is set. 0: Master FSM is in IDLE state so the Master part of the controller is not Active 1: Master FSM is not in IDLE state so the Master part of the controller is Active	0x0
4	R	RFF Receive FIFO Completely Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0: Receive FIFO is not full 1: Receive FIFO is full	0x0
3	R	RFNE Receive FIFO Not Empty. This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty. 0: Receive FIFO is empty 1: Receive FIFO is not empty	0x0
2	R	TFE	0x1

Bit	Mode	Symbol/Description	Reset
		Transmit FIFO Completely Empty. When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: Transmit FIFO is not empty 1: Transmit FIFO is empty	
1	R	TFNF Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0: Transmit FIFO is full 1: Transmit FIFO is not full	0x1
0	R	I2C_ACTIVITY I2C Activity Status.	0x0

Table 245: I2C_TXFLR_REG (0x50001374)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0
5:0	R	TXFLR Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO. Size is constrained by the TXFLR value	0x0

Table 246: I2C_RXFLR_REG (0x50001378)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0
5:0	R	RXFLR Receive FIFO Level. Contains the number of valid data entries in the receive FIFO. Size is constrained by the RXFLR value.	0x0

Table 247: I2C_SDA_HOLD_REG (0x5000137C)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	IC_SDA_HOLD SDA Hold time	0x1

Table 248: I2C_TX_ABRT_SOURCE_REG (0x50001380)

Bit	Mode	Symbol/Description	Reset
15	R	ABRT_SLVRD_INTX 1: When the processor side responds to Slave mode request for data to be transmitted to a remote master and user writes 1 in CMD (bit 8) of 2IC_DATA_CMD register.	0x0
14	R	ABRT_SLV_ARBLOST 1: Slave lost the bus while transmitting data to a remote master. I2C_TX_ABRT_SOURCE[12] is set at the same time. Note: Even though the slave never "owns" the bus, something could go wrong on the bus. This is a fail safe	0x0

Bit	Mode	Symbol/Description	Reset
		check. For instance, during a data transmission at the low-to-high transition of SCL, if what is on the data bus is not what is supposed to be transmitted, then the controller no longer owns the bus.	
13	R	ABRT_SLVFLUSH_TXFIFO 1: Slave has received a read command and some data exists in the TX FIFO so the slave issues a TX_ABRT interrupt to flush old data in TX FIFO.	0x0
12	R	ARB_LOST 1: Master has lost arbitration, or if I2C_TX_ABRT_SOURCE[14] is also set, then the slave transmitter has lost arbitration. Note: I2C can be both master and slave at the same time.	0x0
11	R	ABRT_MASTER_DIS 1: User tries to initiate a Master operation with the Master mode disabled.	0x0
10	R	ABRT_10B_RD_NORSTRT 1: The restart is disabled (IC_RESTART_EN bit (I2C_CON[5]) = 0) and the master sends a read command in 10-bit addressing mode.	0x0
9	R	ABRT_SBYTE_NORSTRT To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; restart must be enabled (I2C_CON[5]=1), the SPECIAL bit must be cleared (I2C_TAR[11]), or the GC_OR_START bit must be cleared (I2C_TAR[10]). When the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, bit 9 clears for one cycle and then gets re-asserted. 1: The restart is disabled (IC_RESTART_EN bit (I2C_CON[5]) = 0) and the user is trying to send a START Byte.	0x0
8	R	ABRT_HS_NORSTRT 1: The restart is disabled (IC_RESTART_EN bit (I2C_CON[5]) = 0) and the user is trying to use the master to transfer data in High Speed mode.	0x0
7	R	ABRT_SBYTE_ACKDET 1: Master has sent a START Byte and the START Byte was acknowledged (wrong behavior).	0x0
6	R	ABRT_HS_ACKDET 1: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior).	0x0
5	R	ABRT_GCALL_READ 1: The controller in master mode sent a General Call but the user programmed the byte following the General Call to be a read from the bus (IC_DATA_CMD[9] is set to 1).	0x0
4	R	ABRT_GCALL_NOACK 1: The controller in master mode sent a General Call and no slave on the bus acknowledged the General Call.	0x0
3	R	ABRT_TXDATA_NOACK 1: This is a master-mode only bit. Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledgement from the remote slave(s).	0x0
2	R	ABRT_10ADDR2_NOACK 1: Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave.	0x0
1	R	ABRT_10ADDR1_NOACK	0x0

Bit	Mode	Symbol/Description	Reset
		1: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave.	
0	R	ABRT_7B_ADDR_NOACK 1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave.	0x0

Table 249: I2C_DMA_CR_REG (0x50001388)

Bit	Mode	Symbol/Description	Reset
1	R/W	TDMAE Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. 0 = Transmit DMA disabled 1 = Transmit DMA enabled	0x0
0	R/W	RDMAE Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel. 0 = Receive DMA disabled 1 = Receive DMA enabled	0x0

Table 250: I2C_DMA_TDLR_REG (0x5000138C)

Bit	Mode	Symbol/Description	Reset
4:0	R/W	DMATDL Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1.	0x0

Table 251: I2C_DMA_RDLR_REG (0x50001390)

Bit	Mode	Symbol/Description	Reset
4:0	R/W	DMARDL Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or more than this field value + 1, and RDMAE =1. For instance, when DMARDL is 0, then dma_rx_req is asserted when 1 or more data entries are present in the receive FIFO.	0x0

Table 252: I2C_SDA_SETUP_REG (0x50001394)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SDA_SETUP SDA Setup. This register controls the amount of time delay (number of I2C clock periods) between the rising edge of SCL and SDA changing by holding SCL low when I2C block services a read request while operating as a slave-transmitter. The relevant I2C requirement is tSU:DAT (note 4) as detailed in the I2C Bus Specification. This register must be programmed with a value equal to or greater than 2.	0x64

Bit	Mode	Symbol/Description	Reset
		It is recommended that if the required delay is 1000 ns, then for an I2C frequency of 10 MHz, IC_SDA_SETUP should be programmed to a value of 11. Writes to this register succeed only when IC_ENABLE[0] = 0.	

Table 253: I2C_ACK_GENERAL_CALL_REG (0x50001398)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	ACK_GEN_CALL ACK General Call. When set to 1, I2C Ctrl responds with a ACK (by asserting ic_data_oe) when it receives a General Call. When set to 0, the controller does not generate General Call interrupts.	0x0

Table 254: I2C_ENABLE_STATUS_REG (0x5000139C)

Bit	Mode	Symbol/Description	Reset
15:3	-	- Reserved	0x0
2	R	SLV_RX_DATA_LOST Slave Received Data Lost. This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to the setting of IC_ENABLE from 1 to 0. When read as 1, the controller is deemed to have been actively engaged in an aborted I2C transfer (with matching address) and the data phase of the I2C transfer has been entered, even though a data byte has been responded with a NACK. NOTE: If the remote I2C master terminates the transfer with a STOP condition before the controller has a chance to NACK a transfer, and IC_ENABLE has been set to 0, then this bit is also set to 1. When read as 0, the controller is deemed to have been disabled without being actively involved in the data phase of a Slave-Receiver transfer. NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0.	0x0
1	R	SLV_DISABLED_WHILE_BUSY Slave Disabled While Busy (Transmit, Receive). This bit indicates if a potential or active Slave operation has been aborted due to the setting of the IC_ENABLE register from 1 to 0. This bit is set when the CPU writes a 0 to the IC_ENABLE register while: (a) I2C Ctrl is receiving the address byte of the Slave-Transmitter operation from a remote master; OR, (b) address and data bytes of the Slave-Receiver operation from a remote master. When read as 1, the controller is deemed to have forced a NACK during any part of an I2C transfer, irrespective of whether the I2C address matches the slave address set in I2C Ctrl (IC_SAR register) OR if the transfer is completed before IC_ENABLE is set to 0 but has not taken effect. NOTE: If the remote I2C master terminates the transfer with a STOP condition before the the controller has a chance to NACK a transfer, and IC_ENABLE has been set to 0, then this bit will also be set to 1. When read as 0, the controller is deemed to have been disabled when there is master activity, or when the I2C bus is idle. NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0.	0x0
0	R	IC_EN ic_en Status. This bit always reflects the value driven on the output port ic_en. When read as 1, the controller is deemed to be in an enabled state.	0x0

Bit	Mode	Symbol/Description	Reset
		When read as 0, the controller is deemed completely inactive. NOTE: The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read SLV_RX_DATA_LOST (bit 2) and SLV_DISABLED_WHILE_BUSY (bit 1).	

Table 255: I2C_IC_FS_SPKLEN_REG (0x500013A0)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	IC_FS_SPKLEN This register must be set before any I2C bus transaction can take place to ensure stable operation. This register sets the duration, measured in ic_clk cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than this being written, and if attempted results in 1 being set.	0x1

31.10 Keyboard Registers

Table 256: Register map KBRD

Address	Register	Description
0x50001400	GPIO_IRQ0_IN_SEL_REG	GPIO interrupt selection for GPIO_IRQ0
0x50001402	GPIO_IRQ1_IN_SEL_REG	GPIO interrupt selection for GPIO_IRQ1
0x50001404	GPIO_IRQ2_IN_SEL_REG	GPIO interrupt selection for GPIO_IRQ2
0x50001406	GPIO_IRQ3_IN_SEL_REG	GPIO interrupt selection for GPIO_IRQ3
0x50001408	GPIO_IRQ4_IN_SEL_REG	GPIO interrupt selection for GPIO_IRQ4
0x5000140c	GPIO_DEBOUNCE_REG	Debounce counter value for GPIO inputs
0x5000140e	GPIO_RESET_IRQ_REG	GPIO interrupt reset register
0x50001410	GPIO_INT_LEVEL_CTRL_REG	High or low level select for GPIO interrupts
0x50001412	KBRD_IRQ_IN_SEL0_REG	GPIO interrupt selection for KBRD_IRQ for P0
0x50001414	KBRD_CTRL_REG	GPIO KBRD control register

Table 257: GPIO_IRQ0_IN_SEL_REG (0x50001400)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0
3:0	R/W	KBRD_IRQ0_SEL Input selection that can generate a GPIO interrupt 1: P0[0] is selected 2: P0[1] is selected 3: P0[2] is selected 4: P0[3] is selected 5: P0[4] is selected 6: P0[5] is selected 7: P0[6] is selected 8: P0[7] is selected 9: P0[8] is selected 10: P0[9] is selected 11: P0[10] is selected 12: P0[11] is selected all others: no input selected	0x0

Table 258: GPIO_IRQ1_IN_SEL_REG (0x50001402)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0

Bit	Mode	Symbol/Description	Reset
3:0	R/W	KBRD_IRQ1_SEL See KBRD_IRQ0_SEL	0x0

Table 259: GPIO_IRQ2_IN_SEL_REG (0x50001404)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0
3:0	R/W	KBRD_IRQ2_SEL See KBRD_IRQ0_SEL	0x0

Table 260: GPIO_IRQ3_IN_SEL_REG (0x50001406)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0
3:0	R/W	KBRD_IRQ3_SEL See KBRD_IRQ0_SEL	0x0

Table 261: GPIO_IRQ4_IN_SEL_REG (0x50001408)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0
3:0	R/W	KBRD_IRQ4_SEL See KBRD_IRQ0_SEL	0x0

Table 262: GPIO_DEBOUNCE_REG (0x5000140C)

Bit	Mode	Symbol/Description	Reset
11	R/W	DEB_ENABLE_KBRD Enables the debounce counter for the KBRD interface	0x0
10	R/W	DEB_ENABLE4 Enables the debounce counter for GPIO IRQ4	0x0
9	R/W	DEB_ENABLE3 Enables the debounce counter for GPIO IRQ3	0x0
8	R/W	DEB_ENABLE2 Enables the debounce counter for GPIO IRQ2	0x0
7	R/W	DEB_ENABLE1 Enables the debounce counter for GPIO IRQ1	0x0
6	R/W	DEB_ENABLE0 Enables the debounce counter for GPIO IRQ0	0x0
5:0	R/W	DEB_VALUE Keyboard debounce time if enabled. Generate KEYB_INT after specified time.	0x0

Bit	Mode	Symbol/Description	Reset
		Debounce time: N*1 ms. N =0..63	

Table 263: GPIO_RESET_IRQ_REG (0x5000140E)

Bit	Mode	Symbol/Description	Reset
15:6	-	- Reserved	0x0
5	R0/W	RESET_KBRD_IRQ Writing 1 to this bit resets the KBRD IRQ. Reading returns 0.	0x0
4	R0/W	RESET_GPIO4_IRQ Writing 1 to this bit resets the GPIO4 IRQ. Reading returns 0.	0x0
3	R0/W	RESET_GPIO3_IRQ Writing 1 to this bit resets the GPIO3 IRQ. Reading returns 0.	0x0
2	R0/W	RESET_GPIO2_IRQ Writing 1 to this bit resets the GPIO2 IRQ. Reading returns 0.	0x0
1	R0/W	RESET_GPIO1_IRQ Writing 1 to this bit resets the GPIO1 IRQ. Reading returns 0.	0x0
0	R0/W	RESET_GPIO0_IRQ Writing 1 to this bit resets the GPIO0 IRQ. Reading returns 0.	0x0

Table 264: GPIO_INT_LEVEL_CTRL_REG (0x50001410)

Bit	Mode	Symbol/Description	Reset
9	R/W	EDGE_LEVELn4 See EDGE_LEVELn0, but for GPIO IRQ4	0x0
8	R/W	EDGE_LEVELn3 See EDGE_LEVELn0, but for GPIO IRQ3	0x0
7	R/W	EDGE_LEVELn2 See EDGE_LEVELn0, but for GPIO IRQ2	0x0
6	R/W	EDGE_LEVELn1 See EDGE_LEVELn0, but for GPIO IRQ1	0x0
5	R/W	EDGE_LEVELn0 0: Do not wait for key release after interrupt was reset for GPIO IRQ0, so a new interrupt can be initiated immediately 1: Wait for key release after interrupt was reset for IRQ0	0x0
4	R/W	INPUT_LEVEL4 See INPUT_LEVEL0, but for GPIO IRQ4	0x0
3	R/W	INPUT_LEVEL3	0x0

Bit	Mode	Symbol/Description	Reset
		See INPUT_LEVEL0, but for GPIO IRQ3	
2	R/W	INPUT_LEVEL2 See INPUT_LEVEL0, but for GPIO IRQ2	0x0
1	R/W	INPUT_LEVEL1 See INPUT_LEVEL0, but for GPIO IRQ1	0x0
0	R/W	INPUT_LEVEL0 0 = Selected input generates GPIO IRQ0 if that input is high 1 = Selected input generates GPIO IRQ0 if that input is low	0x0

Table 265: KBRD_IRQ_IN_SEL0_REG (0x50001412)

Bit	Mode	Symbol/Description	Reset
11	R/W	KBRD_P11_EN Enable P0[11] for the keyboard interrupt	0x0
10	R/W	KBRD_P10_EN Enable P0[10] for the keyboard interrupt	0x0
9	R/W	KBRD_P09_EN Enable P0[9] for the keyboard interrupt	0x0
8	R/W	KBRD_P08_EN Enable P0[8] for the keyboard interrupt	0x0
7	R/W	KBRD_P07_EN Enable P0[7] for the keyboard interrupt	0x0
6	R/W	KBRD_P06_EN Enable P0[6] for the keyboard interrupt	0x0
5	R/W	KBRD_P05_EN Enable P0[5] for the keyboard interrupt	0x0
4	R/W	KBRD_P04_EN Enable P0[4] for the keyboard interrupt	0x0
3	R/W	KBRD_P03_EN Enable P0[3] for the keyboard interrupt	0x0
2	R/W	KBRD_P02_EN Enable P0[2] for the keyboard interrupt	0x0
1	R/W	KBRD_P01_EN Enable P0[1] for the keyboard interrupt	0x0
0	R/W	KBRD_P00_EN Enable P0[0] for the keyboard interrupt	0x0

Table 266: KBRD_CTRL_REG (0x50001414)

Bit	Mode	Symbol/Description	Reset
7	R/W	KBRD_REL 0 = No interrupt on key release 1 = Interrupt also on key release (also debouncing if enabled)	0x0

Bit	Mode	Symbol/Description	Reset
6	R/W	KBRD_LEVEL 0 = Enabled input generates KBRD IRQ if that input is high 1 = Enabled input generates KBRD IRQ if that input is low	0x0
5:0	R/W	KEY_REPEAT While key is pressed, automatically generate repeating KEYB_INT after specified time unequal to 0. Repeat time: N*1 ms. N = 1..63, N = 0 disables the timer.	0x0

31.11 Miscellaneous Registers

Table 267: Register map CRG_AON

Address	Register	Description
0x50000300	HWR_CTRL_REG	Hardware Reset control register
0x50000304	RESET_STAT_REG	Reset status register
0x50000308	RAM_LPMX_REG	
0x5000030c	PAD_LATCH_REG	Control the state retention of the GPIO ports
0x50000310	HIBERN_CTRL_REG	Hibernation control register
0x50000314	PULL_HW_BYPASS_REG	Control the muxes for P0_1 pull up and down
0x50000320	POWER_AON_CTRL_REG	
0x50000324	GP_DATA_REG	

Table 268: HWR_CTRL_REG (0x50000300)

