

Multi-core Bluetooth 5.2 SoC family with system PMU

General Description

The DA1469x is a family of multi-core wireless microcontrollers, combining the latest Arm® Cortex®-M33 application processor with floating-point unit, advanced power management functionality, a cryptographic security engine, analog and digital peripherals, a dedicated sensor node controller, and a software configurable protocol engine with a radio that is compliant to the Bluetooth® 5.2 low energy standard.

The DA1469x is based on an Arm Cortex-M33 CPU with an 8-region MPU and a single-precision FPU offering up to 144 dMIPS. The dedicated application processor executes code from embedded memory (RAM) or external QSPI FLASH via a 16 kB 4-way associative cache controller, which is equipped with an on the fly decrypting capability without extra wait states. Bluetooth 5.2 connectivity is guaranteed by a new software-configurable Bluetooth low energy protocol engine (MAC) with an ultra-low-power radio transceiver, capable of +6 dBm output power and -97 dBm sensitivity offering a total link budget of 103 dB.

An optimized programmable sensor node controller allows sensor node operations and data acquisition without CPU intervention, achieving best-in-class power consumption.

The advanced power management unit of the DA1469x enables it to run on primary and secondary batteries, as well as provide power to external devices through the integrated SIMO DCDC and integrated LDOs. The on-chip JEITA-compliant hardware charger makes it possible to natively charge rechargeable batteries over USB.

A variety of standard and advanced peripherals enable interaction with other system components and the development of advanced user interfaces and feature-rich applications.

Key Features

- Compatible with *Bluetooth 5.2*, ETSI EN 300 328 and EN 300 440 Class 2 (Europe), FCC CFR47 Part 15 (US) and ARIB STD-T66 (Japan)
- Flexible processing power
 - 32 kHz up to 96 MHz 32-bit Arm Cortex-M33 with 16 kBytes, 4-way associative cache and FPU
 - A flexible and configurable BLE MAC engine implementing the controller stack up to HCI
 - A sensor node controller running uCode for sensors manipulation
 - A uCode based motor controller for manipulating analog gear boxes for hands on watches
 - Optimized power modes (Extended sleep, Deep sleep and Hibernation)
- Memories
 - 512 kB Data SRAM with retention capabilities
 - 4 kB One-Time-Programmable (OTP) memory
 - 16 kB Cache SRAM with retention capabilities
- 128 kB ROM (including boot ROM and PKI routines for Flash image authentication)
- Power management
 - SIMO Buck DC-DC converter (2.4 V-4.75 V)
 - Four power supply pins for external devices, alive while DA1469x is asleep
 - Supports Li-Polymer, Li-Ion, coin cells, NiMH and alkaline batteries
 - Hardware charger (up to 5.0 V) with programmable curves and JEITA support
 - Programmable thresholds for brownout detection
- Digital controlled oscillators
 - 32 MHz crystal (± 20 ppm max) and RC oscillator
 - 32 kHz crystal (± 50 ppm, ± 500 ppm max) and RCX oscillator
- Application cryptographic engine with AES-256, SHA-1, SHA-256, SHA-512 and a FIPS 140-2-compliant True Random Number Generator (TRNG)
- A Real Time Clock with 10 ms resolution
- 4 General purpose, 24-bit up/down timers with PWM

Multi-core Bluetooth 5.2 SoC family with system PMU

- Digital interfaces
 - Up to 55 General Purpose I/Os
 - Decrypt-on-the-fly QSPI FLASH interface
 - Separate QSPI PSRAM interface
 - Parallel/SPI LCD Controller with own DMA
 - 3 x UARTs up to 1 Mbps, one UART extended to support ISO7816
 - 2 SPI+™ controllers
 - 2 I2C controllers at 100 kHz, 400 kHz or 3.4 MHz
 - PDM interface with HW sample rate converter
 - I2S/PCM master/slave interface up to 8 channels
 - USB 1.1 Full Speed device interface
- Analog interfaces
 - 8-channel 10-bit SAR ADC, 3.4 Msamples/sec
 - 8-channel 14-bit ΣΔ ADC, 1000 samples/sec
 - Haptic driver interface to LRA/ERM motors
 - 2 matched White LED drivers
- Radio transceiver
 - Single wire antenna: no RF matching or RX/TX switching required
 - Supply current at VBAT2:
TX: 3 mA, RX: 1.8 mA (with ideal DC-DC)
 - Configurable transmit output power -18 to +6 dBm
 - -97 dBm receiver sensitivity
- Packages:
 - VFBGA86, 6x6, 0.55 mm pitch
 - VFBGA100, 5x5, 0.475 mm pitch

Applications

- Fitness trackers
- Sport watches
- Smartwatches
- Voice-controlled remote controls
- Rechargeable keyboards
- Toys
- Consumer appliances
- Home automation
- Industrial automation

Key Benefits

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

Contents

| | |
|--|-----------|
| General Description | 1 |
| Key Features | 1 |
| Applications | 3 |
| Key Benefits | 3 |
| Contents | 4 |
| Figures | 13 |
| Tables | 15 |
| 1 Block Diagram | 33 |
| 2 Application Information | 34 |
| 3 DA1469x Product Family | 36 |
| 4 Pinout | 37 |
| 4.1 VFBGA86 Pinout..... | 37 |
| 4.2 VFBGA100 Pinout..... | 49 |
| 5 Specifications | 63 |
| 5.1 Absolute Maximum Ratings..... | 64 |
| 5.2 Recommended Operating Conditions | 64 |
| 5.3 DC Characteristics | 65 |
| 5.4 Timing Characteristics | 67 |
| 5.5 Thermal Characteristics | 67 |
| 5.6 Reset Characteristics | 68 |
| 5.7 Bandgap Characteristics | 68 |
| 5.8 Brown-Out Detector Characteristics | 68 |
| 5.9 General Purpose ADC Characteristics | 69 |
| 5.10 ΣΔ ADC Characteristics | 70 |
| 5.11 DC-DC Converter Characteristics | 70 |
| 5.12 LDOs Characteristics | 72 |
| 5.13 32 kHz Crystal Oscillator Characteristics | 80 |
| 5.14 32 MHz Crystal Oscillator Characteristics | 80 |
| 5.15 RCX Oscillator Characteristics | 81 |
| 5.16 32MHz RC Oscillator Characteristics | 81 |
| 5.17 PLL Characteristics | 81 |
| 5.18 Charger Characteristics..... | 82 |
| 5.19 Battery Check Characteristics | 85 |
| 5.20 Digital I/O Characteristics..... | 85 |
| 5.21 QSPI Characteristics | 86 |
| 5.22 LED Characteristics | 87 |
| 5.23 LRA Characteristics | 88 |
| 5.24 USB Characteristics | 89 |
| 5.25 USB Charger Detection Characteristics | 89 |
| 5.26 Radio Characteristics | 90 |
| 6 System Overview | 98 |
| 6.1 Internal Blocks..... | 98 |
| 6.2 Digital Power Domains | 99 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | | |
|----------|---|------------|
| 6.3 | HW FSM (POWERUP, WAKEUP, GOTO Sleep)..... | 101 |
| 6.3.1 | HW FSM | 101 |
| 6.3.2 | POWERUP | 101 |
| 6.3.3 | WAKEUP Options | 102 |
| 6.3.3.1 | Slow WAKEUP..... | 103 |
| 6.3.3.2 | Fast WAKEUP..... | 103 |
| 6.3.3.3 | Ultra-Fast WAKEUP | 104 |
| 6.3.4 | Go-To-Sleep | 105 |
| 6.4 | Power Modes and Rails | 105 |
| 6.5 | OTP..... | 109 |
| 6.5.1 | OTP Segments | 109 |
| 6.5.2 | Configuration Script..... | 109 |
| 6.5.3 | Keys and Indexing..... | 110 |
| 6.5.4 | Customer Application Area | 111 |
| 6.6 | FLASH..... | 111 |
| 6.7 | Booting..... | 113 |
| 6.7.1 | Initialization | 115 |
| 6.7.2 | Configuration Script..... | 115 |
| 6.7.3 | Retrieve Application Code..... | 115 |
| 6.7.4 | Device Administration | 115 |
| 6.7.5 | Load Image | 116 |
| 6.8 | Memory Map | 116 |
| 6.9 | Resource Sharing | 118 |
| 6.10 | Remapping..... | 119 |
| 6.11 | Security Features | 119 |
| 6.11.1 | Secure Keys Manipulation | 120 |
| 6.11.2 | Secure Boot | 120 |
| 6.11.3 | Secure Access | 120 |
| 6.11.4 | Validation | 121 |
| 6.11.5 | Cryptography Operations | 121 |
| 7 | Power | 122 |
| 7.1 | Introduction | 122 |
| 7.2 | Architecture..... | 123 |
| 7.2.1 | SIMO DC-DC Converter..... | 123 |
| 7.2.2 | LDOs..... | 124 |
| 7.2.3 | Switching between DC-DC and LDOs..... | 124 |
| 7.2.4 | Low Power Clamps | 125 |
| 7.2.5 | Battery Check | 125 |
| 7.2.6 | Charger..... | 126 |
| 7.2.6.1 | JEITA | 127 |
| 7.2.6.2 | Errors and Flags..... | 129 |
| 7.2.6.3 | Timers | 130 |
| 7.2.7 | Charger Detection..... | 131 |
| 7.2.7.1 | Contact Detection..... | 132 |
| 7.2.7.2 | Primary Charger Detection..... | 132 |
| 7.2.7.3 | Secondary Charger Detection | 132 |
| 7.2.7.4 | Smartphone Charger Detection..... | 133 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | | |
|-----------|--|------------|
| 7.2.8 | Rails Discharge | 133 |
| 8 | Power Domain Controller (PDC) | 134 |
| 8.1 | Introduction | 134 |
| 8.2 | Architecture | 135 |
| 8.2.1 | Look-Up Table | 135 |
| 8.2.2 | Operation | 136 |
| 8.3 | Programming..... | 137 |
| 9 | Brown-Out Detector | 138 |
| 9.1 | Introduction | 138 |
| 9.2 | Architecture | 139 |
| 9.2.1 | Brown-Out Detector Monitor Levels | 139 |
| 9.2.2 | Brown-Out Detector Clock | 139 |
| 9.3 | Programming..... | 139 |
| 10 | Reset | 140 |
| 10.1 | Introduction | 140 |
| 10.2 | Architecture | 141 |
| 10.2.1 | Power-On Reset From Pin | 143 |
| 10.3 | Programming..... | 144 |
| 11 | Arm Cortex-M33..... | 146 |
| 11.1 | Introduction | 146 |
| 11.2 | Architecture | 146 |
| 11.2.1 | Interrupts | 146 |
| 11.2.2 | Reference | 148 |
| 12 | Cache Controller | 149 |
| 12.1 | Introduction | 149 |
| 12.2 | Architecture | 150 |
| 12.2.1 | Cacheable Range | 150 |
| 12.2.2 | Runtime Reconfiguration..... | 150 |
| 12.2.2.1 | Cache Line Reconfiguration | 150 |
| 12.2.2.2 | TAG Memory Word | 151 |
| 12.2.2.3 | Associativity Reconfiguration | 151 |
| 12.2.3 | Replacement Strategy..... | 151 |
| 12.2.4 | Cache Reset | 151 |
| 12.2.5 | Miss Rate Monitor | 152 |
| 12.2.6 | Miss Latency and Power | 152 |
| 12.3 | Programming..... | 153 |
| 12.3.1 | Cache Controller Programming..... | 153 |
| 12.3.2 | Miss Rate Monitor Programming..... | 153 |
| 13 | Bus | 154 |
| 13.1 | Introduction | 154 |
| 13.2 | Architecture | 155 |
| 14 | Configurable MAC (CMAC) | 157 |
| 14.1 | Introduction | 157 |
| 14.2 | Architecture | 158 |
| 14.2.1 | Dataflow | 158 |
| 14.2.2 | Diagnostics | 158 |