Bit	Mode	Symbol/Description	Reset
0	R/W	DISABLE_HWR Disables the RST functionality on P00	0x0

Table 269: RESET_STAT_REG (0x50000304)

Bit	Mode	Symbol/Description	Reset
3	R/W	WDOGRESET_STAT Indicates that a Watchdog has happened. This bit is also set with a PowerOn Reset.	0x1
2	R/W	SWRESET_STAT Indicates that a software reset has been requested. The software reset is requested by SYS_CTRL_REG[SW_RESET] or SCB->AIRCR inside the Cortex. This bit is also set with a PowerOn Reset.	0x1
1	R/W	HWRESET_STAT Indicates that a hardware reset has happened. This bit is also set with a PowerOn Reset.	0x1
0	R/W	PORESET_STAT Indicates that a PowerOn Reset has happened.	0x1

Table 270: RAM_LPMX_REG (0x50000308)

Bit	Mode	Symbol/Description	Reset
2:0	R/W	RAMx_LPMX RAM[3:1] Transparent Light Sleep (TLS) Core Enable for System RAMs. Assert low to enable the TLS core feature, which will result in lower leakage current. In case VDD is below 0.81 V, it is necessary to hold this pin high to maintain data retention.	0x7

Table 271: PAD_LATCH_REG (0x5000030C)

Bit	Mode	Symbol/Description	Reset
0	R/W	PAD_LATCH_EN Controls the state retention of the pads. 0: Latches are closed, pads retain their state. 1: Latches are open, new control values have immediate effect.	0x1

Table 272: HIBERN_CTRL_REG (0x50000310)

Bit	Mode	Symbol/Description	Reset
6:2	R/W	HIBERN_WKUP_MASK Selects which pin to wake up from.	0x0
1	R/W	HIBERN_WKUP_POLARITY Selects the polarity of the wake-up source. The polarity must be chosen such that ANA_STATUS_REG[CLKLESS_WAKEUP_STAT] is 1. Any change on the selected GPIOs makes the CLKLESS_WAKEUP_STAT go to 0, and wakes up the system from hibernation.	0x0
0	R/W	HIBERNATION_ENABLE Enables the hibernation mode when sleeping 0: Deep Sleep mode, PD_SLP remains on 1: Hibernation mode, PD_SLP goes off. REMAP_ADR0 needs to be set to the correct source to boot from before going to sleep.	0x0

Table 273: PULL_HW_BYPASS_REG (0x50000314)

Bit	Mode	Symbol/Description	Reset
0	R/W	PULL_HW_BYPASS 0: Normal operation 1: Force pull-up registor at GPIO P0_1 in boost mode	0x1

Table 274: POWER_AON_CTRL_REG (0x50000320)

Bit	Mode	Symbol/Description	Reset
14	R/W	- Reserved	0x0
13:10	R/W	LDO_RET_TRIM VDD clamp level setting for Hibernation mode.	0x0
9	R/W	CMP_VCONT_SLP_DISABLE Disables vcont comparator in SLP.	0x0
8:7	R/W	BOOST_MODE_FORCE 0x: Automatic selection of boost mode 11: Force boost mode 10: Force buck mode	0x0
6	R/W	CHARGE_VBAT_DISABLE Do not charge vbat high in boost mode	0x0
5	R/W	- Reserved	0x0

Bit	Mode	Symbol/Description	Reset
4	R/W	- Reserved	0x0
3	R/W	POR_VBAT_HIGH_RST_MASK Mask rst from por_vbat_high	0x1
2	R/W	POR_VBAT_LOW_RST_MASK Mask rst from por_vbat_low	0x0
1:0	R/W	VBAT_HL_CONNECT_RES_CTRL 00: OFF 01: Forced ON 10: Active: automatic control, Sleep: forced ON 11: Automatic control	0x0

Table 275: GP_DATA_REG (0x50000324)

Bit	Mode	Symbol/Description	Reset
7	R/W	P03_P04_FILT_DIS 0: RC filtered input enabled for P0_3 and P0_4 (for example, when used for wake-up) 1: RC filtered input disabled for P0_3 and P0_4 (for example, when used for external clk or XTAL32k)	0x0
6	R/W	FORCE_RCX_VDD 0: RCX bias supply open (see FORCE_RCX_VREF) 1: RCX bias supply connected to VDD (use for sleep)	0x0
5	R/W	FORCE_RCX_VREF 0: RCX bias supply connected to clamp and VDD via 400k resistor (old situation) 1: RCX bias supply connected to vref_0v75_0 (use for calibration)	0x0
4	R/W	- Reserved	0x0
3:0	R/W	SW_GP_DATA	0x0

31.12 OTP Controller Registers

Table 276: Register map OTPC

Address	Register	Description
0x07f40000	OTPC_MODE_REG	Mode register
0x07f40004	OTPC_STAT_REG	Status register
0x07f40008	OTPC_PADDR_REG	The address of the word that will be programmed, when the PROG mode is used.
0x07f4000c	OTPC_PWORD_REG	The 32-bit word that will be programmed, when the PROG mode is used.
0x07f40010	OTPC_TIM1_REG	Various timing parameters of the OTP cell.
0x07f40014	OTPC_TIM2_REG	Various timing parameters of the OTP cell.
0x07f40018	OTPC_AHBADR_REG	AHB master start address
0x07f4001c	OTPC_CELADR_REG	OTP cell start address
0x07f40020	OTPC_NWORDS_REG	Number of words

Table 277: OTPC_MODE_REG (0x07F40000)

Bit	Mode	Symbol/Description	Reset
31:9	-	- Reserved	0x0
8:6	R/W	OTPC_MODE_PRG_SEL Defines the part of the OTP cell that is programmed by the controller during the PROG mode, for each program request that is applied. 0x0: Both normal and redundancy arrays are programmed. This is the normal way of programming. 0x1: Only the normal array is programmed. 0x2: Only the redundancy array 0 is programmed. 0x3: Only the redundancy array 1 is programmed. 0x4: Only the redundancy array 2 is programmed. 0x5-0x7: Reserved The value of this configuration field can be modified only when the controller is in inactive mode (DSTBY or STBY). The setting takes effect when the PROG mode is enabled again.	0x0
5	R/W	OTPC_MODE_HT_MARG_EN Defines the temperature condition under which is performed a margin read. It affects only the initial margin read (RINI mode) and the programming verification margin read (PVFY). 0: Regular temperature condition (less than 85 °C) 1: High temperature condition (85 °C or more) The value of this configuration field can be modified only when the controller is in inactive mode (DSTBY or STBY). The selection takes effect at the next PVFY or RINI mode that will be enabled. The READ mode is not affected by the setting of this configuration bit.	0x0
4	R/W	OTPC_MODE_USE_TST_ROW Selects the memory area of the OTP cell that will be used. 0: Uses the main memory area of the OTP cell 1: Uses the test row of the OTP cell	0x0

Bit	Mode	Symbol/Description	Reset
		The value of this configuration field can be modified only when the controller is in inactive mode (DSTBY or STBY). The selection takes effect at the next programming or reading mode that will be enabled.	
3	-	- Reserved	0x0
2:0	R/W	OTPC_MODE_MODE Defines the mode of operation of the OTPC controller. The encoding of the modes is as follows: 0x0: DSTBY. The OTP memory is in deep standby mode (power supply ON and internal LDO OFF). 0x1: STBY. The OTP memory is powered (power supply ON and internal LDO ON, but is not selected). 0x2: READ. The OTP memory is in normal read mode. 0x3: PROG. The OTP memory is in programming mode. 0x4: PVFY. The OTP memory is in programming verification mode (margin read after programming). 0x5: RINI. The OTP memory is in initial read mode (initial margin read). 0x6: AREAD. Copying of data from the OTP memory to a system RAM by using the internal DMA. Also, see the registers OTPC_AHBADR_REG, OTPC_CELADR_REG and OTPC_NWORDS_REG. 0x7: Reserved Whenever the OTPC_MODE_REG[MODE] is changing, the status bit OTPC_STAT_REG[OTPC_STAT_MRDY] gets the value zero. The new mode is ready for use when OTPC_STAT_MRDY becomes again 1. During the mode transition the OTPC_MODE_REG[MODE] becomes read only. Do not try to use or change any function of the controller until the OTPC_STAT_MRDY bit becomes equal to 1.	0x0

Table 278: OTPC_STAT_REG (0x07F40004)

Bit	Mode	Symbol/Description	Reset
31:3	-	- Reserved	0x0
2	R	OTPC_STAT_MRDY Indicates the progress of the transition from a mode of operation to a new mode of operation. 0: There is a transition in progress in a new mode of operation. Wait until the transition to be completed. 1: The transition to the new mode of operation has been completed. The function that has been enabled by the new mode can be used. A new mode can be applied. This status bit gets the value zero every time where OTPC_MODE_REG[MODE] is changing. Do not try to use or change any function of the controller until this status bit becomes equal to 1.	0x1
1	R	OTPC_STAT_PBUF_EMPTY Indicates the status of the programming buffer (PBUF). 0: The PBUF contains the address and the data of a programming request. The OTPC_PADDR_REG and the OTPC_PWORD_REG should not be written as long as this status bit is zero. 1: The PBUF is empty and a new programming request can be registered in the PBUF by using the OTPC_PADDR_REG and the OTPC_PWORD_REG registers.	0x1

Bit	Mode	Symbol/Description	Reset
		This status bit gets the value zero every time when a programming is triggered by the OTPC_PADDR_REG (only if the PROG mode is active).	
0	R	OTPC_STAT_PRDY Indicates the state of the programming process. 0: The controller is busy. A programming is in progress. 1: The logic which performs programming is idle.	0x1

Table 279: OTPC_PADDR_REG (0x07F40008)

Bit	Mode	Symbol/Description	Reset
31:12	-	- Reserved	0x0
11:0	R/W	OTPC_PADDR The OTPC_PADDR_REG and the OTPC_PWORD_REG consist of the PBUF buffer that keeps the information that will be programmed in the OTP by using the PROG mode. The PBUF holds the address (OTPC_PADDR_REG) and the data (OTPC_PWORD_REG) of each of the programming requests that are applied in the OTP memory. The OTPC_PADDR_REG refers to a word address. The OTPC_PADDR_REG has to be writen after the OTP_PWORD_REG and only if the OTPC_STAT_REG[OTPC_STAT_PBUF_EMPTY] = 1. The register is read only for as long the PBUF is not empty (OTPC_STAT_REG[OTPC_STAT_PBUF_EMPTY] = 0). A writing to the OTPC_PADDR_REG triggers the controller to start the programming procedure (only if the PROG mode is active). The maximum valid address is 3071.	0x0

Table 280: OTPC_PWORD_REG (0x07F4000C)

Bit	Mode	Symbol/Description	Reset
31:0	R/W	OTPC_PWORD The OTPC_PADDR_REG and the OTPC_PWORD_REG consist of the PBUF buffer that keeps the information that will be programmed in the OTP memory by using the PROG mode. The PBUF holds the address (OTPC_PADDR_REG) and the data (OTPC_PWORD_REG) of each of the programming requests that are applied in the OTP memory. The OTP_PWORD_REG must be written before the OTPC_PADDR_REG and only if OTPC_STAT_REG[OTPC_STAT_PBUF_EMPTY] = 1. The register is read only for as long the PBUF is not empty (OTPC_STAT_REG[OTPC_STAT_PBUF_EMPTY] = 0).	0x0

Table 281: OTPC_TIM1_REG (0x07F40010)

Bit	Mode	Symbol/Description	Reset
31	-	- Reserved	0x0
30:24	R/W	OTPC_TIM1_US_T_CSP The number of microseconds (minus one) that are required after the selection of the OTP memory, until to be ready for programming. It must be: - at least 10 μs - no more than 100 μs	0x9

Bit	Mode	Symbol/Description	Reset
23:20	R/W	OTPC_TIM1_US_T_CS The number of microseconds (minus one) that are required after the selection of the OTP memory, until to be ready for any kind of read. It must be at least 10 µs.	0x9
19:16	R/W	OTPC_TIM1_US_T_PL The number of microseconds (minus one) that are required until to be enabled the LDO of the OTP. It must be at least 10 µs.	0x9
15:12	R/W	OTPC_TIM1_CC_T_RD The number of hclk_c clock periods (minus one) that give a time interval at least higher than 120 ns. This timing parameter refers to the access time of the OTP memory.	0x1
11	-	- Reserved	0x0
10:8	R/W	OTPC_TIM1_CC_T_20NS The number of hclk_c clock periods (minus one) that give a time interval that is at least higher than 20 ns.	0x0
7:0	R/W	OTPC_TIM1_CC_T_1US The number of hclk_c clock periods (minus one) that give a time interval equal to 1 µs. This setting affects all the timing parameters that refer to microseconds, due to that defines the correspondence of a microsecond to a number of hclk_c clock cycles.	0xF

Table 282: OTPC_TIM2_REG (0x07F40014)

Bit	Mode	Symbol/Description	Reset
31	R/W	OTPC_TIM2_US_ADD_CC_EN Adds an additional hclk_c clock cycle at all the time intervals that count in microseconds. 0: The extra hclk_c clock cycle is not applied 1: The extra hclk_c clock cycle is applied	0x1
30:29	R/W	OTPC_TIM2_US_T_SAS The number of microseconds (minus one) that are required after the exit from the deep sleep standby mode and before becoming ready to enter active mode (reading or programming). It must be at least 2 µs.	0x1
28:24	R/W	OTPC_TIM2_US_T_PPH The number of microseconds (minus one) that are required after the last programming pulse and before the programming mode is disabled in the OTP memory. It must be: - at least 5 µs - no more than 20 µs	0x4
23:21	R/W	OTPC_TIM2_US_T_VDS The number of microseconds (minus one) that are required after enabling the power supply of the OTP memory and before becoming ready for enabling of the internal LDO. It must be at least 1 µs.	0x0
20:16	R/W	OTPC_TIM2_US_T_PPS The number of microseconds (minus one) that are required after enabling the programming in the OTP memory and before the first programming pulse is applied. It must be: - at least 5 µs	0x4

Bit	Mode	Symbol/Description	Reset
		- no more than 20 µs	
15	-	- Reserved	0x0
14:8	R/W	OTPC_TIM2_US_T_PPR The number of microseconds (minus one) for recovery after a programming sequence. It must be: - at least 5 µs - no more than 100 µs	0x4
7:5	R/W	OTPC_TIM2_US_T_PWI The number of microseconds (minus one) between two consecutive programming pulses. It must be: - at least 1 µs - no more than 5 µs	0x0
4:0	R/W	OTPC_TIM2_US_T_PW The number of microseconds (minus one) that lasts the programming of each bit. It must be: - at least 10 µs - no more than 20 µs	0x9

Table 283: OTPC_AHBADR_REG (0x07F40018)

Bit	Mode	Symbol/Description	Reset
31:16	-	- Reserved	0x0
15:2	R/W	OTPC_AHBADR It is the AHB address used by the AHB master interface of the controller (the bits [15:2]). The bits [1:0] of the address are always considered as equal to zero. The value of the register remains unchanged, by the internal logic of the controller.	0x0
1:0	-	- Reserved	0x0

Table 284: OTPC_CELADR_REG (0x07F4001C)

Bit	Mode	Symbol/Description	Reset
31:12	-	- Reserved	0x0
11:0	R/W	OTPC_CELADR Defines a word address inside the OTP cell that will be used during the AREAD mode and the OTP mirroring. The last valid address is 3071.	0x0

Table 285: OTPC_NWORDS_REG (0x07F40020)

Bit	Mode	Symbol/Description	Reset
31:12	-	- Reserved	0x0
11:0	R/W	OTPC_NWORDS	0x0

Bit	Mode	Symbol/Description	Reset
		The number of words (minus one) that will be copied by the AREAD mode. During mirroring, this register reflects the amount of data that will be copied. The maximum valid setting is 3071 (3072 words).	

31.13 Quadrature Decoder Registers

Table 286: Register map QDEC

Address	Register	Description
0x50000200	QDEC_CTRL_REG	Quad Decoder control register
0x50000202	QDEC_XCNT_REG	Counter value of the X Axis
0x50000204	QDEC_YCNT_REG	Counter value of the Y Axis
0x50000206	QDEC_CLOCKDIV_REG	Clock divider register
0x50000208	QDEC_CTRL2_REG	Quad Decoder port selection register
0x5000020a	QDEC_ZCNT_REG	Counter value of the Z Axis
0x5000020c	QDEC_EVENT_CNT_REG	Event counter register

Table 287: QDEC_CTRL_REG (0x50000200)

Bit	Mode	Symbol/Description	Reset
10:3	R/W	QDEC_IRQ_THRES Defines the number of events on either counter (X or Y or Z) that need to be reached before an interrupt is generated. Events are equal to QDEC_IRQ_THRES+1.	0x2
2	R/W	QDEC_IRQ_STATUS 1 = Interrupt has occurred 0 = No interrupt pending Write 1 will clear the pending interrupt	0x0
1	R0/WC	QDEC_EVENT_CNT_CLR Writing 1 QDEC_EVENT_CNT_REG is cleared	0x0
0	R/W	QDEC_IRQ_ENABLE 0 = Interrupt is masked 1 = Interrupt is enabled	0x1

Table 288: QDEC_XCNT_REG (0x50000202)

Bit	Mode	Symbol/Description	Reset
15:0	R	QDEC_X_CNT Contains a signed value of the events. Zero when channel is disabled	0x0

Table 289: QDEC_YCNT_REG (0x50000204)

Bit	Mode	Symbol/Description	Reset
15:0	R	QDEC_Y_CNT Contains a signed value of the events. Zero when channel is disabled	0x0

Table 290: QDEC_CLOCKDIV_REG (0x50000206)

Bit	Mode	Symbol/Description	Reset
10	R/W	QDEC_PRESCALER_EN 0 = No prescaler enabled	0x0

Bit	Mode	Symbol/Description	Reset
		1 = In Sleep and Active mode, quadrature clock is divided by 2	
9:0	R/W	QDEC_CLOCKDIV Contains the number of the input clock cycles minus one, that are required to generate one logic clock cycle. Clock divider is bypassed when system runs at LP_CLK	0x3E7

Table 291: QDEC_CTRL2_REG (0x50000208)

Bit	Mode	Symbol/Description	Reset
11	R/W	QDEC_CHZ_EVENT_MODE 0 = Normal quadrature counting 1 = Counts rising and falling edge of both ports (if both ports change at the same time, counter increases by 1)	0x1
10	R/W	QDEC_CHY_EVENT_MODE 0 = Normal quadrature counting 1 = Counts rising and falling edge of both ports (if both ports change at the same time, counter increases by 1)	0x1
9	R/W	QDEC_CHX_EVENT_MODE 0 = Normal quadrature counting 1 = Counts rising and falling edge of both ports (if both ports change at the same time, counter increases by 1)	0x1
8:6	R/W	QDEC_CHZ_PORT_SEL Defines which GPIOs are mapped on Channel Z 0: none 1: P0[2] -> CHZ_A, P0[5] -> CHZ_B 2: P0[1] -> CHZ_A, P0[4] -> CHZ_B 3: P0[3] -> CHZ_A, P0[10] -> CHZ_B 4: P0[6] -> CHZ_A, P0[7] -> CHZ_B 5: P0[8] -> CHZ_A, P0[9] -> CHZ_B 6: P0[0] -> CHZ_A, P0[11] -> CHZ_B 7: none	3
5:3	R/W	QDEC_CHY_PORT_SEL Defines which GPIOs are mapped on Channel Y 0: none 1: P0[2] -> CHY_A, P0[5] -> CHY_B 2: P0[1] -> CHY_A, P0[4] -> CHY_B 3: P0[3] -> CHY_A, P0[10] -> CHY_B 4: P0[6] -> CHY_A, P0[7] -> CHY_B 5: P0[8] -> CHY_A, P0[9] -> CHY_B 6: P0[0] -> CHY_A, P0[11] -> CHY_B 7: none	2
2:0	R/W	QDEC_CHX_PORT_SEL Defines which GPIOs are mapped on Channel X 0: none 1: P0[2] -> CHX_A, P0[5] -> CHX_B 2: P0[1] -> CHX_A, P0[4] -> CHX_B 3: P0[3] -> CHX_A, P0[10] -> CHX_B	1

Bit	Mode	Symbol/Description	Reset
		4: P0[6] -> CHX_A, P0[7] -> CHX_B 5: P0[8] -> CHX_A, P0[9] -> CHX_B 6: P0[0] -> CHX_A, P0[11] -> CHX_B 7: none	

Table 292: QDEC_ZCNT_REG (0x5000020A)

Bit	Mode	Symbol/Description	Reset
15:0	R	QDEC_Z_CNT Contains a signed value of the events. Zero when channel is disabled	0

Table 293: QDEC_EVENT_CNT_REG (0x5000020C)

Bit	Mode	Symbol/Description	Reset
7:0	R	QDEC_EVENT_CNT Gives the number of events at all channels.	0x0

31.14 Real Time Clock Registers

Table 294: Register map RTC

Address	Register	Description
0x50004100	RTC_CONTROL_REG	RTC Control Register
0x50004104	RTC_HOUR_MODE_REG	RTC Hour Mode Register
0x50004108	RTC_TIME_REG	RTC Time Register
0x5000410c	RTC_CALENDAR_REG	RTC Calendar Register
0x50004110	RTC_TIME_ALARM_REG	RTC Time Alarm Register
0x50004114	RTC_CALENDAR_ALARM_REG	RTC Calendar Alram Register
0x50004118	RTC_ALARM_ENABLE_REG	RTC Alarm Enable Register
0x5000411c	RTC_EVENT_FLAGS_REG	RTC Event Flags Register
0x50004120	RTC_INTERRUPT_ENABLE_REG	RTC Interrupt Enable Register
0x50004124	RTC_INTERRUPT_DISABLE_REG	RTC Interrupt Disable Register
0x50004128	RTC_INTERRUPT_MASK_REG	RTC Interrupt Mask Register
0x5000412c	RTC_STATUS_REG	RTC Status Register
0x50004130	RTC_KEEP_RTC_REG	RTC Keep RTC Register

Table 295: RTC_CONTROL_REG (0x50004100)

Bit	Mode	Symbol/Description	Reset
1	R/W	RTC_CAL_DISABLE When this field is set high the RTC stops incrementing the calendar value.	0x1
0	R/W	RTC_TIME_DISABLE When this field is set high the RTC stops incrementing the time value.	0x1

Table 296: RTC_HOUR_MODE_REG (0x50004104)

Bit	Mode	Symbol/Description	Reset
0	R/W	RTC_HMS When this field is set high the RTC operates in 12-hour clock mode; otherwise, times are in 24-hour clock format.	0x0

Table 297: RTC_TIME_REG (0x50004108)

Bit	Mode	Symbol/Description	Reset
31	R/W	RTC_TIME_CH The value in this register has altered since last read. Read and clear.	0x0
30	R/W	RTC_TIME_PM In 12-hour clock mode, indicates PM when set.	0x0

Bit	Mode	Symbol/Description	Reset
29:28	R/W	RTC_TIME_HR_T Hours tens. Represented in BCD digit (0-2).	0x0
27:24	R/W	RTC_TIME_HR_U Hours units. Represented in BCD digit (0-9).	0x0
23	-	- Reserved	0x0
22:20	R/W	RTC_TIME_M_T Minutes tens. Represented in BCD digit (0-5).	0x0
19:16	R/W	RTC_TIME_M_U Minutes units. Represented in BCD digit (0-9).	0x0
15	-	- Reserved	0x0
14:12	R/W	RTC_TIME_S_T Seconds tens. Represented in BCD digit (0-9).	0x0
11:8	R/W	RTC_TIME_S_U Seconds units. Represented in BCD digit (0-9).	0x0
7:4	R/W	RTC_TIME_H_T Hundredths of a second tens. Represented in BCD digit (0-9).	0x0
3:0	R/W	RTC_TIME_H_U Hundredths of a second units. Represented in BCD digit (0-9).	0x0

Table 298: RTC_CALENDAR_REG (0x5000410C)

Bit	Mode	Symbol/Description	Reset
31	R/W	RTC_CAL_CH The value in this register has altered since last read. Read and clear	0x0
30	-	- Reserved	0x0
29:28	R/W	RTC_CAL_C_T Century tens. Represented in BCD digit (1-2).	0x2
27:24	R/W	RTC_CAL_C_U Century units. Represented in BCD digit (0-9).	0x0
23:20	R/W	RTC_CAL_Y_T Year tens. Represented in BCD digit (0-9).	0x0
19:16	R/W	RTC_CAL_Y_U Year units. Represented in BCD digit (0-9).	0x0
15:14	-	- Reserved	0x0
13:12	R/W	RTC_CAL_D_T Date tens. Represented in BCD digit (0-3).	0x0
11:8	R/W	RTC_CAL_D_U Date units. Represented in BCD digit (0-9).	0x1

Bit	Mode	Symbol/Description	Reset
7	R/W	RTC_CAL_M_T Month tens. Represented in BCD digit (0-1).	0x0
6:3	R/W	RTC_CAL_M_U Month units. Represented in BCD digit (0-9).	0x1
2:0	R/W	RTC_DAY Day of the week (arbitrary) units. Represented in BCD digit (0-7).	0x7

Table 299: RTC_TIME_ALARM_REG (0x50004110)

Bit	Mode	Symbol/Description	Reset
31	-	- Reserved	0x0
30	R/W	RTC_TIME_PM In 12-hour clock mode, indicates PM when set.	0x0
29:28	R/W	RTC_TIME_HR_T Hours tens. Represented in BCD digit (0-2).	0x0
27:24	R/W	RTC_TIME_HR_U Hours units. Represented in BCD digit (0-9).	0x0
23	-	- Reserved	0x0
22:20	R/W	RTC_TIME_M_T Minutes tens. Represented in BCD digit (0-5).	0x0
19:16	R/W	RTC_TIME_M_U Minutes units. Represented in BCD digit (0-9).	0x0
15	-	- Reserved	0x0
14:12	R/W	RTC_TIME_S_T Seconds tens. Represented in BCD digit (0-9).	0x0
11:8	R/W	RTC_TIME_S_U Seconds units. Represented in BCD digit (0-9).	0x0
7:4	R/W	RTC_TIME_H_T Hundredths of a second tens. Represented in BCD digit (0-9).	0x0
3:0	R/W	RTC_TIME_H_U Hundredths of a second units. Represented in BCD digit (0-9).	0x0

Table 300: RTC_CALENDAR_ALARM_REG (0x50004114)

Bit	Mode	Symbol/Description	Reset
31:14	R/W	- Reserved	0x0
13:12	R/W	RTC_CAL_D_T Date tens. Represented in BCD digit (0-3).	0x0

Bit	Mode	Symbol/Description	Reset
11:8	R/W	RTC_CAL_D_U Date units. Represented in BCD digit (0-9).	0x0
7	R/W	RTC_CAL_M_T Month tens. Represented in BCD digit (0-1).	0x0
6:3	R/W	RTC_CAL_M_U Month units. Represented in BCD digit (0-9).	0x0
2:0	-	- Reserved	0x0

Table 301: RTC_ALARM_ENABLE_REG (0x50004118)

Bit	Mode	Symbol/Description	Reset
5	R/W	RTC_ALARM_MNTH_EN Alarm on month enable. Enable to trigger alarm when data specified in Calendar Alarm Register (M_T and M_U) has been reached.	0x0
4	R/W	RTC_ALARM_DATE_EN Alarm on date enable. Enable to trigger alarm when data specified in Calendar Alarm Register (D_T and D_U) has been reached.	0x0
3	R/W	RTC_ALARM_HOUR_EN Alarm on hour enable. Enable to trigger alarm when data specified in Time Alarm Register (PM, HR_T and HR_U) has been reached.	0x0
2	R/W	RTC_ALARM_MIN_EN Alarm on minute enable. Enable to trigger alarm when data specified in Time Alarm Register (M_T and M_U) has been reached.	0x0
1	R/W	RTC_ALARM_SEC_EN Alarm on second enable. Enable to trigger alarm when data specified in Time Alarm Register (S_T and S_U) has been reached.	0x0
0	R/W	RTC_ALARM_HOS_EN Alarm on hundredths of a second enable. Enable to trigger alarm when data specified in Time Alarm Register (H_T and H_U) has been reached.	0x0

Table 302: RTC_EVENT_FLAGS_REG (0x5000411C)

Bit	Mode	Symbol/Description	Reset
6	R	RTC_EVENT_ALRM Alarm event flag. Indicate that alarm event occurred since the last reset.	0x0
5	R	RTC_EVENT_MNTH Month rolls over event flag. Indicate that month rolls over event occurred since the last reset.	0x0
4	R	RTC_EVENT_DATE Date rolls over event flag. Indicate that date rolls over event occurred since the last reset.	0x0
3	R	RTC_EVENT_HOUR Hour rolls over event flag. Indicate that hour rolls over event occurred since the last reset.	0x0

Bit	Mode	Symbol/Description	Reset
2	R	RTC_EVENT_MIN Minute rolls over event flag. Indicate that minute rolls over event occurred since the last reset.	0x0
1	R	RTC_EVENT_SEC Second rolls over event flag. Indicate that second rolls over event occurred since the last reset.	0x0
0	R	RTC_EVENT_HOS Hundredths of a second event flag. Indicate that hundredths of a second rolls over event occurred since the last reset.	0x0

Table 303: RTC_INTERRUPT_ENABLE_REG (0x50004120)

Bit	Mode	Symbol/Description	Reset
6	W	RTC_ALRM_INT_EN Interrupt on alarm enable. Enable to issue the interrupt when alarm event occurred.	0x0
5	W	RTC_MNTH_INT_EN Interrupt on month enable. Enable to issue the interrupt when month event occurred.	0x0
4	W	RTC_DATE_INT_EN Interrupt on date enable. Enable to issue the interrupt when date event occurred.	0x0
3	W	RTC_HOUR_INT_EN Interrupt on hour enable. Enable to issue the interrupt when hour event occurred.	0x0
2	W	RTC_MIN_INT_EN Interrupt on minute enable. Enable to issue the interrupt when minute event occurred.	0x0
1	W	RTC_SEC_INT_EN Interrupt on second enable. Enable to issue the interrupt when second event occurred.	0x0
0	W	RTC_HOS_INT_EN Interrupt on hundredths of a second enable. Enable to issue the interrupt when hundredths of a second event occurred.	0x0

Table 304: RTC_INTERRUPT_DISABLE_REG (0x50004124)

Bit	Mode	Symbol/Description	Reset
6	W	RTC_ALRM_INT_DIS Interrupt on alarm disable. Disable to issue the interrupt when alarm event occurred.	0x0
5	W	RTC_MNTH_INT_DIS Interrupt on month disable. Disable to issue the interrupt when month event occurred.	0x0
4	W	RTC_DATE_INT_DIS Interrupt on date disable. Disable to issue the interrupt when date event occurred.	0x0
3	W	RTC_HOUR_INT_DIS IInterrupt on hour disable. Disable to issue the interrupt when hour event occurred.	0x0

Bit	Mode	Symbol/Description	Reset
2	W	RTC_MIN_INT_DIS Interrupt on minute disable. Disable to issue the interrupt when minute event occurred.	0x0
1	W	RTC_SEC_INT_DIS Interrupt on second disable. Disable to issue the interrupt when second event occurred.	0x0
0	W	RTC_HOS_INT_DIS Interrupt on hundredths of a second disable. Disable to issue the interrupt when hundredths of a second event occurred.	0x0

Table 305: RTC_INTERRUPT_MASK_REG (0x50004128)

Bit	Mode	Symbol/Description	Reset
6	R	RTC_ALRM_INT_MSK Mask alarm interrupt. It can be cleared (set) by setting corresponding bit (ALRM) in Interrupt Enable Register (Interrupt Disable Register).	0x1
5	R	RTC_MNTH_INT_MSK Mask month interrupt. It can be cleared (set) by setting corresponding bit (MNTH) in Interrupt Enable Register (Interrupt Disable Register).	0x1
4	R	RTC_DATE_INT_MSK Mask date interrupt. It can be cleared (set) by setting corresponding bit (DATE) in Interrupt Enable Register (Interrupt Disable Register).	0x1
3	R	RTC_HOUR_INT_MSK Mask hour interrupt. It can be cleared (set) by setting corresponding bit (HOUR) in Interrupt Enable Register (Interrupt Disable Register).	0x1
2	R	RTC_MIN_INT_MSK Mask minute interrupt. It can be cleared (set) by setting corresponding bit (MIN) in Interrupt Enable Register (Interrupt Disable Register).	0x1
1	R	RTC_SEC_INT_MSK Mask second interrupt. It can be cleared (set) by setting corresponding bit (SEC) in Interrupt Enable Register (Interrupt Disable Register).	0x1
0	R	RTC_HOS_INT_MSK Mask hundredths of a second interrupt. It can be cleared (set) by setting corresponding bit (HOS) in Interrupt Enable Register (Interrupt Disable Register).	0x1

Table 306: RTC_STATUS_REG (0x5000412C)

Bit	Mode	Symbol/Description	Reset
3	R	RTC_VALID_CAL_ALM Valid Calendar Alarm. If cleared, then indicates that invalid entry occurred when writing to Calendar Alarm Register.	0x1
2	R	RTC_VALID_TIME_ALM Valid Time Alarm. If cleared, then indicates that invalid entry occurred when writing to Time Alarm Register.	0x1
1	R	RTC_VALID_CAL Valid Calendar. If cleared, then indicates that invalid entry occurred when writing to Calendar Register.	0x1

Bit	Mode	Symbol/Description	Reset
0	R	RTC_VALID_TIME Valid Time. If cleared, then indicates that invalid entry occurred when writing to Time Register.	0x1

Table 307: RTC_KEEP_RTC_REG (0x50004130)

Bit	Mode	Symbol/Description	Reset
0	R/W	RTC_KEEP Keep RTC. When high, the time and calendar registers and any other registers which directly affect or are affected by the time and calendar registers are NOT reset when software reset is applied. When low, the software reset will reset every register except the keep RTC and control registers.	0x1

Table 308: Register map CRG_TIM

Address	Register	Description
0x5000424c	CLK_RTCDIV_REG	Divisor for RTC 100 Hz clock

Table 309: CLK_RTCDIV_REG (0x5000424C)