| | |
|--|------------|
| 15 Sensor Node Controller (SNC) | 159 |
| 15.1 Introduction | 159 |
| 15.2 Architecture | 159 |
| 15.2.1 Sensor Node Instruction Set | 160 |
| 15.2.2 SNC Clocking Considerations..... | 162 |
| 15.3 Programming..... | 162 |
| 16 Memory Controller | 163 |
| 16.1 Introduction | 163 |
| 16.2 Architecture..... | 164 |
| 16.3 Programming..... | 164 |
| 17 Clock Generation..... | 167 |
| 17.1 Clock Tree..... | 167 |
| 17.2 Crystal Oscillators | 169 |
| 17.2.1 Frequency Control (32 MHz Crystal)..... | 169 |
| 17.3 RC Oscillators | 170 |
| 17.3.1 Frequency Calibration | 170 |
| 17.4 PLL..... | 170 |
| 18 OTP Controller..... | 172 |
| 18.1 Introduction | 172 |
| 18.2 Architecture..... | 172 |
| 18.3 Programming..... | 174 |
| 19 Quad SPI FLASH Controller | 175 |
| 19.1 Introduction | 175 |
| 19.2 Architecture | 176 |
| 19.2.1 Interface | 176 |
| 19.2.2 Initialization FSM..... | 176 |
| 19.2.3 SPI Modes | 177 |
| 19.2.4 Access Modes..... | 178 |
| 19.2.5 Endianess | 178 |
| 19.2.6 Erase Suspend/Resume | 178 |
| 19.2.7 On-the-fly Decryption | 180 |
| 19.2.8 Timing | 180 |
| 19.3 Programming..... | 182 |
| 19.3.1 Auto Mode..... | 182 |
| 19.3.2 Manual Mode | 182 |
| 19.3.3 Clock Selection | 183 |
| 19.3.4 Receiving Data..... | 183 |
| 19.3.5 Delay Line Configuration..... | 183 |
| 20 Quad SPI RAM/FLASH Controller | 184 |
| 20.1 Introduction | 184 |
| 20.2 Architecture | 185 |
| 20.2.1 Interface | 185 |
| 20.2.2 SPI Modes | 185 |
| 20.2.3 Access Modes..... | 185 |
| 20.2.4 Endianness | 186 |
| 20.2.5 Erase Suspend/Resume | 186 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | | |
|-----------|---|------------|
| 20.2.6 | Low Power Considerations | 188 |
| 20.3 | Programming..... | 189 |
| 20.3.1 | Auto Mode..... | 189 |
| 20.3.2 | Manual Mode | 191 |
| 20.3.3 | Clock Selection | 191 |
| 20.3.4 | Received Data | 191 |
| 20.3.5 | Delay Line Configuration..... | 191 |
| 21 | Step Motor Controller | 192 |
| 21.1 | Introduction | 192 |
| 21.2 | Architecture | 193 |
| 21.2.1 | Commands and Waves..... | 193 |
| 21.2.2 | Pattern Generators (PGs) | 195 |
| 21.2.3 | Interrupt Generator..... | 196 |
| 21.3 | Programming..... | 196 |
| 22 | LCD Controller..... | 198 |
| 22.1 | Introduction | 198 |
| 22.2 | Architecture | 199 |
| 22.2.1 | Parallel LCD Interfaces | 199 |
| 22.2.1.1 | H SYNC / V SYNC Parallel Interface..... | 199 |
| 22.2.1.2 | JDI Parallel Interface | 199 |
| 22.2.2 | Serial LCD Interfaces | 200 |
| 22.2.2.1 | SPI3/4 Serial Interface | 200 |
| 22.2.2.2 | JDI SPI Serial Interface | 201 |
| 22.2.3 | Color Input Formats | 202 |
| 22.2.4 | Color Output Formats..... | 204 |
| 22.2.4.1 | SPI Output Formats..... | 204 |
| 22.2.4.2 | JDI SPI Output Formats | 204 |
| 22.3 | Programming..... | 205 |
| 23 | DMA Controller | 206 |
| 23.1 | Introduction | 206 |
| 23.2 | Architecture | 207 |
| 23.2.1 | DMA Peripherals..... | 207 |
| 23.2.2 | Input/Output Multiplexer | 207 |
| 23.2.3 | DMA Channel Operation | 207 |
| 23.2.4 | DMA Arbitration..... | 209 |
| 23.2.5 | Freezing DMA Channels | 209 |
| 23.2.6 | Secure DMA Channel | 209 |
| 23.3 | Programming..... | 209 |
| 23.3.1 | Memory to Memory Transfer..... | 209 |
| 23.3.2 | Peripheral to Memory Transfer | 210 |
| 24 | Crypto Engine | 211 |
| 24.1 | Introduction | 211 |
| 24.2 | Architecture | 211 |
| 24.2.1 | AES | 212 |
| 24.2.2 | Operation Modes | 212 |
| 24.2.3 | HASH | 212 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|------------|
| 24.3 Programming..... | 212 |
| 24.3.1 AES Engine Programming | 212 |
| 24.3.2 HASH Engine Programming | 213 |
| 25 True Random Number Generator..... | 214 |
| 25.1 Introduction | 214 |
| 25.2 Architecture | 214 |
| 25.3 Programming..... | 214 |
| 26 Wake-Up Controller..... | 216 |
| 26.1 Introduction | 216 |
| 26.2 Architecture | 216 |
| 26.3 Programming..... | 217 |
| 27 General Purpose ADC | 218 |
| 27.1 Introduction | 218 |
| 27.2 Architecture | 218 |
| 27.2.1 Input Channels and Input Scale | 219 |
| 27.2.2 Operation | 219 |
| 27.2.2.1 Enabling the ADC..... | 221 |
| 27.2.3 Operating Modes | 221 |
| 27.2.3.1 Manual Mode | 221 |
| 27.2.3.2 Continuous Mode | 221 |
| 27.2.4 Conversion Modes | 221 |
| 27.2.4.1 ADC Conversion..... | 221 |
| 27.2.4.2 Oversampling Mode | 223 |
| 27.2.4.3 Chopper Mode | 223 |
| 27.2.5 Additional Settings | 224 |
| 27.2.6 Non-Ideal Effects | 224 |
| 27.2.7 Offset Calibration | 224 |
| 27.2.8 Zero-Scale Adjustment..... | 225 |
| 27.2.9 Common Mode Adjustment..... | 225 |
| 27.2.10 Input Impedance, Inductance and Input Settling | 225 |
| 27.2.11 Reading Temperature Sensors | 226 |
| 27.3 Programming..... | 226 |
| 28 ΣΔ ADC | 228 |
| 28.1 Introduction | 228 |
| 28.2 Architecture | 228 |
| 28.3 Programming..... | 229 |
| 29 Audio Unit (AU)..... | 230 |
| 29.1 Introduction | 230 |
| 29.2 Architecture | 231 |
| 29.2.1 Data Paths | 231 |
| 29.2.2 Up/Down Sampler..... | 232 |
| 29.2.3 PCM Interface | 232 |
| 29.2.3.1 Channel Access and Delay | 232 |
| 29.2.3.2 Clock Generation..... | 232 |
| 29.2.3.3 External Synchronization..... | 235 |
| 29.2.3.4 Data Formats | 235 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | | |
|-----------|---|------------|
| 29.2.4 | PDM Interface | 238 |
| 29.2.5 | DMA Support | 239 |
| 29.2.6 | Interrupts..... | 239 |
| 29.3 | Programming..... | 239 |
| 29.3.1 | PDM Input to PCM Output | 239 |
| 30 | I2C Interface..... | 241 |
| 30.1 | Introduction | 241 |
| 30.2 | Architecture..... | 242 |
| 30.2.1 | I2C Behavior | 242 |
| 30.2.1.1 | START and STOP Generation..... | 243 |
| 30.2.1.2 | Combined Formats..... | 243 |
| 30.2.2 | I2C Protocols | 243 |
| 30.2.2.1 | START and STOP Conditions..... | 243 |
| 30.2.2.2 | Addressing Slave Protocol | 244 |
| 30.2.2.3 | Transmitting and Receiving Protocols..... | 245 |
| 30.2.3 | Multiple Master Arbitration | 246 |
| 30.2.4 | Clock Synchronization..... | 247 |
| 30.3 | Programming..... | 248 |
| 31 | UART | 249 |
| 31.1 | Introduction | 249 |
| 31.2 | Architecture..... | 250 |
| 31.2.1 | UART (RS232) Serial Protocol..... | 250 |
| 31.2.2 | Clock Support | 252 |
| 31.2.3 | Interrupts..... | 252 |
| 31.2.4 | Programmable THRE Interrupt | 253 |
| 31.2.5 | Shadow Registers..... | 255 |
| 31.2.6 | Direct Test Mode..... | 255 |
| 31.3 | Programming..... | 255 |
| 32 | Smart Card Interface | 257 |
| 32.1 | Introduction | 257 |
| 32.2 | Architecture..... | 257 |
| 32.2.1 | ISO7816-3 Clock Generation | 257 |
| 32.2.2 | ISO7816-3 Timer – Guard Time..... | 258 |
| 32.2.3 | ISO7816-3 Error Detection..... | 258 |
| 32.3 | Programming..... | 258 |
| 33 | SPI+ Interface | 260 |
| 33.1 | Introduction | 260 |
| 33.2 | Architecture..... | 261 |
| 33.2.1 | Master Mode | 261 |
| 33.2.2 | Slave Mode | 261 |
| 33.2.3 | SPI_POL and SPI_PHA..... | 261 |
| 33.2.4 | SPI_DO Idle Levels..... | 261 |
| 33.2.5 | Write Only Mode | 261 |
| 33.2.6 | Read Only Mode | 262 |
| 33.2.7 | Bidirectional Transfers with FIFO | 262 |
| 33.2.8 | TXreq Mode | 262 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | | |
|-----------|---|------------|
| 33.2.9 | DMA Operation Requirements | 262 |
| 33.2.10 | The 9-Bit Mode | 262 |
| 33.2.11 | Timing Diagrams | 262 |
| 33.3 | Programming..... | 264 |
| 34 | Real Time Clock..... | 265 |
| 34.1 | Introduction | 265 |
| 34.2 | Architecture | 265 |
| 34.3 | Programming..... | 266 |
| 35 | General Purpose Timers..... | 268 |
| 35.1 | Introduction | 268 |
| 35.2 | Architecture | 268 |
| 35.3 | Programming..... | 269 |
| 36 | Watchdog Timers | 271 |
| 36.1 | Introduction | 271 |
| 36.2 | Architecture | 272 |
| 36.3 | Programming..... | 272 |
| 36.3.1 | System Watchdog | 272 |
| 36.3.2 | CMAC Watchdog | 273 |
| 37 | USB Controller..... | 274 |
| 37.1 | Introduction | 274 |
| 37.2 | Architecture | 275 |
| 37.2.1 | Serial Interface Engine | 275 |
| 37.2.2 | Endpoint Pipe Controller (EPC) | 275 |
| 37.2.3 | Functional States | 276 |
| 37.2.3.1 | Line Condition Detection | 276 |
| 37.2.4 | Functional State Diagram..... | 277 |
| 37.2.5 | Address Detection..... | 279 |
| 37.2.6 | Transmit and Receive Endpoint FIFOs | 280 |
| 37.2.7 | Bidirectional Control Endpoint FIFO0..... | 281 |
| 37.2.8 | Transmit Endpoint FIFO..... | 282 |
| 37.2.9 | Receive Endpoint FIFO..... | 283 |
| 37.2.10 | Interrupt Hierarchy | 284 |
| 37.2.11 | USB Power Saving Modes | 285 |
| 37.2.11.1 | Freezing USB Node | 286 |
| 37.2.11.2 | Integrated resistors..... | 286 |
| 38 | Haptic Driver..... | 288 |
| 38.1 | Introduction | 288 |
| 38.2 | Architecture | 288 |
| 38.2.1 | Resonance Control | 290 |
| 38.3 | Programming..... | 291 |
| 38.4 | Legal | 292 |
| 39 | LEDs Driver..... | 293 |
| 39.1 | Introduction | 293 |
| 39.2 | Architecture | 293 |
| 39.3 | Programming..... | 294 |
| 40 | Input/Output Ports..... | 295 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | | |
|-----------|---|------------|
| 40.1 | Introduction | 295 |
| 40.2 | Architecture | 295 |
| 40.2.1 | Programmable Pin Assignment..... | 295 |
| 40.2.1.1 | Priority..... | 296 |
| 40.2.1.2 | Direction Control..... | 296 |
| 40.2.2 | General Purpose Port Registers | 296 |
| 40.2.2.1 | Port Data Register..... | 296 |
| 40.2.2.2 | Port Set Data Output Register..... | 296 |
| 40.2.2.3 | Port Reset Data Output Register..... | 296 |
| 40.2.3 | Fixed Assignment Functionality | 297 |
| 40.2.4 | GPIO State Retention While Sleeping..... | 300 |
| 40.2.5 | Special I/O Considerations..... | 301 |
| 41 | Radio | 302 |
| 41.1 | Introduction | 302 |
| 41.2 | Architecture | 302 |
| 41.2.1 | Receiver..... | 302 |
| 41.2.2 | Synthesizer | 303 |
| 41.2.3 | Transmitter..... | 303 |
| 41.2.4 | RFIO | 303 |
| 41.2.5 | Biasing | 303 |
| 41.2.6 | RF Monitoring | 303 |
| 42 | Registers | 304 |
| 42.1 | AMBA Bus Registers..... | 304 |
| 42.2 | LCD Controller Registers | 307 |
| 42.3 | LRA/ERM Registers | 318 |
| 42.4 | Memory Controller Registers..... | 326 |
| 42.5 | OTP Controller Registers | 333 |
| 42.6 | LED Controller Registers..... | 338 |
| 42.7 | QSPI Flash Registers..... | 339 |
| 42.8 | QSPI Ram Registers | 354 |
| 42.9 | RF Monitor Registers | 369 |
| 42.10 | Real Time Clock Registers..... | 371 |
| 42.11 | Motor Controller Registers | 377 |
| 42.12 | Sensor Node Controller Registers | 384 |
| 42.13 | SPI Controller Registers..... | 390 |
| 42.14 | Timers Registers | 394 |
| 42.15 | True Random Number Generator Controller Registers | 410 |
| 42.16 | UART Registers | 411 |
| 42.17 | USB Controller Registers | 492 |
| 42.18 | Silicon Version Registers | 527 |
| 42.19 | Wake-Up Controller Registers..... | 529 |
| 42.20 | Watchdog Controller Registers | 531 |
| 42.21 | General Purpose / $\Sigma\Delta$ ADC Registers..... | 532 |
| 42.22 | General Purpose I/O Registers | 538 |
| 42.23 | General Purpose Registers | 561 |
| 42.24 | I2C Controller Registers | 565 |
| 42.25 | Power Domain Controller Registers | 612 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|---|------------|
| 42.26 DCDC Converter Registers | 620 |
| 42.27 DMA Controller Registers..... | 627 |
| 42.28 Charger Registers | 654 |
| 42.29 Clock Generation Controller Registers | 686 |
| 42.30 Cache Controller Registers | 719 |
| 42.31 Audio Unit Registers | 724 |
| 42.32 Analog Miscellaneous Registers | 731 |
| 42.33 Crypto-Engine Registers | 732 |
| 43 Ordering Information | 738 |
| 44 Package information | 739 |
| 44.1 Moisture Sensitivity Level (MSL) | 739 |
| 44.2 Soldering information | 739 |
| 44.3 Package outlines..... | 739 |
| Revision History | 742 |

Figures

| | |
|--|-----|
| Figure 1: DA1469x Block Diagram | 33 |
| Figure 2: Activity Tracker..... | 34 |
| Figure 3: Analog Watch | 34 |
| Figure 4: Sport Watch..... | 35 |
| Figure 5: VFBGA86 Ball Assignment | 37 |
| Figure 6: VFBGA100 Ball Assignment..... | 49 |
| Figure 7: Digital Power Domains and Blocks Mapping | 100 |
| Figure 8: POWERUP, WAKEUP, GOTO Sleep HW FSM | 101 |
| Figure 9: POWERUP Timing Diagram..... | 102 |
| Figure 10: Slow Wake-Up Timing | 103 |
| Figure 11: Fast WAKEUP timing | 104 |
| Figure 12: Ultra Fast Wake-Up timing | 104 |
| Figure 13: FLASH Regions and Layout..... | 112 |
| Figure 14: BootROM Flowchart | 114 |
| Figure 15: Power Management Unit Architecture | 122 |
| Figure 16: SIMO DCDC Block Diagram..... | 123 |
| Figure 17: Battery Check Block Diagram..... | 125 |
| Figure 18: Charger HW Finite State Machine Diagram | 127 |
| Figure 19: NTC Battery Monitoring Connections | 128 |
| Figure 20: Battery Temperature Monitoring FSM | 129 |
| Figure 21: Charger Detection Circuit | 131 |
| Figure 22: Power rail discharging feature | 133 |
| Figure 23: Power Domain Controller Block Diagram | 134 |
| Figure 24: Brown-Out Detector Block Diagram | 138 |
| Figure 25: BOD FSM Timing Diagram in Sleep Mode | 139 |
| Figure 26: Reset Block Diagram..... | 140 |
| Figure 27: Power-on Reset timing diagram | 144 |
| Figure 28: Cache Controller Block Diagram | 150 |
| Figure 29: Bus Architecture | 155 |
| Figure 30: Configurable MAC Block Diagram..... | 157 |
| Figure 31: CMAC Diagnostics Timing Diagram | 158 |
| Figure 32: Sensor Node Controller Block Diagram..... | 159 |
| Figure 33: Internal Architecture of the Memory Controller..... | 163 |
| Figure 34: Example of Memory Segments Mapping..... | 166 |
| Figure 35: Clock Tree Diagram | 169 |
| Figure 36: Crystal Oscillator Circuits | 169 |
| Figure 37: XTAL32MHz Oscillator Frequency Trimming | 170 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Figure 38: OTP Controller Block Diagram | 172 |
| Figure 39: QSPI FLASH Controller Block Diagram | 175 |
| Figure 40: Erase Suspend/Resume in Auto Mode | 179 |
| Figure 41: QSPI Input Timing | 181 |
| Figure 42: QSPI Output Timing | 181 |
| Figure 43: Quad SPI RAM/FLASH Controller..... | 184 |
| Figure 44: Erase Suspend/Resume in Auto Mode | 188 |
| Figure 45: QSPI Split Burst Timing for Low Power (QSPI_FORENSEQ_EN=1)..... | 189 |
| Figure 46: QSPI Burst Timing for High Performance (QSPI_FORENSEQ_EN=0)..... | 189 |
| Figure 47: Motor Controller Block Diagram | 192 |
| Figure 48: Commands and Waveform Format | 193 |
| Figure 49: Loading Waves into Waves Memory Space | 195 |
| Figure 50: Waveform Generation in Signal Pairs | 195 |
| Figure 51: LCD Controller Block Diagram | 198 |
| Figure 52: HSYNC/VSYNC Typical Waveform..... | 199 |
| Figure 53: HSYNC/VSYNC Color Bits Waveform..... | 199 |
| Figure 54: DMA Controller Block Diagram..... | 206 |
| Figure 55: DMA channel diagram..... | 208 |
| Figure 56: AES/HASH Architecture | 211 |
| Figure 57: TRNG Block Diagram | 214 |
| Figure 58: Wake-Up Controller Block Diagram..... | 216 |
| Figure 59: General-Purpose ADC Block Diagram | 218 |
| Figure 60: GPADC Operation Flow Diagram..... | 220 |
| Figure 61: Block diagram of the $\Delta\Sigma$ ADC..... | 228 |
| Figure 62: Audio Unit Block Diagram | 231 |
| Figure 63: PCM Interface Formats | 236 |
| Figure 64: I2S Mode | 237 |
| Figure 65: TDM mode (left justified mode) | 237 |
| Figure 66: IOM format | 238 |
| Figure 67: PDM Mono/Stereo formats | 238 |
| Figure 68: SRC PDM Input Transfer Function..... | 238 |
| Figure 69: I2C Controller Block Diagram | 241 |
| Figure 70: Data Transfer on the I2C Bus..... | 242 |
| Figure 71: START and STOP Conditions | 243 |
| Figure 72: 7-bit Address Format..... | 244 |
| Figure 73: 10-bit Address Format..... | 244 |
| Figure 74: Master-Transmitter Protocol | 245 |
| Figure 75: Master-Receiver Protocol | 246 |
| Figure 76: START BYTE Transfer | 246 |
| Figure 77: Multiple Master Arbitration..... | 247 |
| Figure 78: Multiple Master Clock Synchronization..... | 248 |
| Figure 79: UART Block Diagram | 250 |
| Figure 80: Serial Data Format | 250 |
| Figure 81: Receiver Serial Data Sampling Points..... | 251 |
| Figure 82: Flowchart of Interrupt Generation for Programmable THRE Interrupt Mode | 254 |
| Figure 83: Flowchart of Interrupt Generation When Not in Programmable THRE Interrupt Mode | 255 |
| Figure 84: Smart Card (ISO7816-3) Block Diagram | 257 |
| Figure 85: SPI Block Diagram | 260 |
| Figure 86: SPI Master/Slave, Mode 0: SPI_POL=0 and SPI_PHA=0..... | 263 |
| Figure 87: SPI Master/Slave, Mode 1: SPI_POL=0 and SPI_PHA=1..... | 263 |
| Figure 88: SPI Master/Slave, Mode 2: SPI_POL=1 and SPI_PHA=0..... | 263 |
| Figure 89: SPI Master/slave, Mode 3: SPI_POL=1 and SPI_PHA=1 | 264 |
| Figure 90: Real Time Clock Block Diagram..... | 265 |
| Figure 91: General Purpose Timer's Block Diagram | 268 |
| Figure 92: System Watchdog Block Diagram | 271 |
| Figure 93: USB Node Block Diagram | 275 |
| Figure 94: Endpoint Operation | 276 |
| Figure 95: Node Functional State Diagram | 278 |
| Figure 96: USB Function Address/Endpoint Decoding | 280 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Figure 97: Endpoint 0 Operation | 282 |
| Figure 98: USB Tx FIFO Operation | 283 |
| Figure 99: USB Rx FIFO Operation..... | 284 |
| Figure 100: USB Interrupt Register Hierarchy | 285 |
| Figure 101: USB_Dp Resistor Switching | 286 |
| Figure 102: USB_Dm Resistor Switching | 287 |
| Figure 103: Block Diagram of the Haptic Driver | 288 |
| Figure 104: Circuit Diagram of the Haptic Driver and Equivalent Electrical Model of an LRA | 290 |
| Figure 105: LED Driver Block Diagram..... | 293 |
| Figure 106: Port P0 and P1 with Programmable Pin Assignment | 295 |
| Figure 107: Latching of Digital Pad Signals..... | 300 |
| Figure 108: Latching of QSPI Pad Signals | 300 |
| Figure 109: Radio Block Diagram..... | 302 |
| Figure 110: VFBGA86 Package Outline Drawing | 740 |
| Figure 111: VFBGA100 Package Outline Drawing | 741 |