Bit	Mode	Symbol/Description	Reset
21	R/W	RTC_RESET_REQ Reset request for the RTC module	0x0
20	R/W	RTC_DIV_ENABLE Enable for the 100 Hz generation for the RTC block	0x0
19	R/W	RTC_DIV_DENOM Selects the denominator for the fractional division: 0b0: 1000 0b1: 1024	0x0
18:10	R/W	RTC_DIV_INT Integer divisor part for RTC 100 Hz generation	0x147
9:0	R/W	RTC_DIV_FRAC Fractional divisor part for RTC 100 Hz generation. If RTC_DIV_DENOM = 1, <RTC_DIV_FRAC> out of 1024 cycles will divide by <RTC_DIV_INT+1>, the rest is <RTC_DIV_INT> If RTC_DIV_DENOM = 0, <RTC_DIV_FRAC> out of 1000 cycles will divide by <RTC_DIV_INT+1>, the rest is <RTC_DIV_INT>	0x2A8

31.15 SPI Interface Registers

Table 310: Register map SPI

Address	Register	Description
0x50001200	SPI_CTRL_REG	SPI control register
0x50001204	SPI_CONFIG_REG	SPI control register
0x50001208	SPI_CLOCK_REG	SPI clock register
0x5000120c	SPI_FIFO_CONFIG_REG	SPI FIFO configuration register
0x50001210	SPI_IRQ_MASK_REG	SPI interrupt mask register
0x50001214	SPI_STATUS_REG	SPI status register
0x50001218	SPI_FIFO_STATUS_REG	SPI RX/TX fifo status register
0x5000121c	SPI_FIFO_READ_REG	SPI RX FIFO read register
0x50001220	SPI_FIFO_WRITE_REG	SPI TX FIFO write register
0x50001224	SPI_CS_CONFIG_REG	SPI CS configuration register
0x50001228	SPI_FIFO_HIGH_REG	Spi TX/RX High 16-bit word
0x5000122c	SPI_TXBUFFER_FORCE_L_REG	SPI TX buffer force low value
0x50001230	SPI_TXBUFFER_FORCE_H_REG	SPI TX buffer force high value
0x50001234	SPI_PAUSE_CTRL_REG	SPI pause control register

Table 311: SPI_CTRL_REG (0x50001200)

Bit	Mode	Symbol/Description	Reset
7	R/W	SPI_SWAP_BYTES 0 = Normal operation 1 = LSB and MSB are swaped in the APB interface In case of 8-bit SPI interface, DMA/SPI can be configured in 16-bit mode to off load the bus. Enabling SPI_SWAP_BYTES bytes will read/write correctly.	0x0
6	R/W	SPI_CAPTURE_AT_NEXT_EDGE 0 = SPI captures data at correct clock edge 1 = SPI captures data at the next clock edge (only for Master mode and high clock)	0x0
5	R/W	SPI_FIFO_RESET 0 = FIFO normal operation 1 = FIFO in reset state	0x0
4	R/W	SPI_DMA_RX_EN Applicable only when SPI_RX_EN = 1 0 = No DMA request for RX 1 = DMA request when SPI_STATUS_RX_FULL = 1	0x0
3	R/W	SPI_DMA_TX_EN Applicable only when SPI_TX_EN = 1 0 = No DMA request for TX 1 = DMA request when SPI_STATUS_TX_EMPTY = 1	0x0
2	R/W	SPI_RX_EN 0 = RX path is disabled	0x0

Bit	Mode	Symbol/Description	Reset
		1 = RX path is enabled Note: If master clk async or SPI mode = 1 or SPI mode = 3, readonly is not supported.	
1	R/W	SPI_TX_EN 0 = TX path is disabled 1 = TX path is enabled	0x0
0	R/W	SPI_EN 0 = SPI module is disable 1 = SPI module is enable	0x0

Table 312: SPI_CONFIG_REG (0x50001204)

Bit	Mode	Symbol/Description	Reset
7	R/W	SPI_SLAVE_EN 0 = SPI module master mode 1 = SPI module slave mode	0x0
6:2	R/W	SPI_WORD_LENGTH Define the SPI word length = 1+ SPI_WORD_LENGTH (range 4 to 32)	0x0
1:0	R/W	SPI_MODE Define the SPI mode (CPOL, CPHA) 0 = New data on falling, capture on rising, clk low in idle state 1 = New data on rising, capture on falling, Clk low in idle state 2 = New data on rising, capture on falling, Clk high in idle state 3 = New data on falling, capture on rising Clk high in idle state	0x0

Table 313: SPI_CLOCK_REG (0x50001208)

Bit	Mode	Symbol/Description	Reset
7	R/W	SPI_MASTER_CLK_MODE Should always be 1	0x0
6:0	R/W	SPI_CLK_DIV Applicable only in Master mode Defines the SPI clock frequency in master only mode SPI_CLK = module_clk/2*(SPI_CLK_DIV+1) when SPI_CLK_DIV not 0x7F if SPI_CLK_DIV = 0x7F, then SPI_CLK = module_clk	0x0

Table 314: SPI_FIFO_CONFIG_REG (0x5000120C)

Bit	Mode	Symbol/Description	Reset
7:4	R/W	SPI_RX_TL Receive FIFO threshold level in bytes. Control the level of bytes in FIFO that triggers the RX_FULL interrupt. IRQ occurs when FIFO level is more or equal to SPI_RX_TL+1. FIFO level is from 0 to 4.	0x0
3:0	R/W	SPI_TX_TL Transmit FIFO threshold level in bytes. Control the level of bytes in FIFO that triggers the TX_EMPTY interrupt. IRQ occurs when FIFO level is less or equal to SPI_TX_TL. FIFO level is from 0 to 4.	0x0

Table 315: SPI_IRQ_MASK_REG (0x50001210)

Bit	Mode	Symbol/Description	Reset
1	R/W	SPI_IRQ_MASK_RX_FULL 0 = FIFO RX full IRQ is masked 1 = FIFO RX full IRQ is enabled	0x0
0	R/W	SPI_IRQ_MASK_TX_EMPTY 0 = FIFO TX empty IRG is masked 1 = FIFO TX empty IRG is enabled	0x0

Table 316: SPI_STATUS_REG (0x50001214)

Bit	Mode	Symbol/Description	Reset
1	R	SPI_STATUS_RX_FULL Auto clear 0 = RX FIFO level is less than SPI_RX_TL+1 1 = RX FIFO level is more or equal to SPI_RX_TL+1	0x0
0	R	SPI_STATUS_TX_EMPTY Auto clear 0 = TX FIFO level is larger than SPI_TX_TL 1 = TX FIFO level is less or equal to SPI_TX_TL	0x1

Table 317: SPI_FIFO_STATUS_REG (0x50001218)

Bit	Mode	Symbol/Description	Reset
15	R	SPI_TRANSACTION_ACTIVE In Master mode 0 = SPI transaction is inactive 1 = SPI transaction is active	0x0
14	R	SPI_RX_FIFO_OVFL When 1, receive data is not written to FIFO because FIFO was full and interrupt is generated. It clears with SPI_CTRL_REG.SPI_FIFO_RESET	0x0
13	R	SPI_STATUS_TX_FULL 0 = TX FIFO is not full 1 = TX FIFO is full	0x0
12	R	SPI_STATUS_RX_EMPTY 0 = RX FIFO is not empty 1 = RX FIFO is empty	0x1
11:6	R	SPI_TX_FIFO_LEVEL Gives the number of bytes in TX FIFO	0x0
5:0	R	SPI_RX_FIFO_LEVEL Gives the number of bytes in RX FIFO	0x0

Table 318: SPI_FIFO_READ_REG (0x5000121C)

Bit	Mode	Symbol/Description	Reset
15:0	R	SPI_FIFO_READ Read from RX FIFO Read access is permit only if SPI_STATUS_RX_EMPTY = 0. Returns the 16 LSb.	0x0

Table 319: SPI_FIFO_WRITE_REG (0x50001220)

Bit	Mode	Symbol/Description	Reset
15:0	R0/W	SPI_FIFO_WRITE Write to TX FIFO. Write access is permit only if SPI_STATUS_TX_FULL is 0.	0x0

Table 320: SPI_CS_CONFIG_REG (0x50001224)

Bit	Mode	Symbol/Description	Reset
2:0	R/W	SPI_CS_SELECT Control the cs output in Master mode 0 = None slave device selected 1 = Selected slave device connected to GPIO with FUNC_MODE = SPI_CS0 2 = Selected slave device connected to GPIO with FUNC_MODE = SPI_CS1 4 = Selected slave device connected to GPIO with FUNC_MODE = GPIO	0x0

Table 321: SPI_FIFO_HIGH_REG (0x50001228)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	SPI_FIFO_HIGH RX/TX FIFO data. 16 MSb when SPI word is larger than 16 bits. This register has to be written before SPI_FIFO_WRITE_REG. This register has to be read after SPI_FIFO_READ_REG.	0x0

Table 322: SPI_TXBUFFER_FORCE_L_REG (0x5000122C)

Bit	Mode	Symbol/Description	Reset
15:0	W	SPI_TXBUFFER_FORCE_L Write directly the TX buffer (2 LSB). It must be used only in Slave mode.	0x0

Table 323: SPI_TXBUFFER_FORCE_H_REG (0x50001230)

Bit	Mode	Symbol/Description	Reset
15:0	W	SPI_TXBUFFER_FORCE_H Write directly the TX buffer (2 MSB). It must be used only in Slave mode. This register has to be written before SPI_FIFO_WRITE_REG.	0x0

Table 324: SPI_PAUSE_CTRL_REG (0x50001234)

Bit	Mode	Symbol/Description	Reset
12	R/W	SPI_PAUSE_DATA_RDY_IS_MISO 0 = Data ready is the signal from the PPA	0x0

Bit	Mode	Symbol/Description	Reset
		1 = DAta ready is the MISO signal	
11	R/W	SPI_PAUSE_READY_ACTIVE_HIGH 0 = Ready input is active high (no gap is signal is low) 1 = Ready input is active low (no gap is signal is high)	0x0
10	R/W	SPI_PAUSE_CLOCK_EN In case SPI_PAUSE_EN = 1 Enable pause transcation based on programable delay (SPI_PAUSE_CLOCKS) Note: Can be combined with SPI_PAUSE_DATA_READY_EN	0x0
9	R/W	SPI_PAUSE_DATA_READY_EN In case SPI_PAUSE_EN = 1 Enable pause transcation based on external signal. Its polarity is defined by the SPI_PAUSE_READY_ACTIVE_HIGH Note: Can be combined with SPI_PAUSE_CLOCK_EN	0x0
8	R/W	SPI_PAUSE_EN Pause transaction inserts gap between SPI words (only in Master mode and SPI_TX_EN=1). 0 = SPI pause transaction is disabled 1 = SPI pause transaction is enabled	0x0
7:0	R/W	SPI_PAUSE_CLOCKS Defines the gap period in number of SPI clock cycles minus one	0x0

31.16 Timer and Triple PWM Registers

Table 325: Register map Timer+3PWM

Address	Register	Description
0x50003400	TIMER0_CTRL_REG	Timer0 control register
0x50003402	TIMER0_ON_REG	Timer0 on control register
0x50003404	TIMER0_RELOAD_M_REG	16 bits reload value for Timer0
0x50003406	TIMER0_RELOAD_N_REG	16 bits reload value for Timer0
0x50003408	TRIPLE_PWM_FREQUENCY	Frequency for PWM 2, 3, 4, 5, 6 and 7
0x5000340a	PWM2_START_CYCLE	Defines start Cycle for PWM2
0x5000340c	PWM3_START_CYCLE	Defines start Cycle for PWM3
0x5000340e	PWM4_START_CYCLE	Defines start Cycle for PWM4
0x50003410	PWM5_START_CYCLE	Defines start Cycle for PWM5
0x50003412	PWM6_START_CYCLE	Defines start Cycle for PWM6
0x50003414	PWM7_START_CYCLE	Defines start Cycle for PWM7
0x50003416	PWM2_END_CYCLE	Defines end Cycle for PWM2
0x50003418	PWM3_END_CYCLE	Defines end Cycle for PWM3
0x5000341a	PWM4_END_CYCLE	Defines end Cycle for PWM4
0x5000341c	PWM5_END_CYCLE	Defines end Cycle for PWM5
0x5000341e	PWM6_END_CYCLE	Defines end Cycle for PWM6
0x50003420	PWM7_END_CYCLE	Defines end Cycle for PWM7
0x50003422	TRIPLE_PWM_CTRL_REG	PWM 2, 3, 4, 5, 6, 7 Control

Table 326: TIMER0_CTRL_REG (0x50003400)

Bit	Mode	Symbol/Description	Reset
15:4	-	- Reserved	0x0
3	R/W	PWM_MODE 0 = PWM signals are 1 during high time. 1 = PWM signals send out the (fast) clock divided by 2 during high time. So it will be in the range of 1 to 8 MHz.	0x0
2	R/W	TIM0_CLK_DIV 1 = Timer0 uses selected clock frequency as is. 0 = Timer0 uses selected clock frequency divided by 10. Note that this applies only to the ON-counter.	0x0
1	R/W	TIM0_CLK_SEL 1 = Timer0 uses 16, 8, 4, or 2 MHz (fast) clock frequency 0 = Timer0 uses LP clock	0x0
0	R/W	TIM0_CTRL 0 = Timer0 is off and in reset state 1 = Timer0 is running	0x0

Table 327: TIMER0_ON_REG (0x50003402)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	TIM0_ON Timer0 On reload value: If read the actual ON-counter value is returned	0x0

Table 328: TIMER0_RELOAD_M_REG (0x50003404)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	TIM0_M Timer0 "high" reload value If read, the actual T0-counter value is returned	0x0

Table 329: TIMER0_RELOAD_N_REG (0x50003406)

Bit	Mode	Symbol/Description	Reset
15:0	R/W	TIM0_N Timer0 "low" reload value: If read, the actual T0-counter value is returned	0x0

Table 330: TRIPLE_PWM_FREQUENCY (0x50003408)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	PWM_FREQ Defines the freqeuancy of PWM 2, 3, 4, 5, 6, and 7. PWM freq = module Frequency/(value+1) Module frequency is LP_CLK when TRIPLE_PWM_CLK_SEL = 0, else is the sys_clk divided by TMR_DI.	0x0

Table 331: PWM2_START_CYCLE (0x5000340A)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	START_CYCLE Defines the cycle in which the PWM becomes high. If start_cycle is larger than freq or end_cycle is equal to start_cycle, pwm out is always 0	0x0

Table 332: PWM3_START_CYCLE (0x5000340C)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	START_CYCLE Defines the cycle in which the PWM becomes high. If start_cycle is larger than freq or end_cycle is equal to start_cycle, pwm out is always 0	0x0

Table 333: PWM4_START_CYCLE (0x5000340E)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	START_CYCLE	0x0

Bit	Mode	Symbol/Description	Reset
		Defines the cycle in which the PWM becomes high. If start_cycle is larger than freq or end_cycle is equal to start_cycle, pwm out is always 0	

Table 334: PWM5_START_CYCLE (0x50003410)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	START_CYCLE Defines the cycle in which the PWM becomes high. If start_cycle is larger than freq or end_cycle is equal to start_cycle, pwm out is always 0	0x0

Table 335: PWM6_START_CYCLE (0x50003412)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	START_CYCLE Defines the cycle in which the PWM becomes high. If start_cycle is larger than freq or end_cycle is equal to start_cycle, pwm out is always 0	0x0

Table 336: PWM7_START_CYCLE (0x50003414)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	START_CYCLE Defines the cycle in which the PWM becomes high. if start_cycle is larger than freq or end_cycle is equal to start_cycle, pwm out is always 0	0x0

Table 337: PWM2_END_CYCLE (0x50003416)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	END_CYCLE Defines the cycle in which the PWM becomes low. If end_cycle is larger then freq and start_cycle is not larger then freq, output is always 1	0x0

Table 338: PWM3_END_CYCLE (0x50003418)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	END_CYCLE Defines the cycle in which the PWM becomes low. If end_cycle is larger then freq and start_cycle is not larger then freq, output is always 1	0x0

Table 339: PWM4_END_CYCLE (0x5000341A)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	END_CYCLE Defines the cycle in which the PWM becomes low. If end_cycle is larger then freq and start_cycle is not larger then freq, output is always 1	0x0

Table 340: PWM5_END_CYCLE (0x5000341C)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	END_CYCLE	0x0

Bit	Mode	Symbol/Description	Reset
		Defines the cycle in which the PWM becomes low. If end_cycle is larger then freq and start_cycle is not larger then freq, output is always 1	

Table 341: PWM6_END_CYCLE (0x5000341E)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	END_CYCLE Defines the cycle in which the PWM becomes low. If end_cycle is larger then freq and start_cycle is not larger then freq, output is always 1	0x0

Table 342: PWM7_END_CYCLE (0x50003420)

Bit	Mode	Symbol/Description	Reset
13:0	R/W	END_CYCLE Defines the cycle in which the PWM becomes low. If end_cycle is larger then freq and start_cycle is not larger then freq, output is always 1	0x0

Table 343: TRIPLE_PWM_CTRL_REG (0x50003422)

Bit	Mode	Symbol/Description	Reset
3	R/W	TRIPLE_PWM_CLK_SEL 1 = Timer2 uses 16, 8, 4, or 2 MHz (fast) clock frequency 0 = Timer2 uses LP clock	0x0
2	R/W	HW_PAUSE_EN 1 = Hardware can pause PWM 2, 3, 4, 5, 6, 7	0x1
1	R/W	SW_PAUSE_EN 1 = PWM 2 3 4 5 6 7 are paused	0x0
0	R/W	TRIPLE_PWM_ENABLE 1 = Enable PWM 2 3 4 5 6 7	0x0

31.17 Timer1 Registers

Table 344: Register map Timer1

Address	Register	Description
0x50004000	TIMER1_CTRL_REG	Timer1 control register
0x50004004	TIMER1_CAPTURE_REG	Timer1 Capture control register
0x50004008	TIMER1_STATUS_REG	Timer1 counter value
0x5000400c	TIMER1_CAPCNT1_VALUE_REG	Timer1 value for event on GPIO1
0x50004010	TIMER1_CAPCNT2_VALUE_REG	Timer1 value for event on GPIO2
0x50004014	TIMER1_CLR_EVENT_REG	Clear event register

Table 345: TIMER1_CTRL_REG (0x50004000)

Bit	Mode	Symbol/Description	Reset
16	R/W	TIMER1_CLK_EN 0 = Timer1 clock is disabled 1 = Timer1 clock is enabled	0x0
15	R/W	TIMER1_USE_SYS_CLK 0 = Timer1 use the clock LP clock 1 = Timer1 use the system clock	0x0
14	R/W	TIMER1_FREE_RUN_MODE_EN Applicable when timer counts up 1 = Timer1 goes to zero when it reaches the max value 0 = Timer1 goes to zero when it reaches the reload value	0x0
13	R/W	TIMER1_IRQ_EN 0 = Timer1 IRQ masked 1 = Timer1 IRQ unmasked	0x0
12	R/W	TIMER1_COUNT_DOWN_EN 0 = Timer1 counts up 1 = Timer1 counts down	0x0
11	R/W	TIMER1_ENABLE 0 = Timer1 disabled 1 = Timer1 enabled	0x0
10:0	R/W	TIMER1_RELOAD Reload or max value in timer mode. Actual delay is the register value plus synchronization time (3 clock cycles)	0x0

Table 346: TIMER1_CAPTURE_REG (0x50004004)

Bit	Mode	Symbol/Description	Reset
27	R/W	TIMER1_IN2_STAMP_TYPE 0 = On each event store the counter value 1 = On each event store the RTC time stamp	0x0
26:21	R/W	TIMER1_IN2_PERIOD_MAX Gives the number of periods +1 of IN2, in which module counts	0x0
20	R/W	TIMER1_IN2_IRQ_EN 1 = Interrupt is generated when capture is occurred or was counted TIMER1_IN2_PERIOD_MAX 0 = Interrupt is masked	0x0
19	R/W	TIMER1_IN2_COUNT_EN 0 = Capture mode 1 = Count mode	0x0
18	R/W	TIMER1_IN2_EVENT_FALL_EN 0 = Rising edge event 1 = Falling edge event It should be written when TIMER1_GPIO2_CONF = 0 to prevent false events	0x0
17:14	R/W	TIMER1_GPIO2_CONF	0x0

Bit	Mode	Symbol/Description	Reset
		0,13,14,15 = IN2 is not used 1..12 = Defines the P0 pin (0..11) module will use as IN2	
13	R/W	TIMER1_IN1_STAMP_TYPE 0 = On each event store the counter value 1 = On each event store the RTC time stamp	0x0
12:7	R/W	TIMER1_IN1_PERIOD_MAX Gives the number of periods +1 of IN1, in which module counts	0x0
6	R/W	TIMER1_IN1_IRQ_EN 1 = Interrupt is generated when capture is occurred or was counted TIMER1_IN1_PERIOD_MAX 0 = Interrupt is masked	0x0
5	R/W	TIMER1_IN1_COUNT_EN 0 = Capture mode 1 = Count mode	0x0
4	R/W	TIMER1_IN1_EVENT_FALL_EN 0 = Rising edge event 1 = Falling edge event It should be written when TIMER1_GPIO1_CONF = 0 to prevent false events	0x0
3:0	R/W	TIMER1_GPIO1_CONF 0,13,14,15 = IN1 is not used 1..12 = Defines the P0 pin (0..11) module will use as IN1	0x0

Table 347: TIMER1_STATUS_REG (0x50004008)

Bit	Mode	Symbol/Description	Reset
15	R	TIMER1_IN2_OVRFLW 1 = New IN2 event occurred while Interrupt was pending. TIMER1_CAPCNT2_VALUE_REG gives the time stamp of the first event.	0x0
14	R	TIMER1_IN1_OVRFLW 1 = New IN1 event occurred while Interrupt was pending. TIMER1_CAPCNT1_VALUE_REG gives the time stamp of the first event.	0x0
13	R	TIMER1_IN2_EVENT 1 = Pending Capture 2 interrupt. It has to be clear writing 1 to TIMER1_CLR_IN2_EVENT	0x0
12	R	TIMER1_IN1_EVENT 1 = Pending Capture 1 interrupt. It has to be clear writing 1 to TIMER1_CLR_IN1_EVENT	0x0
11	R	TIMER1_TIMER_EVENT 1 = Pending Timer interrupt. it has to be clear writing 1 to TIMER1_CLR_TIMER_EVENT	0x0
10:0	R	TIMER1_TIMER_VALUE Gives the current timer value	0x0

Table 348: TIMER1_CAPCNT1_VALUE_REG (0x5000400C)

Bit	Mode	Symbol/Description	Reset
21:11	R	TIMER1_CAPCNT1_RTC_HIGH In Counter mode: Not used In Capture mode Gives the RTC time stamp (high part) when an IN1 event was occurred	0x0
10:0	R	TIMER1_CAPCNT1_VALUE In Counter mode: Gives the number of timer clock cycles minus 1 which was measured during TIMER1_IN1_PERIOD_MAX periods of IN1 In Capture mode (TIMER1_IN1_STAMP_TYPE = 0): Gives the Counter value when an IN1 event was occurred In Capture mode (TIMER1_IN1_STAMP_TYPE = 1): Gives the RTC time stamp (low part) when an IN1 event was occurred	0x0

Table 349: TIMER1_CAPCNT2_VALUE_REG (0x50004010)

Bit	Mode	Symbol/Description	Reset
21:11	R	TIMER1_CAPCNT2_RTC_HIGH In Counter mode: Not used In Capture mode Gives the RTC time stamp (high part) when an IN2 event was occurred	0x0
10:0	R	TIMER1_CAPCNT2_VALUE In Counter mode: Gives the number of timer clock cycles minus 1 which was measured during TIMER1_IN2_PERIOD_MAX periods of IN2 In Capture mode (TIMER1_IN2_STAMP_TYPE = 0): Gives the Counter value when an IN2 event was occurred In Capture mode (TIMER1_IN2_STAMP_TYPE = 1): Gives the RTC time stamp (low part) when an IN2 event was occurred	0x0

Table 350: TIMER1_CLR_EVENT_REG (0x50004014)

Bit	Mode	Symbol/Description	Reset
2	R0/WC	TIMER1_CLR_IN2_EVENT Write 1 to clear the TIMER1_IN2_EVENT and TIMER1_IN2_OVRFLW	0x0
1	R0/WC	TIMER1_CLR_IN1_EVENT Write 1 to clear the TIMER1_IN1_EVENT and TIMER1_IN1_OVRFLW	0x0
0	R0/WC	TIMER1_CLR_TIMER_EVENT Write 1 to clear the TIMER1_TIMER_EVENT	0x0

31.18 UART Interface Registers

Table 351: Register map UART

Address	Register	Description
0x50001000	UART_RBR_THR_DLL_REG	Receive Buffer Register/Transmit Holding Register/Divisor Latch Low
0x50001004	UART_IER_DLH_REG	Interrupt Enable Register/Divisor Latch High
0x50001008	UART_IIR_FCR_REG	Interrupt Identification Register/FIFO Control Register
0x5000100c	UART_LCR_REG	Line Control Register
0x50001010	UART_MCR_REG	Modem Control Register
0x50001014	UART_LSR_REG	Line Status Register
0x50001018	UART_MSR_REG	Modem Status Register
0x5000101c	UART_SCR_REG	Scratchpad Register
0x50001030	UART_SRBR_STHR0_REG	Shadow Receive/Transmit Buffer Register
0x50001034	UART_SRBR_STHR1_REG	Shadow Receive/Transmit Buffer Register
0x50001038	UART_SRBR_STHR2_REG	Shadow Receive/Transmit Buffer Register
0x5000103c	UART_SRBR_STHR3_REG	Shadow Receive/Transmit Buffer Register
0x50001040	UART_SRBR_STHR4_REG	Shadow Receive/Transmit Buffer Register
0x50001044	UART_SRBR_STHR5_REG	Shadow Receive/Transmit Buffer Register
0x50001048	UART_SRBR_STHR6_REG	Shadow Receive/Transmit Buffer Register
0x5000104c	UART_SRBR_STHR7_REG	Shadow Receive/Transmit Buffer Register
0x50001050	UART_SRBR_STHR8_REG	Shadow Receive/Transmit Buffer Register
0x50001054	UART_SRBR_STHR9_REG	Shadow Receive/Transmit Buffer Register
0x50001058	UART_SRBR_STHR10_REG	Shadow Receive/Transmit Buffer Register
0x5000105c	UART_SRBR_STHR11_REG	Shadow Receive/Transmit Buffer Register
0x50001060	UART_SRBR_STHR12_REG	Shadow Receive/Transmit Buffer Register
0x50001064	UART_SRBR_STHR13_REG	Shadow Receive/Transmit Buffer Register
0x50001068	UART_SRBR_STHR14_REG	Shadow Receive/Transmit Buffer Register
0x5000106c	UART_SRBR_STHR15_REG	Shadow Receive/Transmit Buffer Register
0x50001070	UART_FAR_REG	FIFO Access Register
0x5000107c	UART_USR_REG	UART Status Register
0x50001080	UART_TFL_REG	Transmit FIFO Level

Address	Register	Description
0x50001084	UART_RFL_REG	Receive FIFO Level
0x50001088	UART_SRR_REG	Software Reset Register.
0x5000108c	UART_SRTS_REG	Shadow Request to Send
0x50001090	UART_SBCR_REG	Shadow Break Control Register
0x50001094	UART_SDMAM_REG	Shadow DMA Mode
0x50001098	UART_SFE_REG	Shadow FIFO Enable
0x5000109c	UART_SRT_REG	Shadow RCVR Trigger
0x500010a0	UART_STET_REG	Shadow TX Empty Trigger
0x500010a4	UART_HTX_REG	Halt TX
0x500010a8	UART_DMASA_REG	DMA Software Acknowledge
0x500010c0	UART_DLF_REG	Divisor Latch Fraction Register
0x500010f8	UART_UCV_REG	Component Version
0x500010fa	UART_UCV_HIGH_REG	Component Version
0x500010fc	UART_CTR_REG	Component Type Register
0x500010fe	UART_CTR_HIGH_REG	Component Type Register
0x50001100	UART2_RBR_THR_DLL_REG	Receive Buffer Register/Transmit Holding Register/Divisor Latch Low
0x50001104	UART2_IER_DLH_REG	Interrupt Enable Register/Divisor Latch High
0x50001108	UART2_IIR_FCR_REG	Interrupt Identification Register/FIFO Control Register
0x5000110c	UART2_LCR_REG	Line Control Register
0x50001110	UART2_MCR_REG	Modem Control Register
0x50001114	UART2_LSR_REG	Line Status Register
0x5000111c	UART2_SCR_REG	Scratchpad Register
0x50001130	UART2_SRBR_STHR0_REG	Shadow Receive/Transmit Buffer Register
0x50001134	UART2_SRBR_STHR1_REG	Shadow Receive/Transmit Buffer Register
0x50001138	UART2_SRBR_STHR2_REG	Shadow Receive/Transmit Buffer Register
0x5000113c	UART2_SRBR_STHR3_REG	Shadow Receive/Transmit Buffer Register
0x50001140	UART2_SRBR_STHR4_REG	Shadow Receive/Transmit Buffer Register
0x50001144	UART2_SRBR_STHR5_REG	Shadow Receive/Transmit Buffer Register
0x50001148	UART2_SRBR_STHR6_REG	Shadow Receive/Transmit Buffer Register
0x5000114c	UART2_SRBR_STHR7_REG	Shadow Receive/Transmit Buffer Register
0x50001150	UART2_SRBR_STHR8_REG	Shadow Receive/Transmit Buffer Register
0x50001154	UART2_SRBR_STHR9_REG	Shadow Receive/Transmit Buffer Register
0x50001158	UART2_SRBR_STHR10_REG	Shadow Receive/Transmit Buffer Register

Address	Register	Description
0x5000115c	UART2_SRBR_STHR11_REG	Shadow Receive/Transmit Buffer Register
0x50001160	UART2_SRBR_STHR12_REG	Shadow Receive/Transmit Buffer Register
0x50001164	UART2_SRBR_STHR13_REG	Shadow Receive/Transmit Buffer Register
0x50001168	UART2_SRBR_STHR14_REG	Shadow Receive/Transmit Buffer Register
0x5000116c	UART2_SRBR_STHR15_REG	Shadow Receive/Transmit Buffer Register
0x50001170	UART2_FAR_REG	FIFO Access Register
0x5000117c	UART2_USR_REG	UART Status Register
0x50001180	UART2_TFL_REG	Transmit FIFO Level
0x50001184	UART2_RFL_REG	Receive FIFO Level
0x50001188	UART2_SRR_REG	Software Reset Register.
0x50001190	UART2_SBCR_REG	Shadow Break Control Register
0x50001194	UART2_SDMAM_REG	Shadow DMA Mode
0x50001198	UART2_SFE_REG	Shadow FIFO Enable
0x5000119c	UART2_SRT_REG	Shadow RCVR Trigger
0x500011a0	UART2_STET_REG	Shadow TX Empty Trigger
0x500011a4	UART2_HTX_REG	Halt TX
0x500011a8	UART2_DMASA_REG	DMA Software Acknowledge
0x500011c0	UART2_DLF_REG	Divisor Latch Fraction Register
0x500011f8	UART2_UCV_REG	Component Version
0x500011fa	UART2_UCV_HIGH_REG	Component Version
0x500011fc	UART2_CTR_REG	Component Type Register
0x500011fe	UART2_CTR_HIGH_REG	Component Type Register

Table 352: UART_RBR_THR_DLL_REG (0x50001000)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	RBR_THR_DLL **Receive Buffer Register: (RBR).** This register contains the data byte received on the serial input port (sin) in UART mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise, it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. **Transmit Holding Register: (THR)**	0x0

Bit	Mode	Symbol/Description	Reset
		This register contains data to be transmitted on the serial output port (sout) in UART mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, 16 number of characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost. **Divisor Latch (Low): (DLL)** This register makes up the lower 8-bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may only be accessed when the DLAB bit (LCR[7]) is set. The output baud rate is equal to the serial clock (sclk) frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq)/(16 * divisor) Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications will occur. Also, when the Divisor Latch is set, at least eight clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data. For the Divisor Latch (High) bits, see the UART_IER_DLH_REG register.	