Tables

| | |
|---|----|
| Table 1: DA1469x Product Family Differentiation | 36 |
| Table 2: DA14691/5 Pin Description | 38 |
| Table 3: DA14697/9 Pin Description | 50 |
| Table 4: Absolute Maximum Ratings | 64 |
| Table 5: Recommended Operating Conditions..... | 64 |
| Table 6: DC Characteristics..... | 65 |
| Table 7: Timing Characteristics | 67 |
| Table 8: Thermal Characteristics | 67 |
| Table 9: PAD_RESET - DC Characteristics | 68 |
| Table 10: BG_REF - DC Characteristics | 68 |
| Table 11: BOD - DC Characteristics..... | 68 |
| Table 12: GP_ADC - Recommended Operating Conditions | 69 |
| Table 13: GP_ADC - DC Characteristics..... | 69 |
| Table 14: GP_ADC - Timing Characteristics | 70 |
| Table 15: SD_ADC - DC Characteristics | 70 |
| Table 16: SD_ADC - Electrical performance | 70 |
| Table 17: DCDC - Recommended Operating Conditions | 71 |
| Table 18: DCDC - DC Characteristics | 71 |
| Table 19: LDO_1v8 - Recommended Operating Conditions | 73 |
| Table 20: LDO_1v8 - DC Characteristics | 73 |
| Table 21: LDO_1v8 - AC Characteristics..... | 73 |
| Table 22: LDO_1v8_RET - Recommended Operating Conditions | 73 |
| Table 23: LDO_1v8_RET - DC Characteristics | 74 |
| Table 24: LDO_1V8P - Recommended Operating Conditions | 74 |
| Table 25: LDO_1V8P - DC Characteristics | 74 |
| Table 26: LDO_1V8P - AC Characteristics..... | 75 |
| Table 27: LDO_1V8P_RET - Recommended Operating Conditions | 75 |
| Table 28: LDO_1V8P_RET - DC Characteristics | 75 |
| Table 29: LDO_CORE - Recommended Operating Conditions | 75 |
| Table 30: LDO_CORE - DC Characteristics | 75 |
| Table 31: LDO_CORE_RET - Recommended Operating Conditions | 76 |
| Table 32: LDO_CORE_RET - DC Characteristics | 76 |
| Table 33: LDO_RADIO - Recommended Operating Conditions | 77 |
| Table 34: LDO_RADIO - DC Characteristics | 77 |
| Table 35: LDO_RADIO - AC Characteristics | 78 |
| Table 36: LDO_USB - Recommended Operating Conditions | 78 |
| Table 37: LDO_USB - DC Characteristics | 78 |
| Table 38: LDO_VBAT - Recommended Operating Conditions | 78 |
| Table 39: LDO_VBAT - DC Characteristics | 79 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|---|-----|
| Table 40: LDO_VBAT_RET - Recommended Operating Conditions..... | 79 |
| Table 41: LDO_VBAT_RET - DC Characteristics..... | 79 |
| Table 42: OSC_XTAL32K - Recommended Operating Conditions | 80 |
| Table 43: OSC_XTAL32K - DC Characteristics | 80 |
| Table 44: OSC_XTAL32K - Timing Characteristics | 80 |
| Table 45: OSC_XTAL32MEG - Recommended Operating Conditions..... | 81 |
| Table 46: OSC_RCX - Timing Characteristics..... | 81 |
| Table 47: OSC_RC32MEG - Timing Characteristics | 81 |
| Table 48: PLL_SYS - DC Characteristics | 82 |
| Table 49: PLL_SYS - Timing Characteristics | 82 |
| Table 50: Charger - DC Characteristics..... | 82 |
| Table 51: BATCHECK - DC Characteristics | 85 |
| Table 52: GPIO - DC Characteristics | 85 |
| Table 53: GPIO_LOWDRV - DC Characteristics | 86 |
| Table 54: QSPIF pad - DC Characteristics..... | 87 |
| Table 55: QSPIR pad - DC Characteristics | 87 |
| Table 56: LED_DRIVER - DC Characteristics | 87 |
| Table 57: LED_DRIVER - AC Characteristics | 88 |
| Table 58: LRA - Recommended Operating Conditions..... | 88 |
| Table 59: LRA - DC Characteristics | 88 |
| Table 60: LRA - AC Characteristics..... | 89 |
| Table 61: GPIO_USB - DC Characteristics | 89 |
| Table 62: USB_CHRG_DET - DC Characteristics | 89 |
| Table 63: Radio BLE 1M - Recommended Operating Conditions | 90 |
| Table 64: Radio BLE 1M - DC Characteristics | 90 |
| Table 65: Radio BLE 1M - AC Characteristics..... | 91 |
| Table 66: Radio BLE 2M - Recommended Operating Conditions | 94 |
| Table 67: Radio BLE 2M - DC Characteristics | 94 |
| Table 68: Radio BLE 2M - AC Characteristics..... | 95 |
| Table 69: Power Domains Description | 100 |
| Table 70: Wake Up Modes | 102 |
| Table 71: Power Rails Configuration | 106 |
| Table 72: OTP Layout | 109 |
| Table 73: Configuration Script Commands and Description | 109 |
| Table 74: Configuration Script example..... | 110 |
| Table 75: Memory Map..... | 116 |
| Table 76: Busy Status Register | 119 |
| Table 77: Remapping Options..... | 119 |
| Table 78: Security Configuration Options | 120 |
| Table 79: Charger Error Flags..... | 130 |
| Table 80: USBP, USBN contact detection..... | 132 |
| Table 81: Charger Type Detection | 132 |
| Table 82: Secondary Charger Detection | 132 |
| Table 83: Smartphone Charger Characteristics | 133 |
| Table 84: PDC LUT format | 135 |
| Table 85: Peripheral Trigger Encoding | 135 |
| Table 86: Master Trigger Encoding | 136 |
| Table 87: Brown-out detectors and default levels..... | 139 |
| Table 88: Reset Signals and Registers | 141 |
| Table 89: Interrupt List..... | 146 |
| Table 90: Cache Line Size Reconfiguration Example | 151 |
| Table 91: TAG Memory Layout | 151 |
| Table 92: QSPI FLASH Cache Miss Latency | 152 |
| Table 93: AHB-DMA Master Priorities | 156 |
| Table 94: CMAC Diagnostic Signals | 158 |
| Table 95: Sensor Node Instruction Set Overview | 160 |
| Table 96: Opcode and Operand sizes | 161 |
| Table 97: Memory Controller Masters Access Metrics | 164 |
| Table 98: Memory Segments Description..... | 165 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 99: Initialization Command Encoding | 177 |
| Table 100: FLASH Initialization uCode Example..... | 177 |
| Table 101: QSPI Timing Parameters..... | 181 |
| Table 102: Commands Format..... | 193 |
| Table 103: Waves Format | 194 |
| Table 104: Hand Position and Restoration Error | 196 |
| Table 105: HSYNC/VSYNC Parallel I/F Pinout | 199 |
| Table 106: JDI Parallel I/F Pinout..... | 200 |
| Table 107: SPI3/4 Serial I/F Pinout | 201 |
| Table 108: JDI SPI Serial I/F Pinout..... | 201 |
| Table 109: L1 Grayscale/Palette | 202 |
| Table 110: L4 Grayscale/Palette | 202 |
| Table 111: L8 Grayscale/Palette | 202 |
| Table 112: 8-bit RGB332..... | 202 |
| Table 113: 16-bit RGB565..... | 202 |
| Table 114: 16-bit RGBX5551 | 202 |
| Table 115: 32-bit RGBX8888 | 202 |
| Table 116: 32-bit XRGB8888 | 203 |
| Table 117: 32-bit ABGR8888 | 203 |
| Table 118: 32-bit BGRA8888 | 203 |
| Table 119: SPI3/4 – 8-bit RGB-332 | 204 |
| Table 120: SPI3/4 – 8-bit RGB-444 | 204 |
| Table 121: SPI3/4 – 8-bit RGB-565 | 204 |
| Table 122: SPI3/4 – 8-bit RGB-666 | 204 |
| Table 123: SPI3/4 – 8-bit RGB-888 | 204 |
| Table 124: JDI SPI – 8-bit RGB-111-1 (3-bit mode)..... | 204 |
| Table 125: JDI SPI – 8-bit RGB-111-2 (4-bit mode)..... | 205 |
| Table 126: JDI SPI – 8-bit RGB-111-3 (3-bit mode)..... | 205 |
| Table 127: JDI SPI – 8-bit RGB-111-4 (1-bit mode)..... | 205 |
| Table 128: DMA served peripherals | 207 |
| Table 129: GPADC Input Channels and Voltage Range | 219 |
| Table 130: ADC_LDO_1V2 Start-Up Delay..... | 221 |
| Table 131: ADC Sampling Time Constant..... | 222 |
| Table 132: Total ADC Conversion Time | 222 |
| Table 133: Oversampling Mode Effective Number of Bits | 223 |
| Table 134: Preferred Settings for ENOB Measurements | 223 |
| Table 135: GPADC Calibration Procedure for Single-Ended and Differential Modes | 225 |
| Table 136: Common Mode Adjustment | 225 |
| Table 137: PCM_FDIV_REG Programming Example | 232 |
| Table 138: Fractional and Integer Only Clock Divisors for Various PCM Frequencies and Sample Rates | 233 |
| Table 139: I2C Definition of Bits in First Byte | 244 |
| Table 140: UART Baud Rate Generation | 251 |
| Table 141: UART2/3 Baud Rate Generation | 251 |
| Table 142: UART Interrupt Priorities | 252 |
| Table 143: Timer's Input GPIOs | 269 |
| Table 144: Functional States | 278 |
| Table 145: USB Node Endpoint Sizes..... | 280 |
| Table 146: Fixed Assignment of Specific Signals | 297 |
| Table 147: Register map DW | 304 |
| Table 148: AHB_DMA_PL1_REG (0x30020000) | 304 |
| Table 149: AHB_DMA_PL2_REG (0x30020004) | 305 |
| Table 150: AHB_DMA_PL3_REG (0x30020008) | 305 |
| Table 151: AHB_DMA_PL4_REG (0x3002000C) | 305 |
| Table 152: AHB_DMA_DFLT_MASTER_REG (0x30020048)..... | 305 |
| Table 153: AHB_DMA_WTEN_REG (0x3002004C) | 305 |
| Table 154: AHB_DMA_TCL_REG (0x30020050)..... | 305 |
| Table 155: AHB_DMA_CCLM1_REG (0x30020054) | 306 |
| Table 156: AHB_DMA_CCLM2_REG (0x30020058) | 306 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 157: AHB_DMA_CCLM3_REG (0x3002005C) | 306 |
| Table 158: AHB_DMA_CCLM4_REG (0x30020060) | 306 |
| Table 159: AHB_DMA_VERSION_REG (0x30020090) | 306 |
| Table 160: Register map LCDC | 307 |
| Table 161: LCDC_MODE_REG (0x30030000) | 308 |
| Table 162: LCDC_CLKCTRL_REG (0x30030004) | 309 |
| Table 163: LCDC_BGCOLOR_REG (0x30030008) | 310 |
| Table 164: LCDC_RESXY_REG (0x3003000C) | 310 |
| Table 165: LCDC_FRONTPORCHXY_REG (0x30030014) | 310 |
| Table 166: LCDC_BLANKINGXY_REG (0x30030018) | 310 |
| Table 167: LCDC_BACKPORCHXY_REG (0x3003001C) | 310 |
| Table 168: LCDC_DBIB_CFG_REG (0x30030028) | 311 |
| Table 169: LCDC_GPIO_REG (0x3003002C) | 312 |
| Table 170: LCDC_LAYER0_MODE_REG (0x30030030) | 312 |
| Table 171: LCDC_LAYER0_STARTXY_REG (0x30030034) | 313 |
| Table 172: LCDC_LAYER0_SIZEXY_REG (0x30030038) | 313 |
| Table 173: LCDC_LAYER0_BASEADDR_REG (0x3003003C) | 313 |
| Table 174: LCDC_LAYER0_STRIDE_REG (0x30030040) | 313 |
| Table 175: LCDC_LAYER0_RESXY_REG (0x30030044) | 314 |
| Table 176: LCDC_JDI_RESXY_REG (0x30030090) | 314 |
| Table 177: LCDC_JDI_FBX_BLANKING_REG (0x30030094) | 314 |
| Table 178: LCDC_JDI_FBY_BLANKING_REG (0x30030098) | 314 |
| Table 179: LCDC_JDI_HCK_WIDTH_REG (0x3003009C) | 314 |
| Table 180: LCDC_JDI_XRST_WIDTH_REG (0x300300A0) | 314 |
| Table 181: LCDC_JDI_VST_DELAY_REG (0x300300A4) | 315 |
| Table 182: LCDC_JDI_VST_WIDTH_REG (0x300300A8) | 315 |
| Table 183: LCDC_JDI_VCK_DELAY_REG (0x300300AC) | 315 |
| Table 184: LCDC_JDI_HST_DELAY_REG (0x300300B0) | 315 |
| Table 185: LCDC_JDI_HST_WIDTH_REG (0x300300B4) | 315 |
| Table 186: LCDC_JDI_ENB_START_HLINE_REG (0x300300B8) | 315 |
| Table 187: LCDC_JDI_ENB_END_HLINE_REG (0x300300BC) | 315 |
| Table 188: LCDC_JDI_ENB_START_CLK_REG (0x300300C0) | 316 |
| Table 189: LCDC_JDI_ENB_WIDTH_CLK_REG (0x300300C4) | 316 |
| Table 190: LCDC_DBIB_CMD_REG (0x300300E8) | 316 |
| Table 191: LCDC_IDREG_REG (0x300300F4) | 316 |
| Table 192: LCDC_INTERRUPT_REG (0x300300F8) | 316 |
| Table 193: LCDC_STATUS_REG (0x300300FC) | 317 |
| Table 194: LCDC_CRC_REG (0x30030184) | 318 |
| Table 195: LCDC_LAYER0_OFFSETX_REG (0x30030188) | 318 |
| Table 196: Register map LRA | 318 |
| Table 197: LRA_CTRL1_REG (0x50030A00) | 319 |
| Table 198: LRA_CTRL2_REG (0x50030A04) | 320 |
| Table 199: LRA_CTRL3_REG (0x50030A08) | 320 |
| Table 200: LRA_FLT_SMP1_REG (0x50030A0C) | 321 |
| Table 201: LRA_FLT_SMP2_REG (0x50030A10) | 321 |
| Table 202: LRA_FLT_SMP3_REG (0x50030A14) | 321 |
| Table 203: LRA_FLT_SMP4_REG (0x50030A18) | 321 |
| Table 204: LRA_FLT_SMP5_REG (0x50030A1C) | 321 |
| Table 205: LRA_FLT_SMP6_REG (0x50030A20) | 322 |
| Table 206: LRA_FLT_SMP7_REG (0x50030A24) | 322 |
| Table 207: LRA_FLT_SMP8_REG (0x50030A28) | 322 |