Table 353: UART_IER_DLH_REG (0x50001004)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7	R/W	PTIME_dlh7 **Interrupt Enable Register: PTIME**, Programmable THRE Interrupt Mode Enable. This is used to enable/disable the generation of THRE Interrupt. 0 = Disabled 1 = Enabled. **Divisor Latch (High): DLH7**, Bit 7 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0
6:4	R/W	dlh6_4 **Divisor Latch (High): DLH6 to DLH4**, Bits 6 to 4 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set, otherwise, this field is reserved. See the UART_RBR_THR_DLL_REG register.	0x0
3	R/W	EDSSI_dlh3 **Interrupt Enable Register: EDSSI**, Enable Modem Status Interrupt. This is used to enable/disable the generation of Modem Status Interrupt. This is the fourth highest priority interrupt. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH3**, Bit 3 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0
2	R/W	ELSI_dhl2	0x0

Bit	Mode	Symbol/Description	Reset
		Interrupt Enable Register: ELSI, Enable Receiver Line Status Interrupt. This is used to enable/disable the generation of Receiver Line Status Interrupt. This is the highest priority interrupt. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH2**, Bit 2 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	
1	R/W	ETBEI_dlh1 **Interrupt Enable Register: ETBEI**, Enable Transmit Holding Register Empty Interrupt. This is used to enable/disable the generation of Transmitter Holding Register Empty Interrupt. This is the third highest priority interrupt. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH1**, Bit 1 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0
0	R/W	ERBFI_dlh0 **Interrupt Enable Register: ERBFI**, Enable Received Data Available Interrupt. This is used to enable/disable the generation of Received Data Available Interrupt and the Character Timeout Interrupt (if in FIFO mode and FIFO's enabled). These are the second highest priority interrupts. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH0**, Bit 0 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0

Table 354: UART_IIR_FCR_REG (0x50001008)

Bit	Mode	Symbol/Description	Reset
7:6	R/W	UART_FIFOSE_RT On read FIFO's Enabled (or FIFOSE): This is used to indicate whether the FIFO's are enabled or disabled. 00 = Disabled. 11 = Enabled. On write RCVR Trigger (or RT): This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt will be generated. In auto flow control mode, it is used to determine when the rts_n signal will be de-asserted. It also determines when the dma_rx_req_n signal will be asserted when in certain modes of operation. The following trigger levels are supported: 00 = 1 character in the FIFO 01 = FIFO 1/4 full 10 = FIFO 1/2 full 11 = FIFO 2 less than full	0x0
5:4	R0/W	UART_TET	0x0

Bit	Mode	Symbol/Description	Reset
		On read Reserved On Write TX Empty Trigger (or TET): This is used to select the empty threshold level at which the THRE Interrupts will be generated when the mode is active. It also determines when the dma_tx_req_n signal will be asserted when in certain modes of operation. The following trigger levels are supported: 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO 1/4 full 11 = FIFO 1/2 full	
3	R/W	UART_IID3_DMAM On Read (Bit3) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect. 1100 = Character timeout. On Write DMA Mode (or DMAM): This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals. 0 = Mode 0 1 = Mode 1	0x0
2	R/W	UART_IID2_XFIFOR On Read (Bit2) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect. 1100 = Character timeout. On Write XMIT FIFO Reset (or XFIFOR): This resets the control portion of the transmit FIFO and treats the FIFO as empty. Note that this bit is "self-clearing" and it is not necessary to clear this bit.	0x0
1	R/W	UART_IID1_RFIFOE On Read (Bit1) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect.	0x0

Bit	Mode	Symbol/Description	Reset
		1100 = Character timeout. On Write RCVR FIFO Reset (or RFIFOR): This resets the control portion of the receive FIFO and treats the FIFO as empty. Note that this bit is "self-clearing" and it is not necessary to clear this bit.	
0	R/W	UART_IID0_FIFOE On Read (Bit0) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect. 1100 = Character timeout. On Write FIFO Enable (or FIFOE): This enables/disables the transmit (XMIT) and receive (RCVR) FIFO's. Whenever the value of this bit is changed both the XMIT and RCVR controller portion of FIFO's will be reset	0x1

Table 355: UART_LCR_REG (0x5000100C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7	R/W	UART_DLAB Divisor Latch Access Bit. Writeable only when UART is not busy (USR[0] is zero). This bit is used to enable reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after the initial baud rate setup to access other registers.	0x0
6	R/W	UART_BC Break Control Bit. This is used to cause a break condition to be transmitted to the receiving device. If set to one, the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. If active (MCR[6] set to one) the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver and the sir_out_n line is forced low.	0x0
5	-	- Reserved	0x0
4	R/W	UART_EPS Even Parity Select. Writeable only when UART is not busy (USR[0] is zero). This is used to select between even and odd parity, when parity is enabled (PEN set to one). If set to one, an even number of logic 1 s is transmitted or checked. If set to zero, an odd number of logic 1 s is transmitted or checked.	0x0
3	R/W	UART_PEN Parity Enable. Writeable only when UART is not busy (USR[0] is zero) This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively.	0x0

Bit	Mode	Symbol/Description	Reset
		0 = Parity disabled 1 = Parity enabled	
2	R/W	UART_STOP Number of stop bits. Writeable only when UART is not busy (USR[0] is zero). This is used to select the number of stop bits per character that the peripheral transmits and receives. If set to zero, one stop bit is transmitted in the serial data. If set to one and the data bits are set to 5 (LCR[1:0] set to zero), one and a half stop bits is transmitted. Otherwise, two stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver checks only the first stop bit. 0 = 1 stop bit 1 = 1.5 stop bits when DLS (LCR[1:0]) is zero, else 2 stop bit	0x0
1:0	R/W	UART_DLS Data Length Select. Writeable only when UART is not busy (USR[0] is zero). This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bits that may be selected areas follows: 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits	0x0

Table 356: UART_MCR_REG (0x50001010)

Bit	Mode	Symbol/Description	Reset
15:7	-	- Reserved	0x0
6	-	- Reserved	0x0
5	R/W	UART_AFCE Auto Flow Control Enable. When FIFOs are enabled and the Auto Flow Control Enable (AFCE) bit is set, Auto Flow Control features are enabled. 0 = Auto Flow Control Mode disabled 1 = Auto Flow Control Mode enabled	0x0
4	R/W	UART_LB LoopBack Bit. This is used to put the UART into a diagnostic mode for test purposes. If operating in UART mode (SIR_MODE not active, MCR[6] set to zero), data on the sout line is held high, while serial data output is looped back to the sin line, internally. In this mode all the interrupts are fully functional. Also, in loopback mode, the modem control inputs (dsr_n, cts_n, ri_n, dcd_n) are disconnected and the modem control outputs (dtr_n, rts_n, out1_n, out2_n) are looped back to the inputs, internally. If operating in infrared mode (SIR_MODE active, MCR[6] set to one), data on the sir_out_n line is held low, while serial data output is inverted and looped back to the sir_in line.	0x0
3	-	- Reserved	0x0

Bit	Mode	Symbol/Description	Reset
2	-	- Reserved	0x0
1	R/W	UART_RTS Request to Send. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data. When Auto RTS Flow Control is not enabled (MCR[5] set to zero), the rts_n signal is set low by programming MCR[1] (RTS) to a high.In Auto Flow Control, AFCE_MODE == Enabled and active (MCR[5] set to one) and FIFOs enable (FCR[0] set to one), the rts_n output is controlled in the same way, but is also gated with the receiver FIFO threshold trigger (rts_n is inactive high when above the threshold). The rts_n signal is de-asserted when MCR[1] is set low. Note that in Loopback mode (MCR[4] set to one), the rts_n output is held inactive high while the value of this location is internally looped back to an input.	0x0
0	-	- Reserved	0x0

Table 357: UART_LSR_REG (0x50001014)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7	R	UART_RFE Receiver FIFO Error bit. This bit is only relevant when FIFOs are enabled (FCR[0] set to one). This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO. 0 = No error in RX FIFO 1 = Error in RX FIFO This bit is cleared when the LSR is read and the character with the error is at the top of the receiver FIFO and there are no subsequent errors in the FIFO.	0x0
6	R	UART_TEMT Transmitter Empty bit. If FIFOs enabled (FCR[0] set to one), this bit is set whenever the Transmitter Shift Register and the FIFO are both empty. If FIFOs are disabled, this bit is set whenever the Transmitter Holding Register(THR) and the Transmitter Shift Register are both empty.	0x1
5	R	UART_THRE Transmit Holding Register Empty bit. If THRE mode is disabled (IER[7] set to zero) and regardless of FIFO's being implemented/enabled or not, this bit indicates that the THR or TX FIFO is empty. This bit is set whenever data is transferred from the THR or TX FIFO to the transmitter shift register and no new data has been written to the THR or TX FIFO. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. If both modes are active (IER[7] set to one and FCR[0] set to one respectively), the functionality is switched to indicate the transmitter FIFO is full, and no longer controls THRE interrupts, which are then controlled by the FCR[5:4] threshold setting.	0x1
4	R	UART_BI	0x0

Bit	Mode	Symbol/Description	Reset
		Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data. If in UART mode (SIR_MODE == Disabled), it is set whenever the serial input, sin, is held in a logic "0" state for longer than the sum of start time + data bits + parity + stop bits. If in infrared mode (SIR_MODE == Enabled), it is set whenever the serial input, sir_in, is continuously pulsed to logic "0" for longer than the sum of start time + data bits + parity + stop bits. A break condition on serial input causes one and only one character, consisting of all zeros, to be received by the UART. In the FIFO mode, the character associated with the break condition is carried through the FIFO and is revealed when the character is at the top of the FIFO. Reading the LSR clears the BI bit. In the non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read.	
3	R	UART_FE Framing Error bit. This is used to indicate the occurrence of a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP bit in the received data. In the FIFO mode, becasue the framing error is associated with a character received, it is revealed when the character with the framing error is at the top of the FIFO. When a framing error occurs, the UART tries to resynchronize. It does this by assuming that the error was due to the start bit of the next character and then continues receiving the other bit, that is data, and/or parity and stop. It should be noted that the Framing Error (FE) bit (LSR[3]) is set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). 0 = No framing error 1 = Framing error Reading the LSR clears the FE bit.	0x0
2	R	UART_PE Parity Error bit. This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set. In the FIFO mode, because the parity error is associated with a character received, it is revealed when the character with the parity error arrives at the top of the FIFO. It should be noted that the Parity Error (PE) bit (LSR[2]) is set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). 0 = No parity error 1 = Parity error Reading the LSR clears the PE bit.	0x0
1	R	UART_OE Overrun error bit. This is used to indicate the occurrence of an overrun error. This occurs if a new data character was received before the previous data was read. In the non-FIFO mode, the OE bit is set when a new character arrives in the receiver before the previous character was read from the RBR. When this happens, the data in the RBR is overwritten. In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. The data in the FIFO is retained and the data in the receive shift register is lost. 0 = No overrun error 1 = Overrun error	0x0

Bit	Mode	Symbol/Description	Reset
		Reading the LSR clears the OE bit.	
0	R	UART_DR Data Ready bit. This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO. 0 = No data ready 1 = Data ready This bit is cleared when the RBR is read in non-FIFO mode, or when the receiver FIFO is empty, in FIFO mode.	0x0

Table 358: UART_MSR_REG (0x50001018)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4	R	UART_CTS Clear to Send. This is used to indicate the current state of the modem control line cts_n. This bit is the complement of cts_n. When the Clear to Send input (cts_n) is asserted it is an indication that the modem or data set is ready to exchange data with the UART Ctrl. 0 = cts_n input is de-asserted (logic 1) 1 = cts_n input is asserted (logic 0) In Loopback Mode (MCR[4] = 1), CTS is the same as MCR[1] (RTS).	0x1
3:1	-	- Reserved	0x0
0	-	- Reserved	0x0

Table 359: UART_SCR_REG (0x5000101C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	UART_SCRATCH_PAD This register is for programmers to use as a temporary storage space. It has no defined purpose in the UART Ctrl.	0x0

Table 360: UART_SRBR_STHR0_REG (0x50001030)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data	0x0

Bit	Mode	Symbol/Description	Reset
		in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 361: UART_SRBR_STHR1_REG (0x50001034)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 362: UART_SRBR_STHR2_REG (0x50001038)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx	0x0

Bit	Mode	Symbol/Description	Reset
		Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 363: UART_SRBR_STHR3_REG (0x5000103C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 364: UART_SRBR_STHR4_REG (0x50001040)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 365: UART_SRBR_STHR5_REG (0x50001044)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined	0x0

Bit	Mode	Symbol/Description	Reset
		by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 366: UART_SRBR_STHR6_REG (0x50001048)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 367: UART_SRBR_STHR7_REG (0x5000104C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single	0x0

Bit	Mode	Symbol/Description	Reset
		character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 368: UART_SRBR_STHR8_REG (0x50001050)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 369: UART_SRBR_STHR9_REG (0x50001054)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses	0x0

Bit	Mode	Symbol/Description	Reset
		from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 370: UART_SRBR_STHR10_REG (0x50001058)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 371: UART_SRBR_STHR11_REG (0x5000105C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read	0x0

Bit	Mode	Symbol/Description	Reset
		before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 372: UART_SRBR_STHR12_REG (0x50001060)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 373: UART_SRBR_STHR13_REG (0x50001064)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register	0x0

Bit	Mode	Symbol/Description	Reset
		(LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 374: UART_SRBR_STHR14_REG (0x50001068)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 375: UART_SRBR_STHR15_REG (0x5000106C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx	0x0

Bit	Mode	Symbol/Description	Reset
		Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 376: UART_FAR_REG (0x50001070)

Bit	Mode	Symbol/Description	Reset
0	R	UART_FAR Description: Writes will have no effect when FIFO_ACCESS == No, always readable. This register is use to enable a FIFO access mode for testing, so that the receive FIFO can be written by the master and the transmit FIFO can be read by the master when FIFO's are implemented and enabled. When FIFO's are not implemented or not enabled it allows the RBR to be written by the master and the THR to be read by the master. 0 = FIFO access mode disabled 1 = FIFO access mode enabled Note, that when the FIFO access mode is enabled/disabled, the control portion of the receive FIFO and transmit FIFO is reset and the FIFO's are treated as empty.	0x0

Table 377: UART_USR_REG (0x5000107C)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4	R	UART_RFF Receive FIFO Full. This is used to indicate that the receive FIFO is completely full. 0 = Receive FIFO not full 1 = Receive FIFO Full This bit is cleared when the RX FIFO is no longer full.	0x0
3	R	UART_RFNE Receive FIFO Not Empty. This is used to indicate that the receive FIFO contains one or more entries. 0 = Receive FIFO is empty	0x0

Bit	Mode	Symbol/Description	Reset
		1 = Receive FIFO is not empty This bit is cleared when the RX FIFO is empty.	
2	R	UART_TFE Transmit FIFO Empty. This is used to indicate that the transmit FIFO is completely empty. 0 = Transmit FIFO is not empty 1 = Transmit FIFO is empty This bit is cleared when the TX FIFO is no longer empty.	0x1
1	R	UART_TFNF Transmit FIFO Not Full. This is used to indicate that the transmit FIFO is not full. 0 = Transmit FIFO is full 1 = Transmit FIFO is not full This bit is cleared when the TX FIFO is full.	0x1
0	R	UART_BUSY UART Busy. This indicates that a serial transfer is in progress, when cleared indicates that the DW_apb_uart is idle or inactive. 0 - DW_apb_uart is idle or inactive 1 - DW_apb_uart is busy (actively transferring data) Note that it is possible for the UART Busy bit to be cleared even though a new character may have been sent from another device. That is, if the DW_apb_uart has no data in the THR and RBR and there is no transmission in progress and a start bit of a new character has just reached the DW_apb_uart. This is due to the fact that a valid start is not seen until the middle of the bit period and this duration is dependent on the baud divisor that has been programmed. If a second system clock has been implemented (CLOCK_MODE == Enabled) the assertion of this bit will also be delayed by several cycles of the slower clock.	0x0

Table 378: UART_TFL_REG (0x50001080)

Bit	Mode	Symbol/Description	Reset
4:0	R	UART_TRANSMIT_FIFO_LEVEL Transmit FIFO Level. This indicates the number of data entries in the transmit FIFO.	0x0

Table 379: UART_RFL_REG (0x50001084)

Bit	Mode	Symbol/Description	Reset
4:0	R	UART_RECEIVE_FIFO_LEVEL Receive FIFO Level. This is indicates the number of data entries in the receive FIFO.	0x0

Table 380: UART_SRR_REG (0x50001088)

Bit	Mode	Symbol/Description	Reset
15:3	-	- Reserved	0x0
2	W	UART_XFR XMIT FIFO Reset.	0x0

Bit	Mode	Symbol/Description	Reset
		This is a shadow register for the XMIT FIFO Reset bit (FCR[2]). This can be used to remove the burden on software having to store previously written FCR values (which are pretty static) just to reset the transmit FIFO. This resets the control portion of the transmit FIFO and treats the FIFO as empty. Note that this bit is 'self-clearing'. It is not necessary to clear this bit.	
1	W	UART_RFR RCVR FIFO Reset. This is a shadow register for the RCVR FIFO Reset bit (FCR[1]). This can be used to remove the burden on software having to store previously written FCR values (which are pretty static) just to reset the receive FIFO. This resets the control portion of the receive FIFO and treats the FIFO as empty. Note that this bit is 'self-clearing'. It is not necessary to clear this bit.	0x0
0	W	UART_UR UART Reset. This asynchronously resets the UART Ctrl and synchronously removes the reset assertion. For a two clock implementation both pclk and sclk domains are reset.	0x0

Table 381: UART_SRTS_REG (0x5000108C)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_SHADOW_REQUEST_TO_SEND Shadow Request to Send. This is a shadow register for the RTS bit (MCR[1]), this can be used to remove the burden of having to performing a read-modify-write on the MCR. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART Ctrl is ready to exchange data. When Auto RTS Flow Control is not enabled (MCR[5] = 0), the rts_n signal is set low by programming MCR[1] (RTS) to a high. In Auto Flow Control, AFCE_MODE == Enabled and active (MCR[5] = 1) and FIFOs enable (FCR[0] = 1), the rts_n output is controlled in the same way, but is also gated with the receiver FIFO threshold trigger (rts_n is inactive high when above the threshold). Note that in Loopback mode (MCR[4] = 1), the rts_n output is held inactive-high while the value of this location is internally looped back to an input.	0x0

Table 382: UART_SBCR_REG (0x50001090)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_SHADOW_BREAK_CONTROL Shadow Break Control Bit. This is a shadow register for the Break bit (LCR[6]), this can be used to remove the burden of having to perform a read modify write on the LCR. This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared.	0x0

Bit	Mode	Symbol/Description	Reset
		If SIR_MODE active (MCR[6] = 1) the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver.	

Table 383: UART_SDMAM_REG (0x50001094)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_SHADOW_DMA_MODE Shadow DMA Mode. This is a shadow register for the DMA mode bit (FCR[3]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the DMA Mode bit gets updated. This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals. 0 = Mode 0 1 = Mode 1	0x0

Table 384: UART_SFE_REG (0x50001098)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_SHADOW_FIFO_ENABLE Shadow FIFO Enable. This is a shadow register for the FIFO enable bit (FCR[0]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the FIFO enable bit gets updated. This enables/disables the transmit (XMIT) and receive (RCVR) FIFOs. If this bit is set to zero (disabled) after being enabled then both the XMIT and RCVR controller portion of FIFOs are reset.	0x0

Table 385: UART_SRT_REG (0x5000109C)

Bit	Mode	Symbol/Description	Reset
15:2	-	- Reserved	0x0
1:0	R/W	UART_SHADOW_RCVR_TRIGGER Shadow RCVR Trigger. This is a shadow register for the RCVR trigger bits (FCR[7:6]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the RCVR trigger bit gets updated. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt is generated. It also determines when the dma_rx_req_n signal is asserted when DMA Mode (FCR[3]) = 1. The following trigger levels are supported: 00 = 1 character in the FIFO 01 = FIFO ¼ full 10 = FIFO ½ full	0x0

Bit	Mode	Symbol/Description	Reset
		11 = FIFO 2 less than full	

Table 386: UART_STET_REG (0x500010A0)

Bit	Mode	Symbol/Description	Reset
15:2	-	- Reserved	0x0
1:0	R/W	UART_SHADOW_TX_EMPTY_TRIGGER Shadow TX Empty Trigger. This is a shadow register for the TX empty trigger bits (FCR[5:4]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the TX empty trigger bit gets updated. This is used to select the empty threshold level at which the THRE Interrupts are generated when the mode is active. The following trigger levels are supported: 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO ¼ full 11 = FIFO ½ full	0x0

Table 387: UART_HTX_REG (0x500010A4)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_HALT_TX This register is used to halt transmissions for testing, so that the transmit FIFO can be filled by the master when FIFOs are implemented and enabled. 0 = Halt TX disabled 1 = Halt TX enabled Note, if FIFOs are implemented and not enabled, the setting of the halt TX register has no effect on operation.	0x0

Table 388: UART_DMASA_REG (0x500010A8)

Bit	Mode	Symbol/Description	Reset
0	W	DMASA This register is use to perform DMA software acknowledge if a transfer needs to be terminated due to an error condition. For example, if the DMA disables the channel, then the DW_apb_uart should clear its request. This will cause the TX request, TX single, RX request and RX single signals to de-assert. Note that this bit is 'self-clearing' and it is not necessary to clear this bit.	0x0

Table 389: UART_DLF_REG (0x500010C0)

Bit	Mode	Symbol/Description	Reset
3:0	R/W	UART_DLF The fractional value is added to integer value set by DLH, DLL. Fractional value is equal UART_DLF/16	0x0

Table 390: UART_UCV_REG (0x500010F8)

Bit	Mode	Symbol/Description	Reset
15:0	R	UCV Component Version	0x352A

Table 391: UART_UCV_HIGH_REG (0x500010FA)

Bit	Mode	Symbol/Description	Reset
15:0	R	UCV Component Version	0x3331

Table 392: UART_CTR_REG (0x500010FC)

Bit	Mode	Symbol/Description	Reset
15:0	R	CTR Component Type Register	0x110

Table 393: UART_CTR_HIGH_REG (0x500010FE)

Bit	Mode	Symbol/Description	Reset
15:0	R	CTR Component Type Register	0x4457

Table 394: UART2_RBR_THR_DLL_REG (0x50001100)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	RBR_THR_DLL **Receive Buffer Register: (RBR).** This register contains the data byte received on the serial input port (sin) in UART mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise, it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. **Transmit Holding Register: (THR)** This register contains data to be transmitted on the serial output port (sout) in UART mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, 16 number of characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost. **Divisor Latch (Low): (DLL)**	0x0

Bit	Mode	Symbol/Description	Reset
		This register makes up the lower 8-bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may only be accessed when the DLAB bit (LCR[7]) is set. The output baud rate is equal to the serial clock (sclk) frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq)/(16 * divisor) Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications will occur. Also, when the Divisor Latch is set, at least eight clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data. For the Divisor Latch (High) bits, see the UART_IER_DLH_REG register.	

Table 395: UART2_IER_DLH_REG (0x50001104)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7	R/W	PTIME_dlh7 **Interrupt Enable Register: PTIME**, Programmable THRE Interrupt Mode Enable. This is used to enable/disable the generation of THRE Interrupt. 0 = Disabled 1 = Enabled. **Divisor Latch (High): DLH7**, Bit 7 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0
6:4	R/W	dlh6_4 **Divisor Latch (High): DLH6 to DLH4**, Bits 6 to 4 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set, otherwise, this field is reserved. See the UART_RBR_THR_DLL_REG register.	0x0
3	R/W	EDSSI_dlh3 **Interrupt Enable Register: EDSSI**, Enable Modem Status Interrupt. This is used to enable/disable the generation of Modem Status Interrupt. This is the fourth highest priority interrupt. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH3**, Bit 3 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0
2	R/W	ELSI_dhl2 **Interrupt Enable Register: ELSI**, Enable Receiver Line Status Interrupt. This is used to enable/disable the generation of Receiver Line Status Interrupt. This is the highest priority interrupt. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH2**, Bit 2 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0

Bit	Mode	Symbol/Description	Reset
1	R/W	ETBEI_dlh1 **Interrupt Enable Register: ETBEI**, Enable Transmit Holding Register Empty Interrupt. This is used to enable/disable the generation of Transmitter Holding Register Empty Interrupt. This is the third highest priority interrupt. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH1**, Bit 1 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0
0	R/W	ERBFI_dlh0 **Interrupt Enable Register: ERBFI**, Enable Received Data Available Interrupt. This is used to enable/disable the generation of Received Data Available Interrupt and the Character Timeout Interrupt (if in FIFO mode and FIFO's enabled). These are the second highest priority interrupts. 0 = Disabled 1 = Enabled **Divisor Latch (High): DLH0**, Bit 0 of the upper part of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may be accessed only when the DLAB bit (LCR[7]) is set. See the UART_RBR_THR_DLL_REG register.	0x0

Table 396: UART2_IIR_FCR_REG (0x50001108)

Bit	Mode	Symbol/Description	Reset
7:6	R/W	UART_FIFOSE_RT On read FIFO's Enabled (or FIFOSE): This is used to indicate whether the FIFO's are enabled or disabled. 00 = Disabled. 11 = Enabled. On write RCVR Trigger (or RT):. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt will be generated. In auto flow control mode it is used to determine when the rts_n signal will be de-asserted. It also determines when the dma_rx_req_n signal will be asserted when in certain modes of operation. The following trigger levels are supported: 00 = 1 character in the FIFO 01 = FIFO 1/4 full 10 = FIFO 1/2 full 11 = FIFO 2 less than full	0x0
5:4	R0/W	UART_TET On read Reserved On Write TX Empty Trigger (or TET): This is used to select the empty threshold level at which the THRE Interrupts will be generated when the mode is active. It also determines when the dma_tx_req_n signal will be asserted when in certain modes of operation. The following trigger levels are supported: 00 = FIFO empty 01 = 2 characters in the FIFO	0x0

Bit	Mode	Symbol/Description	Reset
		10 = FIFO 1/4 full 11 = FIFO 1/2 full	
3	R/W	UART_IID3_DMAM On Read (Bit3) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect. 1100 = Character timeout. On Write DMA Mode (or DMAM): This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals. 0 = Mode 0 1 = Mode 1	0x0
2	R/W	UART_IID2_XFIFOR On Read (Bit2) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect. 1100 = Character timeout. On Write XMIT FIFO Reset (or XFIFOR): This resets the control portion of the transmit FIFO and treats the FIFO as empty. Note that this bit is "self-clearing" and it is not necessary to clear this bit.	0x0
1	R/W	UART_IID1_RFIFOE On Read (Bit1) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect. 1100 = Character timeout. On Write RCVR FIFO Reset (or RFIFOR): This resets the control portion of the receive FIFO and treats the FIFO as empty. Note that this bit is "self-clearing" and it is not necessary to clear this bit.	0x0
0	R/W	UART_IID0_FIFOE On Read (Bit0) Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types:	0x1

Bit	Mode	Symbol/Description	Reset
		0001 = No interrupt pending. 0010 = THR empty. 0100 = Received data available. 0110 = Receiver line status. 0111 = Busy detect. 1100 = Character timeout. On Write FIFO Enable (or FIFOE): This enables/disables the transmit (XMIT) and receive (RCVR) FIFO's. Whenever the value of this bit is changed both the XMIT and RCVR controller portion of FIFO's will be reset	

Table 397: UART2_LCR_REG (0x5000110C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7	R/W	UART_DLAB Divisor Latch Access Bit.Writeable only when UART is not busy (USR[0] is zero). This bit is used to enable reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after the initial baud rate setup to access other registers.	0x0
6	R/W	UART_BC Break Control Bit. This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. If active (MCR[6] set to one) the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver and the sir_out_n line is forced low.	0x0
5	-	- Reserved	0x0
4	R/W	UART_EPS Even Parity Select. Writeable only when UART is not busy (USR[0] is zero). This is used to select between even and odd parity, when parity is enabled (PEN set to one). If set to one, an even number of logic 1s is transmitted or checked. If set to zero, an odd number of logic 1s is transmitted or checked.	0x0
3	R/W	UART_PEN Parity Enable. Writeable only when UART is not busy (USR[0] is zero) This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively. 0 = parity disabled 1 = parity enabled	0x0
2	R/W	UART_STOP Number of stop bits. Writeable only when UART is not busy (USR[0] is zero). This is used to select the number of stop bits per character that the peripheral transmits and receives. If set to zero, one stop bit is transmitted in the serial data. If set to one and the data bits are set to 5 (LCR[1:0] set to zero) one and a half stop bits is transmitted. Otherwise, two stop bits are transmitted. Note that	0x0

Bit	Mode	Symbol/Description	Reset
		regardless of the number of stop bits selected, the receiver checks only the first stop bit. 0 = 1 stop bit 1 = 1.5 stop bits when DLS (LCR[1:0]) is zero, else 2 stop bit	
1:0	R/W	UART_DLS Data Length Select. Writeable only when UART is not busy (USR[0] is zero). This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bits that may be selected areas follows: 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits	0x0

Table 398: UART2_MCR_REG (0x50001110)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4	R/W	UART_LB LoopBack Bit. This is used to put the UART into a diagnostic mode for test purposes. If operating in UART mode (SIR_MODE not active, MCR[6] set to zero), data on the sout line is held high, while serial data output is looped back to the sin line, internally. In this mode all the interrupts are fully functional. Also, in loopback mode, the modem control inputs (dsr_n, cts_n, ri_n, dcd_n) are disconnected and the modem control outputs (dtr_n, rts_n, out1_n, out2_n) are looped back to the inputs, internally. If operating in infrared mode (SIR_MODE active, MCR[6] set to one), data on the sir_out_n line is held low, while serial data output is inverted and looped back to the sir_in line.	0x0
3:0	-	- Reserved	0x0

Table 399: UART2_LSR_REG (0x50001114)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7	R	UART_RFE Receiver FIFO Error bit. This bit is only relevant when FIFOs are enabled (FCR[0] set to one). This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO. 0 = No error in RX FIFO 1 = Error in RX FIFO This bit is cleared when the LSR is read and the character with the error is at the top of the receiver FIFO and there are no subsequent errors in the FIFO.	0x0
6	R	UART_TEMT Transmitter Empty bit.	0x1

Bit	Mode	Symbol/Description	Reset
		If FIFOs enabled (FCR[0] set to one), this bit is set whenever the Transmitter Shift Register and the FIFO are both empty. If FIFOs are disabled, this bit is set whenever the Transmitter Holding Register(THR) and the Transmitter Shift Register are both empty.	
5	R	UART_THRE Transmit Holding Register Empty bit. If THRE mode is disabled (IER[7] set to zero) and regardless of FIFO's being implemented/enabled or not, this bit indicates that the THR or TX FIFO is empty. This bit is set whenever data is transferred from the THR or TX FIFO to the transmitter shift register and no new data has been written to the THR or TX FIFO. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. If both modes are active (IER[7] set to one and FCR[0] set to one respectively), the functionality is switched to indicate the transmitter FIFO is full, and no longer controls THRE interrupts, which are then controlled by the FCR[5:4] threshold setting.	0x1
4	R	UART_BI Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data. If in UART mode (SIR_MODE == Disabled), it is set whenever the serial input, sin, is held in a logic "0" state for longer than the sum of start time + data bits + parity + stop bits. If in infrared mode (SIR_MODE == Enabled), it is set whenever the serial input, sir_in, is continuously pulsed to logic "0" for longer than the sum of start time + data bits + parity + stop bits. A break condition on serial input causes one and only one character, consisting of all zeros, to be received by the UART. In the FIFO mode, the character associated with the break condition is carried through the FIFO and is revealed when the character is at the top of the FIFO. Reading the LSR clears the BI bit. In the non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read.	0x0
3	R	UART_FE Framing Error bit. This is used to indicate the occurrence of a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP bit in the received data. In the FIFO mode, because the framing error is associated with a character received, it is revealed when the character with the framing error is at the top of the FIFO. When a framing error occurs, the UART tries to resynchronize. It does this by assuming that the error was due to the start bit of the next character and then continues receiving the other bit, that is data, and/or parity and stop. It should be noted that the Framing Error (FE) bit (LSR[3]) is set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). 0 = No framing error 1 = Framing error Reading the LSR clears the FE bit.	0x0
2	R	UART_PE Parity Error bit. This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set. In the FIFO mode, because the parity error is associated with a character received, it is revealed when the character with the parity error arrives at the top of the FIFO. It should be noted that the Parity Error (PE) bit (LSR[2]) is set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]).	0x0

Bit	Mode	Symbol/Description	Reset
		0 = No parity error 1 = Parity error Reading the LSR clears the PE bit.	
1	R	UART_OE Overrun error bit. This is used to indicate the occurrence of an overrun error. This occurs if a new data character was received before the previous data was read. In the non-FIFO mode, the OE bit is set when a new character arrives in the receiver before the previous character was read from the RBR. When this happens, the data in the RBR is overwritten. In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. The data in the FIFO is retained and the data in the receive shift register is lost. 0 = No overrun error 1 = Overrun error Reading the LSR clears the OE bit.	0x0
0	R	UART_DR Data Ready bit. This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO. 0 = No data ready 1 = Data ready This bit is cleared when the RBR is read in non-FIFO mode, or when the receiver FIFO is empty, in FIFO mode.	0x0

Table 400: UART2_SCR_REG (0x5000111C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	UART_SCRATCH_PAD This register is for programmers to use as a temporary storage space. It has no defined purpose in the UART Ctrl.	0x0

Table 401: UART2_SRBR_STHR0_REG (0x50001130)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be	0x0

Bit	Mode	Symbol/Description	Reset
		preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 402: UART2_SRBR_STHR1_REG (0x50001134)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 403: UART2_SRBR_STHR2_REG (0x50001138)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must	0x0

Bit	Mode	Symbol/Description	Reset
		be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 404: UART2_SRBR_STHR3_REG (0x5000113C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 405: UART2_SRBR_STHR4_REG (0x50001140)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from	0x0

Bit	Mode	Symbol/Description	Reset
		the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 406: UART2_SRBR_STHR5_REG (0x50001144)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 407: UART2_SRBR_STHR6_REG (0x50001148)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0

Bit	Mode	Symbol/Description	Reset
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 408: UART2_SRBR_STHR7_REG (0x5000114C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 409: UART2_SRBR_STHR8_REG (0x50001150)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 410: UART2_SRBR_STHR9_REG (0x50001154)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined	0x0

Bit	Mode	Symbol/Description	Reset
		by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 411: UART2_SRBR_STHR10_REG (0x50001158)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 412: UART2_SRBR_STHR11_REG (0x5000115C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single	0x0

Bit	Mode	Symbol/Description	Reset
		character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 413: UART2_SRBR_STHR12_REG (0x50001160)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 414: UART2_SRBR_STHR13_REG (0x50001164)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses	0x0

Bit	Mode	Symbol/Description	Reset
		from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 415: UART2_SRBR_STHR14_REG (0x50001168)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	0x0

Table 416: UART2_SRBR_STHR15_REG (0x5000116C)

Bit	Mode	Symbol/Description	Reset
15:8	-	- Reserved	0x0
7:0	R/W	SRBR_STHRx Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read	0x0

Bit	Mode	Symbol/Description	Reset
		before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Shadow Transmit Holding Register 0: This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFO's are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.	