| Table 208: LRA_FLT_COEF1_REG (0x50030A2C) | 322 |
| Table 209: LRA_FLT_COEF2_REG (0x50030A30) | 322 |
| Table 210: LRA_FLT_COEF3_REG (0x50030A34) | 323 |
| Table 211: LRA_BRD_LS_REG (0x50030A38) | 323 |
| Table 212: LRA_BRD_HS_REG (0x50030A3C) | 323 |
| Table 213: LRA_BRD_STAT_REG (0x50030A40) | 324 |
| Table 214: LRA_ADC_CTRL1_REG (0x50030A44) | 324 |
| Table 215: LRA_ADC_RESULT_REG (0x50030A50) | 325 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 216: LRA_LDO_REG (0x50030A54) | 325 |
| Table 217: LRA_DFT_REG (0x50030A58) | 325 |
| Table 218: Register map MEMCTRL | 326 |
| Table 219: MEM_PRIO_REG (0x50050004) | 327 |
| Table 220: MEM_STALL_REG (0x50050008) | 327 |
| Table 221: MEM_STATUS_REG (0x5005000C) | 328 |
| Table 222: MEM_STATUS2_REG (0x50050010) | 328 |
| Table 223: CMI_CODE_BASE_REG (0x50050020) | 329 |
| Table 224: CMI_DATA_BASE_REG (0x50050024) | 329 |
| Table 225: CMI_SHARED_BASE_REG (0x50050028) | 329 |
| Table 226: CMI_END_REG (0x5005002C) | 329 |
| Table 227: SNC_BASE_REG (0x50050030) | 330 |
| Table 228: BUSY_SET_REG (0x50050074) | 330 |
| Table 229: BUSY_RESET_REG (0x50050078) | 331 |
| Table 230: BUSY_STAT_REG (0x5005007C) | 333 |
| Table 231: Register map OTPC | 334 |
| Table 232: OTPC_MODE_REG (0x30070000) | 334 |
| Table 233: OTPC_STAT_REG (0x30070004) | 335 |
| Table 234: OTPC_PADDR_REG (0x30070008) | 336 |
| Table 235: OTPC_PWORD_REG (0x3007000C) | 336 |
| Table 236: OTPC_TIM1_REG (0x30070010) | 337 |
| Table 237: OTPC_TIM2_REG (0x30070014) | 337 |
| Table 238: Register map PWM for LEDs | 338 |
| Table 239: PWMLED_DUTY_CYCLE_LED1_REG (0x50030500) | 338 |
| Table 240: PWMLED_DUTY_CYCLE_LED2_REG (0x50030504) | 339 |
| Table 241: PWMLED_FREQUENCY_REG (0x50030508) | 339 |
| Table 242: PWMLED_CTRL_REG (0x5003050C) | 339 |
| Table 243: Register map QSPIC | 339 |
| Table 244: QSPIC_CTRLBUS_REG (0x38000000) | 341 |
| Table 245: QSPIC_CTRLMODE_REG (0x38000004) | 341 |
| Table 246: QSPIC_RECVDATA_REG (0x38000008) | 343 |
| Table 247: QSPIC_BURSTCMDA_REG (0x3800000C) | 343 |
| Table 248: QSPIC_BURSTCMDB_REG (0x38000010) | 344 |
| Table 249: QSPIC_STATUS_REG (0x38000014) | 345 |
| Table 250: QSPIC_WRITEDATA_REG (0x38000018) | 346 |
| Table 251: QSPIC_READDATA_REG (0x3800001C) | 346 |
| Table 252: QSPIC_DUMMYDATA_REG (0x38000020) | 346 |
| Table 253: QSPIC_ERASECTRL_REG (0x38000024) | 346 |
| Table 254: QSPIC_ERASECMDA_REG (0x38000028) | 347 |
| Table 255: QSPIC_ERASECMDB_REG (0x3800002C) | 347 |
| Table 256: QSPIC_BURSTBRK_REG (0x38000030) | 348 |
| Table 257: QSPIC_STATUSCMD_REG (0x38000034) | 349 |
| Table 258: QSPIC_CHCKERASE_REG (0x38000038) | 350 |
| Table 259: QSPIC_GP_REG (0x3800003C) | 350 |
| Table 260: QSPIC_UCODE_START (0x38000040) | 351 |
| Table 261: QSPIC_CTR_CTRL_REG (0x38000080) | 351 |
| Table 262: QSPIC_CTR_SADDR_REG (0x38000084) | 351 |
| Table 263: QSPIC_CTR_EADDR_REG (0x38000088) | 352 |
| Table 264: QSPIC_CTR_NONCE_0_3_REG (0x3800008C) | 352 |
| Table 265: QSPIC_CTR_NONCE_4_7_REG (0x38000090) | 353 |
| Table 266: QSPIC_CTR_KEY_0_3_REG (0x38000094) | 353 |
| Table 267: QSPIC_CTR_KEY_4_7_REG (0x38000098) | 353 |
| Table 268: QSPIC_CTR_KEY_8_11_REG (0x3800009C) | 354 |
| Table 269: QSPIC_CTR_KEY_12_15_REG (0x380000A0) | 354 |
| Table 270: QSPIC_CTR_KEY_16_19_REG (0x380000A4) | 354 |
| Table 271: QSPIC_CTR_KEY_20_23_REG (0x380000A8) | 354 |
| Table 272: QSPIC_CTR_KEY_24_27_REG (0x380000AC) | 354 |
| Table 273: QSPIC_CTR_KEY_28_31_REG (0x380000B0) | 354 |
| Table 274: Register map QSPIC2 | 354 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 275: QSPIC2_CTRLBUS_REG (0x34000000)..... | 355 |
| Table 276: QSPIC2_CTRLMODE_REG (0x34000004)..... | 356 |
| Table 277: QSPIC2_RECVDATA_REG (0x34000008) | 358 |
| Table 278: QSPIC2_BURSTCMDA_REG (0x3400000C) | 358 |
| Table 279: QSPIC2_BURSTCMDB_REG (0x34000010) | 359 |
| Table 280: QSPIC2_STATUS_REG (0x34000014) | 361 |
| Table 281: QSPIC2_WRITEDATA_REG (0x34000018)..... | 361 |
| Table 282: QSPIC2_READDATA_REG (0x3400001C) | 361 |
| Table 283: QSPIC2_DUMMYDATA_REG (0x34000020) | 362 |
| Table 284: QSPIC2_ERASECTRL_REG (0x34000024) | 362 |
| Table 285: QSPIC2_ERASECMDA_REG (0x34000028) | 363 |
| Table 286: QSPIC2_ERASECMDB_REG (0x3400002C) | 363 |
| Table 287: QSPIC2_BURSTBRK_REG (0x34000030) | 364 |
| Table 288: QSPIC2_STATUSCMD_REG (0x34000034) | 365 |
| Table 289: QSPIC2_CHCKERASE_REG (0x34000038) | 366 |
| Table 290: QSPIC2_GP_REG (0x3400003C) | 366 |
| Table 291: QSPIC2_AWRITECMD_REG (0x34000040) | 366 |
| Table 292: QSPIC2_MEMBLEN_REG (0x34000044) | 367 |
| Table 293: Register map RFPT | 369 |
| Table 294: RFMON_CTRL_REG (0x50040600) | 369 |
| Table 295: RFMON_ADDR_REG (0x50040604) | 370 |
| Table 296: RFMON_LEN_REG (0x50040608)..... | 370 |
| Table 297: RFMON_STAT_REG (0x5004060C) | 370 |
| Table 298: RFMON_CRV_ADDR_REG (0x50040610) | 370 |
| Table 299: RFMON_CRV_LEN_REG (0x50040614) | 370 |
| Table 300: Register map RTC | 371 |
| Table 301: RTC_CONTROL_REG (0x50000400) | 372 |
| Table 302: RTC_HOUR_MODE_REG (0x50000404) | 372 |
| Table 303: RTC_TIME_REG (0x50000408) | 372 |
| Table 304: RTC_CALENDAR_REG (0x5000040C) | 372 |
| Table 305: RTC_TIME_ALARM_REG (0x50000410) | 373 |
| Table 306: RTC_CALENDAR_ALARM_REG (0x50000414) | 373 |
| Table 307: RTC_ALARM_ENABLE_REG (0x50000418) | 373 |
| Table 308: RTC_EVENT_FLAGS_REG (0x5000041C) | 374 |
| Table 309: RTC_INTERRUPT_ENABLE_REG (0x50000420) | 374 |
| Table 310: RTC_INTERRUPT_DISABLE_REG (0x50000424) | 375 |
| Table 311: RTC_INTERRUPT_MASK_REG (0x50000428) | 375 |
| Table 312: RTC_STATUS_REG (0x5000042C) | 376 |
| Table 313: RTC_KEEP_RTC_REG (0x50000430) | 376 |
| Table 314: RTC_EVENT_CTRL_REG (0x50000480) | 376 |
| Table 315: RTC_MOTOR_EVENT_PERIOD_REG (0x50000484) | 376 |
| Table 316: RTC_PDC_EVENT_PERIOD_REG (0x50000488) | 377 |
| Table 317: RTC_PDC_EVENT_CLEAR_REG (0x5000048C) | 377 |
| Table 318: RTC_MOTOR_EVENT_CNT_REG (0x50000490) | 377 |
| Table 319: RTC_PDC_EVENT_CNT_REG (0x50000494) | 377 |
| Table 320: Register map SMOTOR | 377 |
| Table 321: SMOTOR_CTRL_REG (0x50030E00) | 378 |
| Table 322: PG0_CTRL_REG (0x50030E04) | 379 |
| Table 323: PG1_CTRL_REG (0x50030E08) | 379 |
| Table 324: PG2_CTRL_REG (0x50030E0C) | 380 |
| Table 325: PG3_CTRL_REG (0x50030E10) | 381 |
| Table 326: PG4_CTRL_REG (0x50030E14) | 382 |
| Table 327: SMOTOR_TRIGGER_REG (0x50030E18) | 382 |
| Table 328: SMOTOR_CMD_FIFO_REG (0x50030E20) | 383 |
| Table 329: SMOTOR_CMD_READ_PTR_REG (0x50030E24) | 383 |
| Table 330: SMOTOR_CMD_WRITE_PTR_REG (0x50030E28) | 383 |
| Table 331: SMOTOR_STATUS_REG (0x50030E2C) | 383 |
| Table 332: SMOTOR_IRQ_CLEAR_REG (0x50030E30) | 384 |
| Table 333: WAVETABLE_BASE (0x50030E40) | 384 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 334: CMD_TABLE_BASE (0x50030E80) | 384 |
| Table 335: Register map SNC..... | 385 |
| Table 336: SNC_CTRL_REG (0x50020C00) | 385 |
| Table 337: SNC_STATUS_REG (0x50020C04) | 386 |
| Table 338: SNC_LP_TIMER_REG (0x50020C08) | 389 |
| Table 339: SNC_PC_REG (0x50020C0C) | 389 |
| Table 340: SNC_R1_REG (0x50020C10) | 389 |
| Table 341: SNC_R2_REG (0x50020C14) | 389 |
| Table 342: SNC_TMP1_REG (0x50020C18) | 389 |
| Table 343: SNC_TMP2_REG (0x50020C1C) | 390 |
| Table 344: Register map SPI | 390 |
| Table 345: SPI_CTRL_REG (0x50020300)..... | 390 |
| Table 346: SPI_RX_TX_REG (0x50020304) | 392 |
| Table 347: SPI_CLEAR_INT_REG (0x50020308) | 392 |
| Table 348: SPI2_CTRL_REG (0x50020400)..... | 392 |
| Table 349: SPI2_RX_TX_REG (0x50020404) | 394 |
| Table 350: SPI2_CLEAR_INT_REG (0x50020408) | 394 |
| Table 351: Register map Timer1 | 394 |
| Table 352: TIMER_CTRL_REG (0x50010200) | 397 |
| Table 353: TIMER_TIMER_VAL_REG (0x50010204)..... | 398 |
| Table 354: TIMER_STATUS_REG (0x50010208) | 398 |
| Table 355: TIMER_GPIO1_CONF_REG (0x5001020C) | 399 |
| Table 356: TIMER_GPIO2_CONF_REG (0x50010210)..... | 399 |
| Table 357: TIMER_RELOAD_REG (0x50010214) | 399 |
| Table 358: TIMER_SHOTWIDTH_REG (0x50010218) | 399 |
| Table 359: TIMER_PRESCALER_REG (0x5001021C) | 399 |
| Table 360: TIMER_CAPTURE_GPIO1_REG (0x50010220)..... | 400 |
| Table 361: TIMER_CAPTURE_GPIO2_REG (0x50010224)..... | 400 |
| Table 362: TIMER_PRESCALER_VAL_REG (0x50010228) | 400 |
| Table 363: TIMER_PWM_FREQ_REG (0x5001022C) | 400 |
| Table 364: TIMER_PWM_DC_REG (0x50010230) | 400 |
| Table 365: TIMER_GPIO3_CONF_REG (0x50010234) | 400 |
| Table 366: TIMER_GPIO4_CONF_REG (0x50010238)..... | 401 |
| Table 367: TIMER_CAPTURE_GPIO3_REG (0x5001023C) | 401 |
| Table 368: TIMER_CAPTURE_GPIO4_REG (0x50010240)..... | 401 |
| Table 369: TIMER_CLEAR_GPIO_EVENT_REG (0x50010244) | 401 |
| Table 370: TIMER_CLEAR_IRQ_REG (0x50010248) | 401 |
| Table 371: TIMER2_CTRL_REG (0x50010300) | 401 |
| Table 372: TIMER2_TIMER_VAL_REG (0x50010304)..... | 402 |
| Table 373: TIMER2_STATUS_REG (0x50010308) | 402 |
| Table 374: TIMER2_GPIO1_CONF_REG (0x5001030C) | 403 |
| Table 375: TIMER2_GPIO2_CONF_REG (0x50010310)..... | 403 |
| Table 376: TIMER2_RELOAD_REG (0x50010314) | 403 |
| Table 377: TIMER2_SHOTWIDTH_REG (0x50010318) | 403 |
| Table 378: TIMER2_PRESCALER_REG (0x5001031C) | 403 |
| Table 379: TIMER2_CAPTURE_GPIO1_REG (0x50010320)..... | 403 |
| Table 380: TIMER2_CAPTURE_GPIO2_REG (0x50010324) | 404 |
| Table 381: TIMER2_PRESCALER_VAL_REG (0x50010328) | 404 |
| Table 382: TIMER2_PWM_FREQ_REG (0x5001032C) | 404 |
| Table 383: TIMER2_PWM_DC_REG (0x50010330)..... | 404 |
| Table 384: TIMER2_CLEAR_IRQ_REG (0x50010334) | 404 |
| Table 385: TIMER3_CTRL_REG (0x50040A00) | 404 |
| Table 386: TIMER3_TIMER_VAL_REG (0x50040A04) | 405 |
| Table 387: TIMER3_STATUS_REG (0x50040A08) | 405 |
| Table 388: TIMER3_GPIO1_CONF_REG (0x50040A0C) | 405 |
| Table 389: TIMER3_GPIO2_CONF_REG (0x50040A10) | 406 |
| Table 390: TIMER3_RELOAD_REG (0x50040A14) | 406 |
| Table 391: TIMER3_PRESCALER_REG (0x50040A1C) | 406 |
| Table 392: TIMER3_CAPTURE_GPIO1_REG (0x50040A20) | 406 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 393: TIMER3_CAPTURE_GPIO2_REG (0x50040A24) | 406 |
| Table 394: TIMER3_PRESCALER_VAL_REG (0x50040A28) | 406 |
| Table 395: TIMER3_PWM_FREQ_REG (0x50040A2C) | 407 |
| Table 396: TIMER3_PWM_DC_REG (0x50040A30) | 407 |
| Table 397: TIMER3_CLEAR_IRQ_REG (0x50040A34) | 407 |
| Table 398: TIMER4_CTRL_REG (0x50040B00) | 407 |
| Table 399: TIMER4_TIMER_VAL_REG (0x50040B04) | 408 |
| Table 400: TIMER4_STATUS_REG (0x50040B08) | 408 |
| Table 401: TIMER4_GPIO1_CONF_REG (0x50040B0C) | 408 |
| Table 402: TIMER4_GPIO2_CONF_REG (0x50040B10) | 408 |
| Table 403: TIMER4_RELOAD_REG (0x50040B14) | 408 |
| Table 404: TIMER4_PRESCALER_REG (0x50040B1C) | 409 |
| Table 405: TIMER4_CAPTURE_GPIO1_REG (0x50040B20) | 409 |
| Table 406: TIMER4_CAPTURE_GPIO2_REG (0x50040B24) | 409 |
| Table 407: TIMER4_PRESCALER_VAL_REG (0x50040B28) | 409 |
| Table 408: TIMER4_PWM_FREQ_REG (0x50040B2C) | 409 |
| Table 409: TIMER4_PWM_DC_REG (0x50040B30) | 409 |
| Table 410: TIMER4_CLEAR_IRQ_REG (0x50040B34) | 410 |
| Table 411: Register map TRNG | 410 |
| Table 412: TRNG_CTRL_REG (0x50040C00) | 410 |
| Table 413: TRNG_FIFOLVL_REG (0x50040C04) | 410 |
| Table 414: TRNG_VER_REG (0x50040C08) | 410 |
| Table 415: Register map UART | 411 |
| Table 416: UART_RBR_THR_DLL_REG (0x50020000) | 415 |
| Table 417: UART_IER_DLH_REG (0x50020004) | 416 |
| Table 418: UART_IIR_FCR_REG (0x50020008) | 417 |
| Table 419: UART_LCR_REG (0x5002000C) | 418 |
| Table 420: UART_MCR_REG (0x50020010) | 419 |
| Table 421: UART_LSR_REG (0x50020014) | 420 |
| Table 422: UART_SCR_REG (0x5002001C) | 422 |
| Table 423: UART_SRBR_STHR0_REG (0x50020030) | 422 |
| Table 424: UART_SRBR_STHR1_REG (0x50020034) | 422 |
| Table 425: UART_SRBR_STHR2_REG (0x50020038) | 423 |
| Table 426: UART_SRBR_STHR3_REG (0x5002003C) | 424 |
| Table 427: UART_SRBR_STHR4_REG (0x50020040) | 425 |
| Table 428: UART_SRBR_STHR5_REG (0x50020044) | 425 |
| Table 429: UART_SRBR_STHR6_REG (0x50020048) | 426 |
| Table 430: UART_SRBR_STHR7_REG (0x5002004C) | 427 |
| Table 431: UART_SRBR_STHR8_REG (0x50020050) | 428 |
| Table 432: UART_SRBR_STHR9_REG (0x50020054) | 428 |
| Table 433: UART_SRBR_STHR10_REG (0x50020058) | 429 |
| Table 434: UART_SRBR_STHR11_REG (0x5002005C) | 430 |
| Table 435: UART_SRBR_STHR12_REG (0x50020060) | 430 |
| Table 436: UART_SRBR_STHR13_REG (0x50020064) | 431 |
| Table 437: UART_SRBR_STHR14_REG (0x50020068) | 432 |
| Table 438: UART_SRBR_STHR15_REG (0x5002006C) | 433 |
| Table 439: UART_USR_REG (0x5002007C) | 433 |
| Table 440: UART_TFL_REG (0x50020080) | 434 |
| Table 441: UART_RFL_REG (0x50020084) | 435 |
| Table 442: UART_SRR_REG (0x50020088) | 435 |
| Table 443: UART_SBCR_REG (0x50020090) | 435 |
| Table 444: UART_SDMAM_REG (0x50020094) | 436 |
| Table 445: UART_SFE_REG (0x50020098) | 436 |
| Table 446: UART_SRT_REG (0x5002009C) | 436 |
| Table 447: UART_STET_REG (0x500200A0) | 437 |
| Table 448: UART_HTX_REG (0x500200A4) | 437 |
| Table 449: UART_DMASA_REG (0x500200A8) | 437 |
| Table 450: UART_DLF_REG (0x500200C0) | 437 |
| Table 451: UART_UCV_REG (0x500200F8) | 438 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|------|
| Table 452: UART_CTR_REG (0x500200FC)..... | .438 |
| Table 453: UART2_RBR_THR_DLL_REG (0x50020100)..... | .438 |
| Table 454: UART2_IER_DLH_REG (0x50020104)..... | .439 |
| Table 455: UART2_IIR_FCR_REG (0x50020108)..... | .440 |
| Table 456: UART2_LCR_REG (0x5002010C)..... | .441 |
| Table 457: UART2_MCR_REG (0x50020110)..... | .442 |
| Table 458: UART2_LSR_REG (0x50020114)..... | .443 |
| Table 459: UART2_MSR_REG (0x50020118)..... | .445 |
| Table 460: UART2_SCR_REG (0x5002011C)..... | .446 |
| Table 461: UART2_SRBR_STHR0_REG (0x50020130)..... | .446 |
| Table 462: UART2_SRBR_STHR1_REG (0x50020134)..... | .447 |
| Table 463: UART2_SRBR_STHR2_REG (0x50020138)..... | .447 |
| Table 464: UART2_SRBR_STHR3_REG (0x5002013C)..... | .448 |
| Table 465: UART2_SRBR_STHR4_REG (0x50020140)..... | .449 |
| Table 466: UART2_SRBR_STHR5_REG (0x50020144)..... | .450 |
| Table 467: UART2_SRBR_STHR6_REG (0x50020148)..... | .450 |
| Table 468: UART2_SRBR_STHR7_REG (0x5002014C)..... | .451 |
| Table 469: UART2_SRBR_STHR8_REG (0x50020150)..... | .452 |
| Table 470: UART2_SRBR_STHR9_REG (0x50020154)..... | .452 |
| Table 471: UART2_SRBR_STHR10_REG (0x50020158)..... | .453 |
| Table 472: UART2_SRBR_STHR11_REG (0x5002015C)..... | .454 |
| Table 473: UART2_SRBR_STHR12_REG (0x50020160)..... | .455 |
| Table 474: UART2_SRBR_STHR13_REG (0x50020164)..... | .455 |
| Table 475: UART2_SRBR_STHR14_REG (0x50020168)..... | .456 |
| Table 476: UART2_SRBR_STHR15_REG (0x5002016C)..... | .457 |
| Table 477: UART2_USR_REG (0x5002017C)..... | .458 |
| Table 478: UART2_TFL_REG (0x50020180)..... | .458 |
| Table 479: UART2_RFL_REG (0x50020184)..... | .459 |
| Table 480: UART2_SRR_REG (0x50020188)..... | .459 |
| Table 481: UART2_SRTS_REG (0x5002018C)..... | .459 |
| Table 482: UART2_SBCR_REG (0x50020190)..... | .460 |
| Table 483: UART2_SDMA_M_REG (0x50020194)..... | .460 |
| Table 484: UART2_SFE_REG (0x50020198)..... | .460 |
| Table 485: UART2_SRT_REG (0x5002019C)..... | .461 |
| Table 486: UART2_STET_REG (0x500201A0)..... | .461 |
| Table 487: UART2_HTX_REG (0x500201A4)..... | .461 |
| Table 488: UART2_DMASA_REG (0x500201A8)..... | .462 |
| Table 489: UART2_DLF_REG (0x500201C0)..... | .462 |
| Table 490: UART2_RAR_REG (0x500201C4)..... | .462 |
| Table 491: UART2_TAR_REG (0x500201C8)..... | .462 |
| Table 492: UART2_LCR_EXT (0x500201CC)..... | .463 |
| Table 493: UART2_UCV_REG (0x500201F8)..... | .464 |
| Table 494: UART2_CTR_REG (0x500201FC)..... | .464 |
| Table 495: UART3_RBR_THR_DLL_REG (0x50020200)..... | .464 |
| Table 496: UART3_IER_DLH_REG (0x50020204)..... | .465 |
| Table 497: UART3_IIR_FCR_REG (0x50020208)..... | .466 |
| Table 498: UART3_LCR_REG (0x5002020C)..... | .467 |
| Table 499: UART3_MCR_REG (0x50020210)..... | .468 |
| Table 500: UART3_LSR_REG (0x50020214)..... | .469 |
| Table 501: UART3_MSR_REG (0x50020218)..... | .471 |
| Table 502: UART3_CONFIG_REG (0x5002021C)..... | .472 |
| Table 503: UART3_SRBR_STHR0_REG (0x50020230)..... | .472 |
| Table 504: UART3_SRBR_STHR1_REG (0x50020234)..... | .473 |
| Table 505: UART3_SRBR_STHR2_REG (0x50020238)..... | .474 |
| Table 506: UART3_SRBR_STHR3_REG (0x5002023C)..... | .475 |
| Table 507: UART3_SRBR_STHR4_REG (0x50020240)..... | .475 |
| Table 508: UART3_SRBR_STHR5_REG (0x50020244)..... | .476 |
| Table 509: UART3_SRBR_STHR6_REG (0x50020248)..... | .477 |
| Table 510: UART3_SRBR_STHR7_REG (0x5002024C)..... | .478 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|---|-----|
| Table 511: UART3_SRBR_STHR8_REG (0x50020250) | 478 |
| Table 512: UART3_SRBR_STHR9_REG (0x50020254) | 479 |
| Table 513: UART3_SRBR_STHR10_REG (0x50020258) | 480 |
| Table 514: UART3_SRBR_STHR11_REG (0x5002025C) | 480 |
| Table 515: UART3_SRBR_STHR12_REG (0x50020260) | 481 |
| Table 516: UART3_SRBR_STHR13_REG (0x50020264) | 482 |
| Table 517: UART3_SRBR_STHR14_REG (0x50020268) | 483 |
| Table 518: UART3_SRBR_STHR15_REG (0x5002026C) | 483 |
| Table 519: UART3_USR_REG (0x5002027C) | 484 |
| Table 520: UART3_TFL_REG (0x50020280) | 485 |
| Table 521: UART3_RFL_REG (0x50020284) | 485 |
| Table 522: UART3_SRR_REG (0x50020288) | 485 |
| Table 523: UART3_SRTS_REG (0x5002028C) | 486 |
| Table 524: UART3_SBCR_REG (0x50020290) | 486 |
| Table 525: UART3_SDMAM_REG (0x50020294) | 487 |
| Table 526: UART3_SFE_REG (0x50020298) | 487 |
| Table 527: UART3_SRT_REG (0x5002029C) | 487 |
| Table 528: UART3_STET_REG (0x500202A0) | 488 |
| Table 529: UART3_HTX_REG (0x500202A4) | 488 |
| Table 530: UART3_DMASA_REG (0x500202A8) | 488 |
| Table 531: UART3_DLF_REG (0x500202C0) | 489 |
| Table 532: UART3_RAR_REG (0x500202C4) | 489 |
| Table 533: UART3_TAR_REG (0x500202C8) | 489 |
| Table 534: UART3_LCR_EXT (0x500202CC) | 489 |
| Table 535: UART3_CTRL_REG (0x500202E0) | 490 |
| Table 536: UART3_TIMER_REG (0x500202E4) | 491 |
| Table 537: UART3_ERR_CTRL_REG (0x500202E8) | 491 |
| Table 538: UART3_IRQ_STATUS_REG (0x500202EC) | 492 |
| Table 539: UART3_UCV_REG (0x500202F8) | 492 |
| Table 540: UART3_CTR_REG (0x500202FC) | 492 |
| Table 541: Register map USB | 492 |
| Table 542: USB_MCTRL_REG (0x50040000) | 494 |
| Table 543: USB_XCVDIAG_REG (0x50040004) | 495 |
| Table 544: USB_TCR_REG (0x50040008) | 495 |
| Table 545: USB_UTR_REG (0x5004000C) | 496 |
| Table 546: USB_FAR_REG (0x50040010) | 496 |
| Table 547: USB_NFSR_REG (0x50040014) | 497 |
| Table 548: USB_MAEV_REG (0x50040018) | 497 |
| Table 549: USB_MAMSK_REG (0x5004001C) | 499 |
| Table 550: USB_ALTEV_REG (0x50040020) | 499 |
| Table 551: USB_ALTMSK_REG (0x50040024) | 500 |
| Table 552: USB_TXEV_REG (0x50040028) | 500 |
| Table 553: USB_TXMSK_REG (0x5004002C) | 501 |
| Table 554: USB_RXEV_REG (0x50040030) | 501 |
| Table 555: USB_RXMSK_REG (0x50040034) | 502 |
| Table 556: USB_NAKEV_REG (0x50040038) | 502 |
| Table 557: USB_NAKMSK_REG (0x5004003C) | 502 |
| Table 558: USB_FWEV_REG (0x50040040) | 503 |
| Table 559: USB_FWMSK_REG (0x50040044) | 503 |
| Table 560: USB_FNH_REG (0x50040048) | 503 |
| Table 561: USB_FNL_REG (0x5004004C) | 504 |
| Table 562: USB_UX20CDR_REG (0x5004007C) | 504 |
| Table 563: USB_EPC0_REG (0x50040080) | 505 |
| Table 564: USB_TXD0_REG (0x50040084) | 506 |
| Table 565: USB_TXS0_REG (0x50040088) | 506 |
| Table 566: USB_TXC0_REG (0x5004008C) | 507 |
| Table 567: USB_EP0_NAK_REG (0x50040090) | 507 |
| Table 568: USB_RXD0_REG (0x50040094) | 508 |
| Table 569: USB_RXS0_REG (0x50040098) | 508 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|---|------|
| Table 570: USB_RXC0_REG (0x5004009C) | .508 |
| Table 571: USB_EPC1_REG (0x500400A0)..... | .509 |
| Table 572: USB_TXD1_REG (0x500400A4)..... | .510 |
| Table 573: USB_TXS1_REG (0x500400A8)..... | .510 |
| Table 574: USB_TXC1_REG (0x500400AC) | .511 |
| Table 575: USB_EPC2_REG (0x500400B0)..... | .512 |
| Table 576: USB_RXD1_REG (0x500400B4) | .513 |
| Table 577: USB_RXS1_REG (0x500400B8)..... | .513 |
| Table 578: USB_RXC1_REG (0x500400BC)..... | .514 |
| Table 579: USB_EPC3_REG (0x500400C0) | .514 |
| Table 580: USB_TXD2_REG (0x500400C4) | .515 |
| Table 581: USB_TXS2_REG (0x500400C8)..... | .515 |
| Table 582: USB_TXC2_REG (0x500400CC) | .516 |
| Table 583: USB_EPC4_REG (0x500400D0) | .518 |
| Table 584: USB_RXD2_REG (0x500400D4) | .518 |
| Table 585: USB_RXS2_REG (0x500400D8) | .519 |
| Table 586: USB_RXC2_REG (0x500400DC) | .519 |
| Table 587: USB_EPC5_REG (0x500400E0) | .520 |
| Table 588: USB_TXD3_REG (0x500400E4) | .521 |
| Table 589: USB_TXS3_REG (0x500400E8) | .521 |
| Table 590: USB_TXC3_REG (0x500400EC) | .522 |
| Table 591: USB_EPC6_REG (0x500400F0) | .523 |
| Table 592: USB_RXD3_REG (0x500400F4) | .524 |
| Table 593: USB_RXS3_REG (0x500400F8) | .524 |
| Table 594: USB_RXC3_REG (0x500400FC) | .525 |
| Table 595: USB_DMA_CTRL_REG (0x500401A0)..... | .525 |
| Table 596: USB_CHARGER_CTRL_REG (0x500401A8)..... | .526 |
| Table 597: USB_CHARGER_STAT_REG (0x500401AC) | .526 |
| Table 598: Register map Version | .527 |
| Table 599: CHIP_ID1_REG (0x50040200)..... | .527 |
| Table 600: CHIP_ID2_REG (0x50040204)..... | .527 |
| Table 601: CHIP_ID3_REG (0x50040208) | .528 |
| Table 602: CHIP_ID4_REG (0x5004020C) | .528 |
| Table 603: CHIP_SWC_REG (0x50040210)..... | .528 |
| Table 604: CHIP_REVISION_REG (0x50040214) | .528 |
| Table 605: CHIP_TEST1_REG (0x500402F8) | .528 |
| Table 606: CHIP_TEST2_REG (0x500402FC) | .528 |
| Table 607: Register map WakeUp | .529 |
| Table 608: WKUP_CTRL_REG (0x50000100)..... | .529 |
| Table 609: WKUP_RESET_IRQ_REG (0x50000108)..... | .529 |
| Table 610: WKUP_SELECT_P0_REG (0x50000114)..... | .530 |
| Table 611: WKUP_SELECT_P1_REG (0x50000118) | .530 |
| Table 612: WKUP_POL_P0_REG (0x50000128) | .530 |
| Table 613: WKUP_POL_P1_REG (0x5000012C) | .530 |
| Table 614: WKUP_STATUS_P0_REG (0x5000013C) | .530 |
| Table 615: WKUP_STATUS_P1_REG (0x50000140) | .530 |
| Table 616: WKUP_CLEAR_P0_REG (0x50000148) | .530 |
| Table 617: WKUP_CLEAR_P1_REG (0x5000014C) | .531 |