Table 417: UART2_FAR_REG (0x50001170)

Bit	Mode	Symbol/Description	Reset
0	R	UART_FAR Description: Writes will have no effect when FIFO_ACCESS == No, always readable. This register is use to enable a FIFO access mode for testing, so that the receive FIFO can be written by the master and the transmit FIFO can be read by the master when FIFO's are implemented and enabled. When FIFO's are not implemented or not enabled it allows the RBR to be written by the master and the THR to be read by the master. 0 = FIFO access mode disabled 1 = FIFO access mode enabled Note, that when the FIFO access mode is enabled/disabled, the control portion of the receive FIFO and transmit FIFO is reset and the FIFO's are treated as empty.	0x0

Table 418: UART2_USR_REG (0x5000117C)

Bit	Mode	Symbol/Description	Reset
15:5	-	- Reserved	0x0
4	R	UART_RFF Receive FIFO Full. This is used to indicate that the receive FIFO is completely full. 0 = Receive FIFO not full 1 = Receive FIFO Full This bit is cleared when the RX FIFO is no longer full.	0x0
3	R	UART_RFNE Receive FIFO Not Empty. This is used to indicate that the receive FIFO contains one or more entries. 0 = Receive FIFO is empty 1 = Receive FIFO is not empty This bit is cleared when the RX FIFO is empty.	0x0
2	R	UART_TFE Transmit FIFO Empty. This is used to indicate that the transmit FIFO is completely empty. 0 = Transmit FIFO is not empty 1 = Transmit FIFO is empty	0x1

Bit	Mode	Symbol/Description	Reset
		This bit is cleared when the TX FIFO is no longer empty.	
1	R	UART_TFNF Transmit FIFO Not Full. This is used to indicate that the transmit FIFO is not full. 0 = Transmit FIFO is full 1 = Transmit FIFO is not full This bit is cleared when the TX FIFO is full.	0x1
0	R	UART_BUSY UART Busy. This indicates that a serial transfer is in progress, when cleared indicates that the DW_apb_uart is idle or inactive. 0 - DW_apb_uart is idle or inactive 1 - DW_apb_uart is busy (actively transferring data) Note that it is possible for the UART Busy bit to be cleared even though a new character may have been sent from another device. That is, if the DW_apb_uart has no data in the THR and RBR and there is no transmission in progress and a start bit of a new character has just reached the DW_apb_uart. This is due to the fact that a valid start is not seen until the middle of the bit period and this duration is dependent on the baud divisor that has been programmed. If a second system clock has been implemented (CLOCK_MODE == Enabled) the assertion of this bit will also be delayed by several cycles of the slower clock.	0x0

Table 419: UART2_TFL_REG (0x50001180)

Bit	Mode	Symbol/Description	Reset
4:0	R	UART_TRANSMIT_FIFO_LEVEL Transmit FIFO Level. This indicates the number of data entries in the transmit FIFO.	0x0

Table 420: UART2_RFL_REG (0x50001184)

Bit	Mode	Symbol/Description	Reset
4:0	R	UART_RECEIVE_FIFO_LEVEL Receive FIFO Level. This is indicates the number of data entries in the receive FIFO.	0x0

Table 421: UART2_SRR_REG (0x50001188)

Bit	Mode	Symbol/Description	Reset
15:3	-	- Reserved	0x0
2	W	UART_XFR XMIT FIFO Reset. This is a shadow register for the XMIT FIFO Reset bit (FCR[2]). This can be used to remove the burden on software having to store previously written FCR values (which are pretty static) just to reset the transmit FIFO. This resets the control portion of the transmit FIFO and treats the FIFO as empty. Note that this bit is 'self-clearing'. It is not necessary to clear this bit.	0x0
1	W	UART_RFR RCVR FIFO Reset. This is a shadow register for the RCVR FIFO Reset bit (FCR[1]). This can be used to remove the burden on software having to store previously written FCR values	0x0

Bit	Mode	Symbol/Description	Reset
		(which are pretty static) just to reset the receive FIFO. This resets the control portion of the receive FIFO and treats the FIFO as empty. Note that this bit is 'self-clearing'. It is not necessary to clear this bit.	
0	W	UART_UR UART Reset. This asynchronously resets the UART Ctrl and synchronously removes the reset assertion. For a two clock implementation both pclk and sclk domains are reset.	0x0

Table 422: UART2_SBCR_REG (0x50001190)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_SHADOW_BREAK_CONTROL Shadow Break Control Bit. This is a shadow register for the Break bit (LCR[6]), this can be used to remove the burden of having to perform a read modify write on the LCR. This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. If SIR_MODE active (MCR[6] = 1) the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver.	0x0

Table 423: UART2_SDMAM_REG (0x50001194)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_SHADOW_DMA_MODE Shadow DMA Mode. This is a shadow register for the DMA mode bit (FCR[3]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the DMA Mode bit gets updated. This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals. 0 = Mode 0 1 = Mode 1	0x0

Table 424: UART2_SFE_REG (0x50001198)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_SHADOW_FIFO_ENABLE Shadow FIFO Enable. This is a shadow register for the FIFO enable bit (FCR[0]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the FIFO enable bit gets updated. This enables/disables the transmit (XMIT) and receive (RCVR) FIFOs. If	0x0

Bit	Mode	Symbol/Description	Reset
		this bit is set to zero (disabled) after being enabled then both the XMIT and RCVR controller portion of FIFOs are reset.	

Table 425: UART2_SRT_REG (0x5000119C)

Bit	Mode	Symbol/Description	Reset
15:2	-	- Reserved	0x0
1:0	R/W	UART_SHADOW_RCVR_TRIGGER Shadow RCVR Trigger. This is a shadow register for the RCVR trigger bits (FCR[7:6]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the RCVR trigger bit gets updated. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt is generated. It also determines when the dma_rx_req_n signal is asserted when DMA Mode (FCR[3]) = 1. The following trigger levels are supported: 00 = 1 character in the FIFO 01 = FIFO ¼ full 10 = FIFO ½ full 11 = FIFO 2 less than full	0x0

Table 426: UART2_STET_REG (0x500011A0)

Bit	Mode	Symbol/Description	Reset
15:2	-	- Reserved	0x0
1:0	R/W	UART_SHADOW_TX_EMPTY_TRIGGER Shadow TX Empty Trigger. This is a shadow register for the TX empty trigger bits (FCR[5:4]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the TX empty trigger bit gets updated. This is used to select the empty threshold level at which the THRE Interrupts are generated when the mode is active. The following trigger levels are supported: 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO ¼ full 11 = FIFO ½ full	0x0

Table 427: UART2_HTX_REG (0x500011A4)

Bit	Mode	Symbol/Description	Reset
15:1	-	- Reserved	0x0
0	R/W	UART_HALT_TX This register is used to halt transmissions for testing, so that the transmit FIFO can be filled by the master when FIFOs are implemented and enabled.	0x0

Bit	Mode	Symbol/Description	Reset
		0 = Halt TX disabled 1 = Halt TX enabled Note, if FIFOs are implemented and not enabled, the setting of the halt TX register has no effect on operation.	

Table 428: UART2_DMASA_REG (0x500011A8)

Bit	Mode	Symbol/Description	Reset
0	W	DMASA This register is use to perform DMA software acknowledge if a transfer needs to be terminated due to an error condition. For example, if the DMA disables the channel, then the DW_apb_uart should clear its request. This will cause the TX request, TX single, RX request and RX single signals to de-assert. Note that this bit is 'self-clearing' and it is not necessary to clear this bit.	0x0

Table 429: UART2_DLF_REG (0x500011C0)

Bit	Mode	Symbol/Description	Reset
3:0	R/W	UART_DLF The fractional value is added to integer value set by DLH, DLL. Fractional value is equal UART_DLF/16	0x0

Table 430: UART2_UCV_REG (0x500011F8)

Bit	Mode	Symbol/Description	Reset
15:0	R	UCV Component Version	0x352A

Table 431: UART2_UCV_HIGH_REG (0x500011FA)

Bit	Mode	Symbol/Description	Reset
15:0	R	UCV Component Version	0x3331

Table 432: UART2_CTR_REG (0x500011FC)

Bit	Mode	Symbol/Description	Reset
15:0	R	CTR Component Type Register	0x110

Table 433: UART2_CTR_HIGH_REG (0x500011FE)

Bit	Mode	Symbol/Description	Reset
15:0	R	CTR Component Type Register	0x4457

31.19 Chip Version Registers

Table 434: Register map Version

Address	Register	Description
0x50003200	CHIP_ID1_REG	Chip identification register 1.
0x50003204	CHIP_ID2_REG	Chip identification register 2.
0x50003208	CHIP_ID3_REG	Chip identification register 3.
0x5000320c	CHIP_ID4_REG	Chip identification register 4.

Table 435: CHIP_ID1_REG (0x50003200)

Bit	Mode	Symbol/Description	Reset
7:0	R	CHIP_ID1 First character of device type "3081" in ASCII.	0x33

Table 436: CHIP_ID2_REG (0x50003204)

Bit	Mode	Symbol/Description	Reset
7:0	R	CHIP_ID2 Second character of device type "3081" in ASCII.	0x30

Table 437: CHIP_ID3_REG (0x50003208)

Bit	Mode	Symbol/Description	Reset
7:0	R	CHIP_ID3 Third character of device type "3081" in ASCII.	0x38

Table 438: CHIP_ID4_REG (0x5000320C)

Bit	Mode	Symbol/Description	Reset
7:0	R	CHIP_ID4 Fourth character of device type "3081" in ASCII.	0x31

31.20 Wake-up Registers

Table 439: Register map WKUP

Address	Register	Description
0x50000100	WKUP_CTRL_REG	Control register for the wake-up counter
0x50000102	WKUP_COMPARE_REG	Number of events before wake-up interrupt
0x50000104	WKUP_IRQ_STATUS_REG	Reset wake-up interrupt
0x50000106	WKUP_COUNTER_REG	Actual number of events of the wake-up counter
0x50000108	WKUP_SELECT_GPIO_REG	Select which inputs from P0 port can trigger wake-up counter
0x5000010a	WKUP2_SELECT_GPIO_REG	Select which inputs from P1 port can trigger wake-up counter
0x5000010c	WKUP_POL_GPIO_REG	Select the sensitivity polarity for each P0 input
0x5000010e	WKUP2_POL_GPIO_REG	Select the sensitivity polarity for each P1 input

Table 440: WKUP_CTRL_REG (0x50000100)

Bit	Mode	Symbol/Description	Reset
9	R/W	ENABLE_125US_CLK 0 = Debounce unit time 1 ms 1 = Debounce unit time 125 μs	0x0
8	R/W	WKUP2_ENABLE_IRQ 0 = No interrupt will be generated 1 = If the event counter2 reaches the value set by WKUP_COMPARE_REG an IRQ will be generated	0x0
7	R/W	WKUP_ENABLE_IRQ 0 = No interrupt will be generated 1 = If the event counter reaches the value set by WKUP_COMPARE_REG an IRQ will be generated	0x0
6	R/W	WKUP_SFT_KEYHIT 0 = No effect 1 = Emulate key hit. The event counter and counter2 will increment by 1 (after debouncing if enabled). First make this bit 0 before any new key hit can be sensed.	0x0
5:0	R/W	WKUP_DEB_VALUE Keyboard debounce time (N*1 ms with N = 1 to 63). 0x0: No debouncing 0x1 to 0x3F: 1 ms to 63 ms debounce time (when ENABLE_125US_CLK = 0) else 125 μs to 7.875 ms)	0x0

Table 441: WKUP_COMPARE_REG (0x50000102)

Bit	Mode	Symbol/Description	Reset
7:0	R/W	WKUP_COMPARE Defines the number of events -1 that have to be counted before the wake-up interrupt will be given. value 0 means one event.	0x0

Table 442: WKUP_IRQ_STATUS_REG (0x50000104)

Bit	Mode	Symbol/Description	Reset
3	R0/W	WKUP2_CNTR_RST Writing 1 resets the event2 counter	0x0
2	R0/W	WKUP_CNTR_RST writing 1 resets the event counter	0x0
1	R/W	WKUP2_IRQ_STATUS Gives 1 when there is a wkup2 pending IRQ. Writing 1 resets the interrupt.	0x0
0	R/W	WKUP_IRQ_STATUS Gives 1 when there is a wkup pending IRQ. Writing 1 resets the interrupt.	0x0

Table 443: WKUP_COUNTER_REG (0x50000106)

Bit	Mode	Symbol/Description	Reset
15:8	R	EVENT2_VALUE This value represents the number of events that have been counted so far. It will be reset by writing to the WKUP_CNTR_RST bit field of WKUP_IRQ_STATUS_REG.	0x0
7:0	R	EVENT_VALUE This value represents the number of events that have been counted so far. It will be reset by writing to the WKUP_CNTR_RST bit field of WKUP_IRQ_STATUS_REG.	0x0

Table 444: WKUP_SELECT_GPIO_REG (0x50000108)

Bit	Mode	Symbol/Description	Reset
11:0	R/W	WKUP_SELECT_GPIO 0 = Input P0x is not enabled for wake-up event counter 1 = Input P0x is enabled for wake-up event counter	0x0

Table 445: WKUP2_SELECT_GPIO_REG (0x5000010A)

Bit	Mode	Symbol/Description	Reset
11:0	R/W	WKUP2_SELECT_GPIO 0 = Input P0x is not enabled for wake-up event counter 1 = Input P0x is enabled for wake-up event counter	0x0

Table 446: WKUP_POL_GPIO_REG (0x5000010C)

Bit	Mode	Symbol/Description	Reset
11:0	R/W	WKUP_POL_GPIO 0 = The enabled input P0x increments the event counter if that input goes high 1 = The enabled input P0x increments the event counter if that input goes low	0x0

Table 447: WKUP2_POL_GPIO_REG (0x5000010E)

Bit	Mode	Symbol/Description	Reset
11:0	R/W	WKUP2_POL_GPIO 0 = The enabled input P0x increments the event2 counter if that input goes high 1 = The enabled input P0x increments the event2 counter if that input goes low	0x0

31.21 Watchdog Registers

Table 448: Register map WDOG

Address	Register	Description
0x50003100	WATCHDOG_REG	Watchdog timer register.
0x50003102	WATCHDOG_CTRL_REG	Watchdog control register.

Table 449: WATCHDOG_REG (0x50003100)

Bit	Mode	Symbol/Description	Reset
15:9	R0/W	WDOG_WEN 0000.000 = Write enable for Watchdog timer else Write disable. This filter prevents unintentional presetting the watchdog with a software run-away.	0x0
8	R/W	WDOG_VAL_NEG 0 = Watchdog timer value is positive. 1 = Watchdog timer value is negative.	0x0

7:0	R/W	WDOG_VAL
		Write: Watchdog timer reload value. Note that all bits 15-9 must be 0 to reload this register.
		Read: Actual Watchdog timer value. Decremented by 1 every 10.24 msec. Bit 8 indicates a negative counter value. 2, 1, 0, $1FF_{16}$, $1FE_{16}$ etc. An NMI or WDOG (SYS) reset is generated under the following conditions:
		If WATCHDOG_CTRL_REG[NMI_RST] = 0 then
		If WDOG_VAL = 0 -> NMI (Non Maskable Interrupt)
		if WDOG_VAL = $1F0_{16}$ -> WDOG reset -> reload FF_{16}
		If WATCHDOG_CTRL_REG[NMI_RST] = 1 then
		if WDOG_VAL <= 0 -> WDOG reset -> reload FF_{16}
			0xFF

Table 450: WATCHDOG_CTRL_REG (0x50003102)

Bit	Mode	Symbol/Description	Reset
15:2	-	- Reserved	0x0
1	R/W	- Reserved	0x0
0	R/W	NMI_RST 0 = Watchdog timer generates NMI at value 0, and WDOG (SYS) reset at <=-16. Timer can be frozen/resumed using SET_FREEZE_REG[FRZ_WDOG]/RESET_FREEZE_REG[FRZ_WDOG]. 1 = Watchdog timer generates a WDOG (SYS) reset at value 0 and can not be frozen by software. Note that this bit can only be set to 1 by software and only be reset with a WDOG (SYS) reset or software reset. The watchdog is always frozen when the Cortex-M0 is halted in DEBUG State.	0x0

32. Ordering Information

Table 451: Ordering information (samples)

Part number	Package	Size (mm)	Shipment form	Pack quantity
DA14535-00000FX2	FCGQFN24	2.2 × 3.0	Reel	100/1000

Table 452: Ordering information (production)

Part number	Package	Size (mm)	Shipment form	Pack quantity
DA14535-00000FX2	FCGQFN24	2.2 × 3.0	Reel	4000

Part Number Legend:

DA14535-RRXXXYYZ

RR: chip revision number

XXX: variant (000: No Flash)

YY: package code (FX: FCGQFN24)

Z: packing method (1: Tray, 2: Reel, A: Mini-Reel)

33. Package Information

33.1 Moisture Sensitivity Level (MSL)

The MSL is an indicator for the maximum allowable time period (floor lifetime) in which a moisture sensitive plastic device, once removed from the dry bag, can be exposed to an environment with a maximum temperature of 30 °C and a maximum relative humidity of 60% relative humidity (RH.) before the solder reflow process.

FCGQFN packages are qualified for MSL 1.

Table 453: MSL classification

MSL level	Floor lifetime	Conditions
MSL 4	72 hours	30 °C/60% RH
MSL 3	168 hours	30 °C/60% RH
MSL 2A	4 weeks	30 °C/60% RH
MSL 2	1 year	30 °C/60% RH
MSL 1	Unlimited	30 °C/85% RH

33.2 Soldering Information

Refer to the JEDEC standard J-STD-020 for relevant soldering information. This document can be downloaded from http://www.jedec.org.

33.3 Package Outline

The module's dimensions are accessible from the Renesas website: FCGQFN24 package outline drawing.

34. Revision History

Revision	Date	Description
3.1	May 26, 2025	Converted into Renesas template.
3.0	Jan 5, 2024	Datasheet status: Final. Product status: Production
2.0	July 21, 2023	Datasheet status: Preliminary. Product status: Qualification
1.1	May 24, 2023	Datasheet status: Target. Product status: Development
1.0	June 2, 2022	Datasheet status: Target. Product status: Development

RoHS Compliance

Renesas Electronics' suppliers certify that its products are in compliance with the requirements of Directive 2011/65/EU of the European Parliament on the restriction of the use of certain hazardous substances in electrical and electronic equipment. RoHS certificates from our suppliers are available on request.

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us

Trademarks