| Table 618: WKUP_SEL_GPIO_P0_REG (0x50000154) | .531 |
| Table 619: WKUP_SEL_GPIO_P1_REG (0x50000158) | .531 |
| Table 620: Register map WDOG | .531 |
| Table 621: WATCHDOG_REG (0x50000700) | .531 |
| Table 622: WATCHDOG_CTRL_REG (0x50000704) | .532 |
| Table 623: Register map GPADC | .533 |
| Table 624: SDADC_CTRL_REG (0x50020800)..... | .533 |
| Table 625: SDADC_TEST_REG (0x50020808) | .534 |
| Table 626: SDADC_GAIN_CORR_REG (0x5002080C) | .535 |
| Table 627: SDADC_OFFSET_CORR_REG (0x50020810) | .535 |
| Table 628: SDADC_CLEAR_INT_REG (0x50020814) | .535 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|------|
| Table 629: SDADC_RESULT_REG (0x50020818) | .535 |
| Table 630: GP_ADC_CTRL_REG (0x50030900) | .535 |
| Table 631: GP_ADC_CTRL2_REG (0x50030904) | .537 |
| Table 632: GP_ADC_CTRL3_REG (0x50030908) | .537 |
| Table 633: GP_ADC_OFFP_REG (0x5003090C) | .538 |
| Table 634: GP_ADC_OFFN_REG (0x50030910) | .538 |
| Table 635: GP_ADC_CLEAR_INT_REG (0x50030914) | .538 |
| Table 636: GP_ADC_RESULT_REG (0x50030918) | .538 |
| Table 637: Register map GPIO | .538 |
| Table 638: P0_DATA_REG (0x50020A00) | .540 |
| Table 639: P1_DATA_REG (0x50020A04) | .540 |
| Table 640: P0_SET_DATA_REG (0x50020A08) | .541 |
| Table 641: P1_SET_DATA_REG (0x50020A0C) | .541 |
| Table 642: P0_RESET_DATA_REG (0x50020A10) | .541 |
| Table 643: P1_RESET_DATA_REG (0x50020A14) | .541 |
| Table 644: P0_00_MODE_REG (0x50020A18) | .541 |
| Table 645: P0_01_MODE_REG (0x50020A1C) | .543 |
| Table 646: P0_02_MODE_REG (0x50020A20) | .543 |
| Table 647: P0_03_MODE_REG (0x50020A24) | .543 |
| Table 648: P0_04_MODE_REG (0x50020A28) | .544 |
| Table 649: P0_05_MODE_REG (0x50020A2C) | .544 |
| Table 650: P0_06_MODE_REG (0x50020A30) | .544 |
| Table 651: P0_07_MODE_REG (0x50020A34) | .545 |
| Table 652: P0_08_MODE_REG (0x50020A38) | .545 |
| Table 653: P0_09_MODE_REG (0x50020A3C) | .545 |
| Table 654: P0_10_MODE_REG (0x50020A40) | .545 |
| Table 655: P0_11_MODE_REG (0x50020A44) | .546 |
| Table 656: P0_12_MODE_REG (0x50020A48) | .546 |
| Table 657: P0_13_MODE_REG (0x50020A4C) | .546 |
| Table 658: P0_14_MODE_REG (0x50020A50) | .547 |
| Table 659: P0_15_MODE_REG (0x50020A54) | .547 |
| Table 660: P0_16_MODE_REG (0x50020A58) | .547 |
| Table 661: P0_17_MODE_REG (0x50020A5C) | .548 |
| Table 662: P0_18_MODE_REG (0x50020A60) | .548 |
| Table 663: P0_19_MODE_REG (0x50020A64) | .548 |
| Table 664: P0_20_MODE_REG (0x50020A68) | .548 |
| Table 665: P0_21_MODE_REG (0x50020A6C) | .549 |
| Table 666: P0_22_MODE_REG (0x50020A70) | .549 |
| Table 667: P0_23_MODE_REG (0x50020A74) | .549 |
| Table 668: P0_24_MODE_REG (0x50020A78) | .550 |
| Table 669: P0_25_MODE_REG (0x50020A7C) | .550 |
| Table 670: P0_26_MODE_REG (0x50020A80) | .550 |
| Table 671: P0_27_MODE_REG (0x50020A84) | .551 |
| Table 672: P0_28_MODE_REG (0x50020A88) | .551 |
| Table 673: P0_29_MODE_REG (0x50020A8C) | .551 |
| Table 674: P0_30_MODE_REG (0x50020A90) | .551 |
| Table 675: P0_31_MODE_REG (0x50020A94) | .552 |
| Table 676: P1_00_MODE_REG (0x50020A98) | .552 |
| Table 677: P1_01_MODE_REG (0x50020A9C) | .552 |
| Table 678: P1_02_MODE_REG (0x50020AA0) | .553 |
| Table 679: P1_03_MODE_REG (0x50020AA4) | .553 |
| Table 680: P1_04_MODE_REG (0x50020AA8) | .553 |
| Table 681: P1_05_MODE_REG (0x50020AAC) | .553 |
| Table 682: P1_06_MODE_REG (0x50020AB0) | .554 |
| Table 683: P1_07_MODE_REG (0x50020AB4) | .554 |
| Table 684: P1_08_MODE_REG (0x50020AB8) | .554 |
| Table 685: P1_09_MODE_REG (0x50020ABC) | .555 |
| Table 686: P1_10_MODE_REG (0x50020AC0) | .555 |
| Table 687: P1_11_MODE_REG (0x50020AC4) | .555 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 688: P1_12_MODE_REG (0x50020AC8)..... | 555 |
| Table 689: P1_13_MODE_REG (0x50020ACC)..... | 556 |
| Table 690: P1_14_MODE_REG (0x50020AD0)..... | 556 |
| Table 691: P1_15_MODE_REG (0x50020AD4)..... | 556 |
| Table 692: P1_16_MODE_REG (0x50020AD8)..... | 557 |
| Table 693: P1_17_MODE_REG (0x50020ADC)..... | 557 |
| Table 694: P1_18_MODE_REG (0x50020AE0)..... | 557 |
| Table 695: P1_19_MODE_REG (0x50020AE4)..... | 558 |
| Table 696: P1_20_MODE_REG (0x50020AE8)..... | 558 |
| Table 697: P1_21_MODE_REG (0x50020AEC)..... | 558 |
| Table 698: P1_22_MODE_REG (0x50020AF0)..... | 558 |
| Table 699: P0_PADPWR_CTRL_REG (0x50020AF4)..... | 559 |
| Table 700: P1_PADPWR_CTRL_REG (0x50020AF8)..... | 559 |
| Table 701: GPIO_CLK_SEL_REG (0x50020AFC)..... | 559 |
| Table 702: PAD_WEAK_CTRL_REG (0x50020B00)..... | 560 |
| Table 703: Register map GPREG | 562 |
| Table 704: SET_FREEZE_REG (0x50040300)..... | 562 |
| Table 705: RESET_FREEZE_REG (0x50040304)..... | 562 |
| Table 706: DEBUG_REG (0x50040308)..... | 563 |
| Table 707: GP_STATUS_REG (0x5004030C)..... | 564 |
| Table 708: GP_CONTROL_REG (0x50040310)..... | 564 |
| Table 709: USBPAD_REG (0x50040318)..... | 564 |
| Table 710: Register map I2C..... | 565 |
| Table 711: I2C_CON_REG (0x50020600)..... | 568 |
| Table 712: I2C_TAR_REG (0x50020604)..... | 569 |
| Table 713: I2C_SAR_REG (0x50020608)..... | 570 |
| Table 714: I2C_HS_MADDR_REG (0x5002060C) | 570 |
| Table 715: I2C_DATA_CMD_REG (0x50020610)..... | 570 |
| Table 716: I2C_SS_SCL_HCNT_REG (0x50020614) | 572 |
| Table 717: I2C_SS_SCL_LCNT_REG (0x50020618) | 572 |
| Table 718: I2C_FS_SCL_HCNT_REG (0x5002061C) | 572 |
| Table 719: I2C_FS_SCL_LCNT_REG (0x50020620) | 573 |
| Table 720: I2C_HS_SCL_HCNT_REG (0x50020624) | 573 |
| Table 721: I2C_HS_SCL_LCNT_REG (0x50020628) | 573 |
| Table 722: I2C_INTR_STAT_REG (0x5002062C) | 574 |
| Table 723: I2C_INTR_MASK_REG (0x50020630)..... | 576 |
| Table 724: I2C_RAW_INTR_STAT_REG (0x50020634) | 577 |
| Table 725: I2C_RX_TL_REG (0x50020638)..... | 579 |
| Table 726: I2C_TX_TL_REG (0x5002063C)..... | 579 |
| Table 727: I2C_CLR_INTR_REG (0x50020640)..... | 579 |
| Table 728: I2C_CLR_RX_UNDER_REG (0x50020644) | 580 |
| Table 729: I2C_CLR_RX_OVER_REG (0x50020648) | 580 |
| Table 730: I2C_CLR_TX_OVER_REG (0x5002064C) | 580 |
| Table 731: I2C_CLR_RD_REQ_REG (0x50020650) | 580 |
| Table 732: I2C_CLR_TX_ABRT_REG (0x50020654) | 580 |
| Table 733: I2C_CLR_RX_DONE_REG (0x50020658) | 581 |
| Table 734: I2C_CLR_ACTIVITY_REG (0x5002065C) | 581 |
| Table 735: I2C_CLR_STOP_DET_REG (0x50020660) | 581 |
| Table 736: I2C_CLR_START_DET_REG (0x50020664) | 581 |
| Table 737: I2C_CLR_GEN_CALL_REG (0x50020668) | 581 |
| Table 738: I2C_ENABLE_REG (0x5002066C) | 582 |
| Table 739: I2C_STATUS_REG (0x50020670) | 582 |
| Table 740: I2C_TXFLR_REG (0x50020674)..... | 584 |
| Table 741: I2C_RXFLR_REG (0x50020678) | 584 |
| Table 742: I2C_SDA_HOLD_REG (0x5002067C) | 584 |
| Table 743: I2C_TX_ABRT_SOURCE_REG (0x50020680) | 584 |
| Table 744: I2C_DMA_CR_REG (0x50020688) | 587 |
| Table 745: I2C_DMA_TDRL_REG (0x5002068C) | 587 |
| Table 746: I2C_DMA_RDRL_REG (0x50020690) | 587 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|---|-----|
| Table 747: I2C_SDA_SETUP_REG (0x50020694) | 587 |
| Table 748: I2C_ACK_GENERAL_CALL_REG (0x50020698) | 588 |
| Table 749: I2C_ENABLE_STATUS_REG (0x5002069C) | 588 |
| Table 750: I2C_IC_FS_SPKLEN_REG (0x500206A0) | 589 |
| Table 751: I2C_IC_HS_SPKLEN_REG (0x500206A4) | 589 |
| Table 752: I2C2_CON_REG (0x50020700) | 590 |
| Table 753: I2C2_TAR_REG (0x50020704) | 591 |
| Table 754: I2C2_SAR_REG (0x50020708) | 592 |
| Table 755: I2C2_HS_MADDR_REG (0x5002070C) | 592 |
| Table 756: I2C2_DATA_CMD_REG (0x50020710) | 592 |
| Table 757: I2C2_SS_SCL_HCNT_REG (0x50020714) | 594 |
| Table 758: I2C2_SS_SCL_LCNT_REG (0x50020718) | 594 |
| Table 759: I2C2_FS_SCL_HCNT_REG (0x5002071C) | 594 |
| Table 760: I2C2_FS_SCL_LCNT_REG (0x50020720) | 595 |
| Table 761: I2C2_HS_SCL_HCNT_REG (0x50020724) | 595 |
| Table 762: I2C2_HS_SCL_LCNT_REG (0x50020728) | 595 |
| Table 763: I2C2_INTR_STAT_REG (0x5002072C) | 596 |
| Table 764: I2C2_INTR_MASK_REG (0x50020730) | 598 |
| Table 765: I2C2_RAW_INTR_STAT_REG (0x50020734) | 599 |
| Table 766: I2C2_RX_TL_REG (0x50020738) | 601 |
| Table 767: I2C2_TX_TL_REG (0x5002073C) | 601 |
| Table 768: I2C2_CLR_INTR_REG (0x50020740) | 601 |
| Table 769: I2C2_CLR_RX_UNDER_REG (0x50020744) | 602 |
| Table 770: I2C2_CLR_RX_OVER_REG (0x50020748) | 602 |
| Table 771: I2C2_CLR_TX_OVER_REG (0x5002074C) | 602 |
| Table 772: I2C2_CLR_RD_REQ_REG (0x50020750) | 602 |
| Table 773: I2C2_CLR_TX_ABRT_REG (0x50020754) | 602 |
| Table 774: I2C2_CLR_RX_DONE_REG (0x50020758) | 603 |
| Table 775: I2C2_CLR_ACTIVITY_REG (0x5002075C) | 603 |
| Table 776: I2C2_CLR_STOP_DET_REG (0x50020760) | 603 |
| Table 777: I2C2_CLR_START_DET_REG (0x50020764) | 603 |
| Table 778: I2C2_CLR_GEN_CALL_REG (0x50020768) | 603 |
| Table 779: I2C2_ENABLE_REG (0x5002076C) | 604 |
| Table 780: I2C2_STATUS_REG (0x50020770) | 604 |
| Table 781: I2C2_TXFLR_REG (0x50020774) | 606 |
| Table 782: I2C2_RXFLR_REG (0x50020778) | 606 |
| Table 783: I2C2_SDA_HOLD_REG (0x5002077C) | 606 |
| Table 784: I2C2_TX_ABRT_SOURCE_REG (0x50020780) | 606 |
| Table 785: I2C2_DMA_CR_REG (0x50020788) | 609 |
| Table 786: I2C2_DMA_TDLR_REG (0x5002078C) | 609 |
| Table 787: I2C2_DMA_RDLR_REG (0x50020790) | 609 |
| Table 788: I2C2_SDA_SETUP_REG (0x50020794) | 609 |
| Table 789: I2C2_ACK_GENERAL_CALL_REG (0x50020798) | 610 |
| Table 790: I2C2_ENABLE_STATUS_REG (0x5002079C) | 610 |
| Table 791: I2C2_IC_FS_SPKLEN_REG (0x500207A0) | 611 |
| Table 792: I2C2_IC_HS_SPKLEN_REG (0x500207A4) | 611 |
| Table 793: Register map PDC | 612 |
| Table 794: PDC_CTRL0_REG (0x50000200) | 613 |
| Table 795: PDC_CTRL1_REG (0x50000204) | 613 |
| Table 796: PDC_CTRL2_REG (0x50000208) | 614 |
| Table 797: PDC_CTRL3_REG (0x5000020C) | 614 |
| Table 798: PDC_CTRL4_REG (0x50000210) | 615 |
| Table 799: PDC_CTRL5_REG (0x50000214) | 615 |
| Table 800: PDC_CTRL6_REG (0x50000218) | 615 |
| Table 801: PDC_CTRL7_REG (0x5000021C) | 616 |
| Table 802: PDC_CTRL8_REG (0x50000220) | 616 |
| Table 803: PDC_CTRL9_REG (0x50000224) | 617 |
| Table 804: PDC_CTRL10_REG (0x50000228) | 617 |
| Table 805: PDC_CTRL11_REG (0x5000022C) | 617 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|---|-----|
| Table 806: PDC_CTRL12_REG (0x500000230)..... | 618 |
| Table 807: PDC_CTRL13_REG (0x500000234)..... | 618 |
| Table 808: PDC_CTRL14_REG (0x500000238)..... | 618 |
| Table 809: PDC_CTRL15_REG (0x50000023C) | 619 |
| Table 810: PDC_ACKNOWLEDGE_REG (0x500000280)..... | 619 |
| Table 811: PDC_PENDING_REG (0x500000284)..... | 619 |
| Table 812: PDC_PENDING_SNC_REG (0x500000288) | 620 |
| Table 813: PDC_PENDING_CM33_REG (0x50000028C)..... | 620 |
| Table 814: PDC_PENDING_CMAC_REG (0x500000290) | 620 |
| Table 815: PDC_SET_PENDING_REG (0x500000294)..... | 620 |
| Table 816: Register map DCDC | 620 |
| Table 817: DCDC_CTRL1_REG (0x500000304) | 621 |
| Table 818: DCDC_CTRL2_REG (0x500000308) | 621 |
| Table 819: DCDC_V14_REG (0x50000030C) | 622 |
| Table 820: DCDC_VDD_REG (0x500000310)..... | 623 |
| Table 821: DCDC_V18_REG (0x500000314)..... | 623 |
| Table 822: DCDC_V18P_REG (0x500000318)..... | 624 |
| Table 823: DCDC_STATUS1_REG (0x500000320) | 624 |
| Table 824: DCDC_IRQ_STATUS_REG (0x500000330) | 626 |
| Table 825: DCDC_IRQ_CLEAR_REG (0x500000334) | 626 |
| Table 826: DCDC_IRQ_MASK_REG (0x500000338) | 626 |
| Table 827: Register map DMA | 627 |
| Table 828: DMA0_A_START_REG (0x50040800)..... | 628 |
| Table 829: DMA0_B_START_REG (0x50040804)..... | 628 |
| Table 830: DMA0_INT_REG (0x50040808)..... | 628 |
| Table 831: DMA0_LEN_REG (0x5004080C) | 629 |
| Table 832: DMA0_CTRL_REG (0x50040810) | 629 |
| Table 833: DMA0_IDX_REG (0x50040814)..... | 631 |
| Table 834: DMA1_A_START_REG (0x50040820)..... | 631 |
| Table 835: DMA1_B_START_REG (0x50040824)..... | 631 |
| Table 836: DMA1_INT_REG (0x50040828) | 631 |
| Table 837: DMA1_LEN_REG (0x5004082C) | 631 |
| Table 838: DMA1_CTRL_REG (0x50040830) | 632 |
| Table 839: DMA1_IDX_REG (0x50040834) | 633 |
| Table 840: DMA2_A_START_REG (0x50040840)..... | 634 |
| Table 841: DMA2_B_START_REG (0x50040844)..... | 634 |
| Table 842: DMA2_INT_REG (0x50040848) | 634 |
| Table 843: DMA2_LEN_REG (0x5004084C) | 634 |
| Table 844: DMA2_CTRL_REG (0x50040850) | 634 |
| Table 845: DMA2_IDX_REG (0x50040854) | 636 |
| Table 846: DMA3_A_START_REG (0x50040860)..... | 636 |
| Table 847: DMA3_B_START_REG (0x50040864)..... | 637 |
| Table 848: DMA3_INT_REG (0x50040868) | 637 |
| Table 849: DMA3_LEN_REG (0x5004086C) | 637 |
| Table 850: DMA3_CTRL_REG (0x50040870) | 637 |
| Table 851: DMA3_IDX_REG (0x50040874) | 639 |
| Table 852: DMA4_A_START_REG (0x50040880)..... | 639 |
| Table 853: DMA4_B_START_REG (0x50040884)..... | 639 |
| Table 854: DMA4_INT_REG (0x50040888) | 639 |
| Table 855: DMA4_LEN_REG (0x5004088C) | 640 |
| Table 856: DMA4_CTRL_REG (0x50040890) | 640 |
| Table 857: DMA4_IDX_REG (0x50040894) | 642 |
| Table 858: DMA5_A_START_REG (0x500408A0) | 642 |
| Table 859: DMA5_B_START_REG (0x500408A4) | 642 |
| Table 860: DMA5_INT_REG (0x500408A8)..... | 642 |
| Table 861: DMA5_LEN_REG (0x500408AC) | 642 |
| Table 862: DMA5_CTRL_REG (0x500408B0) | 643 |
| Table 863: DMA5_IDX_REG (0x500408B4) | 644 |
| Table 864: DMA6_A_START_REG (0x500408C0) | 645 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 865: DMA6_B_START_REG (0x500408C4) | 645 |
| Table 866: DMA6_INT_REG (0x500408C8) | 645 |
| Table 867: DMA6_LEN_REG (0x500408CC) | 645 |
| Table 868: DMA6_CTRL_REG (0x500408D0) | 645 |
| Table 869: DMA6_IDX_REG (0x500408D4) | 647 |
| Table 870: DMA7_A_START_REG (0x500408E0) | 648 |
| Table 871: DMA7_B_START_REG (0x500408E4) | 648 |
| Table 872: DMA7_INT_REG (0x500408E8) | 648 |
| Table 873: DMA7_LEN_REG (0x500408EC) | 648 |
| Table 874: DMA7_CTRL_REG (0x500408F0) | 648 |
| Table 875: DMA7_IDX_REG (0x500408F4) | 650 |
| Table 876: DMA_REQ_MUX_REG (0x50040900) | 651 |
| Table 877: DMA_INT_STATUS_REG (0x50040904) | 652 |
| Table 878: DMA_CLEAR_INT_REG (0x50040908) | 653 |
| Table 879: DMA_INT_MASK_REG (0x5004090C) | 654 |
| Table 880: Register map CHARGER | 654 |
| Table 881: CHARGER_CTRL_REG (0x50040400) | 655 |
| Table 882: CHARGER_TEST_CTRL_REG (0x50040404) | 660 |
| Table 883: CHARGER_STATUS_REG (0x50040408) | 660 |
| Table 884: CHARGER_VOLTAGE_PARAM_REG (0x5004040C) | 666 |
| Table 885: CHARGER_CURRENT_PARAM_REG (0x50040410) | 668 |
| Table 886: CHARGER_TEMPSET_PARAM_REG (0x50040414) | 671 |
| Table 887: CHARGER_PRE_CHARGE_TIMER_REG (0x50040418) | 673 |
| Table 888: CHARGER_CC_CHARGE_TIMER_REG (0x5004041C) | 674 |
| Table 889: CHARGER_CV_CHARGE_TIMER_REG (0x50040420) | 674 |
| Table 890: CHARGER_TOTAL_CHARGE_TIMER_REG (0x50040424) | 675 |
| Table 891: CHARGER_JEITA_V_CHARGE_REG (0x50040428) | 675 |
| Table 892: CHARGER_JEITA_V_PRECHARGE_REG (0x5004042C) | 676 |
| Table 893: CHARGER_JEITA_V_REPLENISH_REG (0x50040430) | 676 |
| Table 894: CHARGER_JEITA_V_OVP_REG (0x50040434) | 676 |
| Table 895: CHARGER_JEITA_CURRENT_REG (0x50040438) | 676 |
| Table 896: CHARGER_VBAT_COMP_TIMER_REG (0x5004043C) | 677 |
| Table 897: CHARGER_VOVP_COMP_TIMER_REG (0x50040440) | 677 |
| Table 898: CHARGER_TDIE_COMP_TIMER_REG (0x50040444) | 678 |
| Table 899: CHARGER_TBAT_MON_TIMER_REG (0x50040448) | 679 |
| Table 900: CHARGER_TBAT_COMP_TIMER_REG (0x5004044C) | 679 |
| Table 901: CHARGER_THOT_COMP_TIMER_REG (0x50040450) | 680 |
| Table 902: CHARGER_PWR_UP_TIMER_REG (0x50040454) | 680 |
| Table 903: CHARGER_STATE_IRQ_MASK_REG (0x50040458) | 681 |
| Table 904: CHARGER_ERROR_IRQ_MASK_REG (0x5004045C) | 682 |
| Table 905: CHARGER_STATE_IRQ_STATUS_REG (0x50040460) | 682 |
| Table 906: CHARGER_ERROR_IRQ_STATUS_REG (0x50040464) | 683 |
| Table 907: CHARGER_STATE_IRQ_CLR_REG (0x50040468) | 685 |
| Table 908: CHARGER_ERROR_IRQ_CLR_REG (0x5004046C) | 685 |
| Table 909: Register map CRG | 686 |
| Table 910: CLK_AMBA_REG (0x50000000) | 688 |
| Table 911: CLK_RADIO_REG (0x50000010) | 689 |
| Table 912: CLK_CTRL_REG (0x50000014) | 689 |
| Table 913: CLK_TMR_REG (0x50000018) | 690 |
| Table 914: CLK_SWITCH2XTAL_REG (0x5000001C) | 690 |
| Table 915: PMU_CTRL_REG (0x50000020) | 690 |
| Table 916: SYS_CTRL_REG (0x50000024) | 691 |
| Table 917: SYS_STAT_REG (0x50000028) | 692 |
| Table 918: CLK_RC32K_REG (0x5000003C) | 693 |
| Table 919: CLK_XTAL32K_REG (0x50000040) | 693 |
| Table 920: CLK_RC32M_REG (0x50000044) | 693 |
| Table 921: CLK_RCX_REG (0x50000048) | 694 |
| Table 922: CLK_RTCDIV_REG (0x5000004C) | 694 |
| Table 923: BANDGAP_REG (0x50000050) | 695 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|------|
| Table 924: VBUS_IRQ_MASK_REG (0x50000054) | .695 |
| Table 925: VBUS_IRQ_CLEAR_REG (0x50000058)..... | .695 |
| Table 926: BOD_CTRL_REG (0x50000060)..... | .695 |
| Table 927: BOD_LVL_CTRL0_REG (0x50000064) | .696 |
| Table 928: BOD_LVL_CTRL1_REG (0x50000068) | .696 |
| Table 929: BOD_LVL_CTRL2_REG (0x5000006C) | .697 |
| Table 930: P0_PAD_LATCH_REG (0x50000070) | .697 |
| Table 931: P0_SET_PAD_LATCH_REG (0x50000074) | .697 |
| Table 932: P0_RESET_PAD_LATCH_REG (0x50000078) | .697 |
| Table 933: P1_PAD_LATCH_REG (0x5000007C)..... | .697 |
| Table 934: P1_SET_PAD_LATCH_REG (0x50000080) | .697 |
| Table 935: P1_RESET_PAD_LATCH_REG (0x50000084) | .698 |
| Table 936: BOD_STATUS_REG (0x50000090)..... | .698 |
| Table 937: POR_VBAT_CTRL_REG (0x50000094) | .698 |
| Table 938: POR_PIN_REG (0x50000098) | .699 |
| Table 939: POR_TIMER_REG (0x5000009C) | .700 |
| Table 940: LDO_VDDD_HIGH_CTRL_REG (0x500000A0) | .700 |
| Table 941: BIAS_VREF_SEL_REG (0x500000A4) | .700 |
| Table 942: RESET_STAT_REG (0x500000BC)..... | .701 |
| Table 943: RAM_PWR_CTRL_REG (0x500000C0) | .701 |
| Table 944: SECURE_BOOT_REG (0x500000CC) | .702 |
| Table 945: DISCHARGE_RAIL_REG (0x500000D4) | .702 |
| Table 946: ANA_STATUS_REG (0x500000EC) | .703 |
| Table 947: POWER_CTRL_REG (0x500000F0)..... | .703 |
| Table 948: PMU_SLEEP_REG (0x500000F4) | .705 |
| Table 949: PMU_TRIM_REG (0x500000F8)..... | .706 |
| Table 950: CLK_FREQ_TRIM_REG (0x50010000) | .706 |
| Table 951: TRIM_CTRL_REG (0x50010010)..... | .706 |
| Table 952: XTALRDY_CTRL_REG (0x50010018) | .707 |
| Table 953: XTALRDY_STAT_REG (0x5001001C) | .707 |
| Table 954: XTAL32M_CTRL0_REG (0x50010030) | .707 |
| Table 955: XTAL32M_CTRL1_REG (0x50010034) | .708 |
| Table 956: XTAL32M_CTRL2_REG (0x50010038) | .709 |
| Table 957: XTAL32M_CTRL3_REG (0x5001003C) | .710 |
| Table 958: XTAL32M_CTRL4_REG (0x50010040) | .711 |
| Table 959: XTAL32M_STAT0_REG (0x50010050) | .711 |
| Table 960: XTAL32M_STAT1_REG (0x50010054) | .711 |
| Table 961: PLL_SYS_CTRL1_REG (0x50010060) | .712 |
| Table 962: PLL_SYS_CTRL2_REG (0x50010064) | .713 |
| Table 963: PLL_SYS_CTRL3_REG (0x50010068) | .713 |
| Table 964: PLL_SYS_STATUS_REG (0x50010070) | .713 |
| Table 965: CLK_COM_REG (0x50020904) | .714 |
| Table 966: SET_CLK_COM_REG (0x50020908) | .715 |
| Table 967: RESET_CLK_COM_REG (0x5002090C) | .716 |
| Table 968: CLK_PER_REG (0x50030C04) | .716 |
| Table 969: SET_CLK_PER_REG (0x50030C08) | .717 |
| Table 970: RESET_CLK_PER_REG (0x50030C0C) | .717 |
| Table 971: PCM_DIV_REG (0x50030C40) | .718 |
| Table 972: PCM_FDIV_REG (0x50030C44) | .718 |
| Table 973: PDM_DIV_REG (0x50030C48) | .718 |
| Table 974: SRC_DIV_REG (0x50030C4C) | .719 |
| Table 975: CLK_SYS_REG (0x50040500) | .719 |
| Table 976: BATCHECK_REG (0x50040504) | .719 |
| Table 977: Register map CACHE | .720 |
| Table 978: CACHE_CTRL1_REG (0x100C0000) | .720 |
| Table 979: CACHE_LNSIZECFG_REG (0x100C0004) | .720 |
| Table 980: CACHE_ASSOCCFG_REG (0x100C0008) | .721 |
| Table 981: CACHE_CTRL2_REG (0x100C0020) | .721 |
| Table 982: CACHE_MRM_HITS_REG (0x100C0028) | .722 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | |
|--|-----|
| Table 983: CACHE_MRM_MISSES_REG (0x100C002C)..... | 722 |
| Table 984: CACHE_MRM_CTRL_REG (0x100C0030)..... | 722 |
| Table 985: CACHE_MRM_TINT_REG (0x100C0034)..... | 722 |
| Table 986: CACHE_MRM_MISSES_THRES_REG (0x100C0038) | 723 |
| Table 987: CACHE_MRM_HITS_THRES_REG (0x100C003C) | 723 |
| Table 988: CACHE_FLASH_REG (0x100C0040)..... | 723 |
| Table 989: SWD_RESET_REG (0x100C0050)..... | 724 |
| Table 990: Register map APU..... | 724 |
| Table 991: SRC1_CTRL_REG (0x50030600)..... | 725 |
| Table 992: SRC1_IN_FS_REG (0x50030604)..... | 727 |
| Table 993: SRC1_OUT_FS_REG (0x50030608)..... | 727 |
| Table 994: SRC1_IN1_REG (0x5003060C)..... | 728 |
| Table 995: SRC1_IN2_REG (0x50030610)..... | 728 |
| Table 996: SRC1_OUT1_REG (0x50030614)..... | 728 |
| Table 997: SRC1_OUT2_REG (0x50030618)..... | 728 |
| Table 998: APU_MUX_REG (0x5003061C)..... | 728 |
| Table 999: COEF10_SET1_REG (0x50030620)..... | 729 |
| Table 1000: COEF32_SET1_REG (0x50030624)..... | 729 |
| Table 1001: COEF54_SET1_REG (0x50030628)..... | 729 |
| Table 1002: COEF76_SET1_REG (0x5003062C)..... | 729 |
| Table 1003: COEF98_SET1_REG (0x50030630)..... | 729 |
| Table 1004: COEF0A_SET1_REG (0x50030634) | 729 |
| Table 1005: PCM1_CTRL_REG (0x50030700) | 730 |
| Table 1006: PCM1_IN1_REG (0x50030704) | 730 |
| Table 1007: PCM1_IN2_REG (0x50030708) | 730 |
| Table 1008: PCM1_OUT1_REG (0x5003070C)..... | 731 |
| Table 1009: PCM1_OUT2_REG (0x50030710) | 731 |
| Table 1010: Register map ANAMISC | 731 |
| Table 1011: CLK_REF_SEL_REG (0x50030B10) | 731 |
| Table 1012: CLK_REF_CNT_REG (0x50030B14) | 732 |
| Table 1013: CLK_REF_VAL_REG (0x50030B18) | 732 |
| Table 1014: Register map AES_HASH | 732 |
| Table 1015: CRYPTO_CTRL_REG (0x30040000) | 732 |
| Table 1016: CRYPTO_START_REG (0x30040004) | 734 |
| Table 1017: CRYPTO_FETCH_ADDR_REG (0x30040008) | 735 |
| Table 1018: CRYPTO_LEN_REG (0x3004000C) | 735 |
| Table 1019: CRYPTO_DEST_ADDR_REG (0x30040010) | 735 |
| Table 1020: CRYPTO_STATUS_REG (0x30040014) | 735 |
| Table 1021: CRYPTO_CLRIRQ_REG (0x30040018) | 736 |
| Table 1022: CRYPTO_MREG0_REG (0x3004001C) | 736 |
| Table 1023: CRYPTO_MREG1_REG (0x30040020) | 736 |
| Table 1024: CRYPTO_MREG2_REG (0x30040024) | 736 |
| Table 1025: CRYPTO_MREG3_REG (0x30040028) | 736 |
| Table 1026: CRYPTO_KEYS_START (0x30040100) | 737 |
| Table 1027: Ordering Information (Samples) | 738 |
| Table 1028: Ordering Information (Production) | 738 |

1 Block Diagram

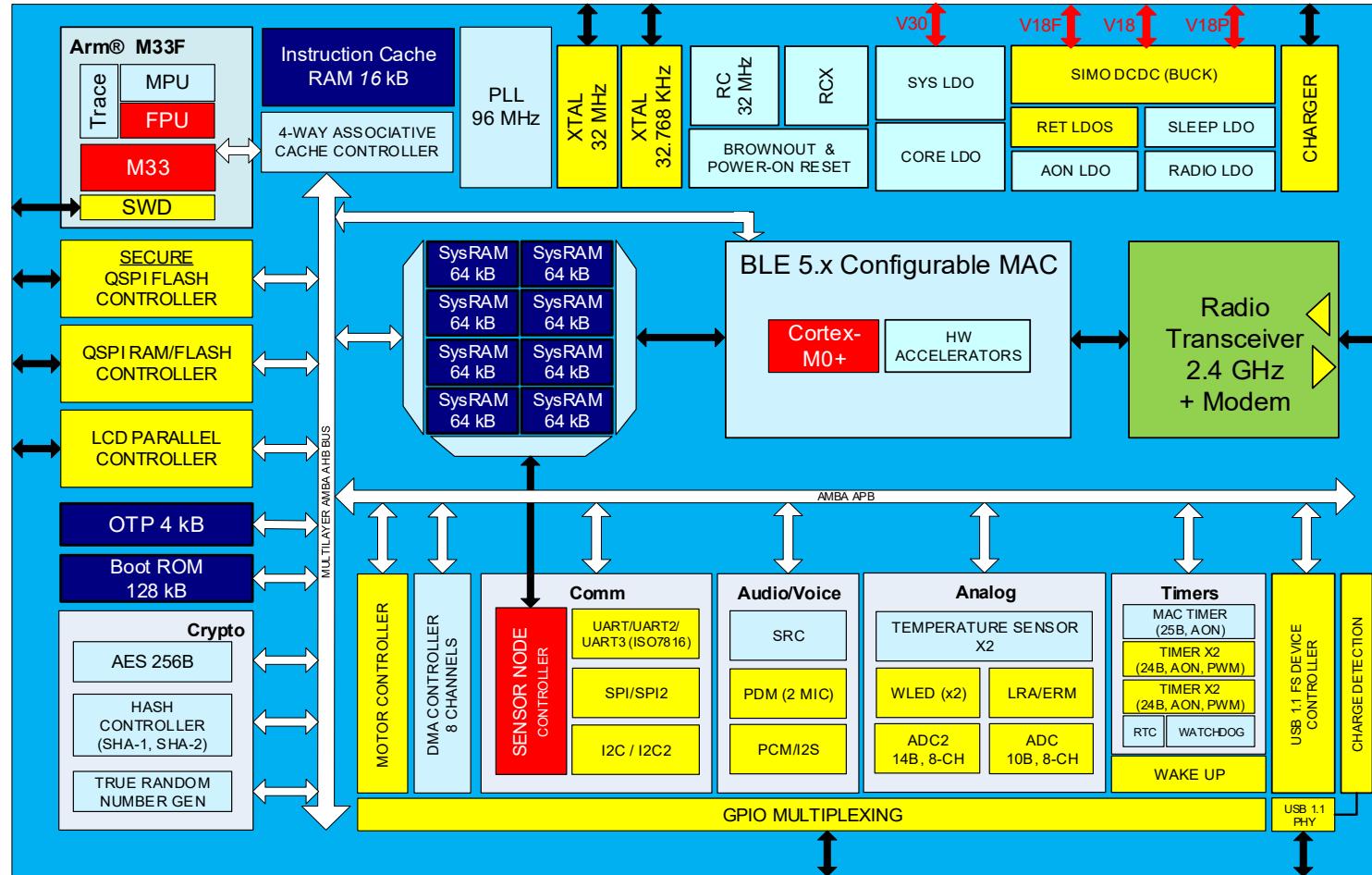


Figure 1: DA1469x Block Diagram

2 Application Information

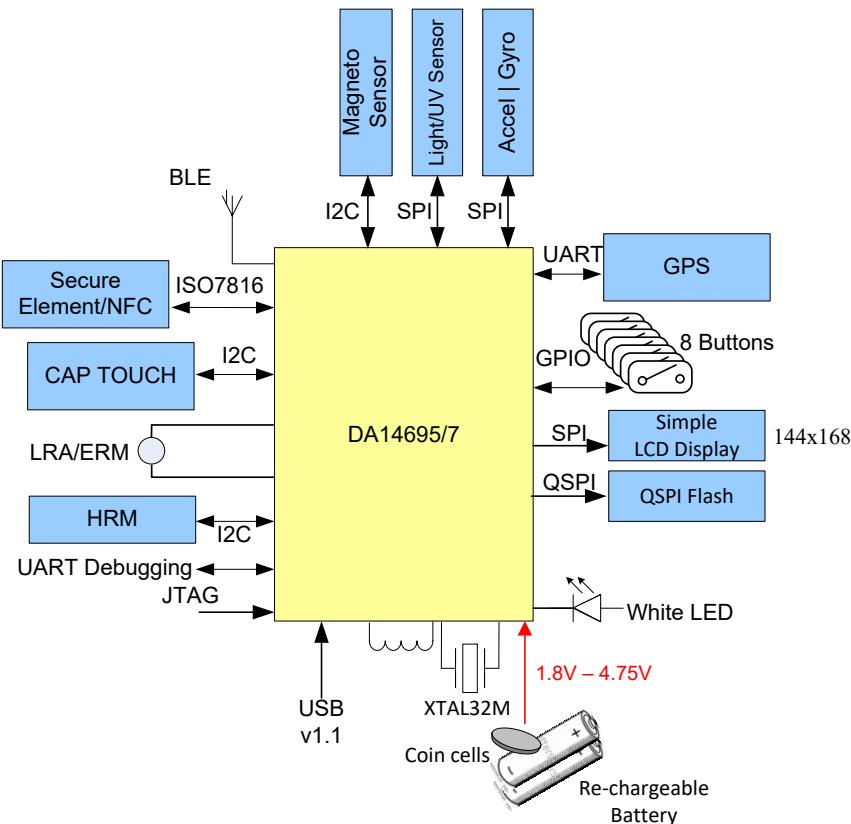


Figure 2: Activity Tracker

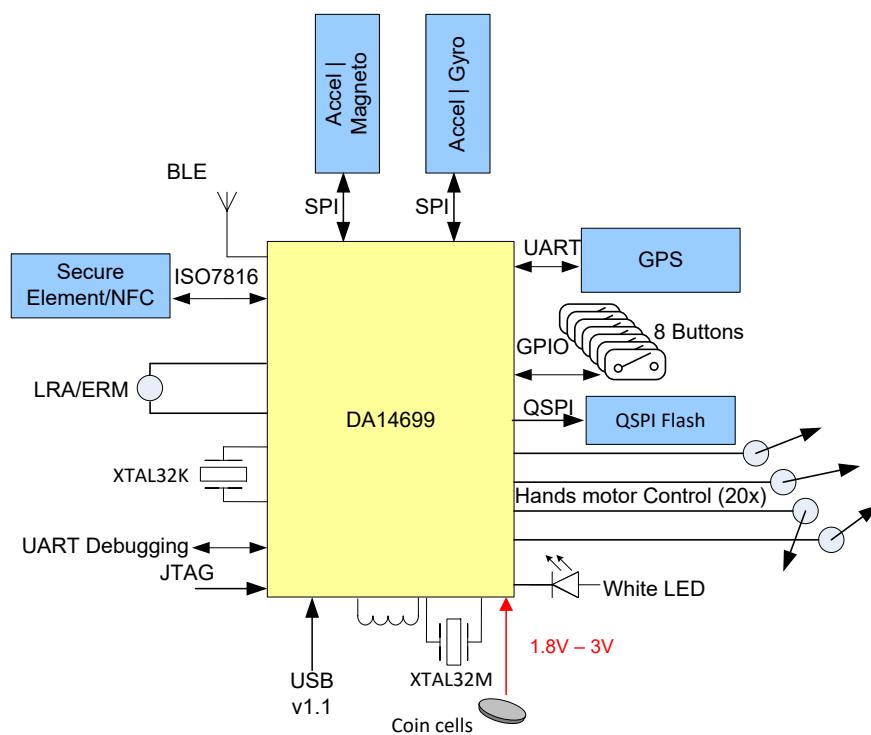
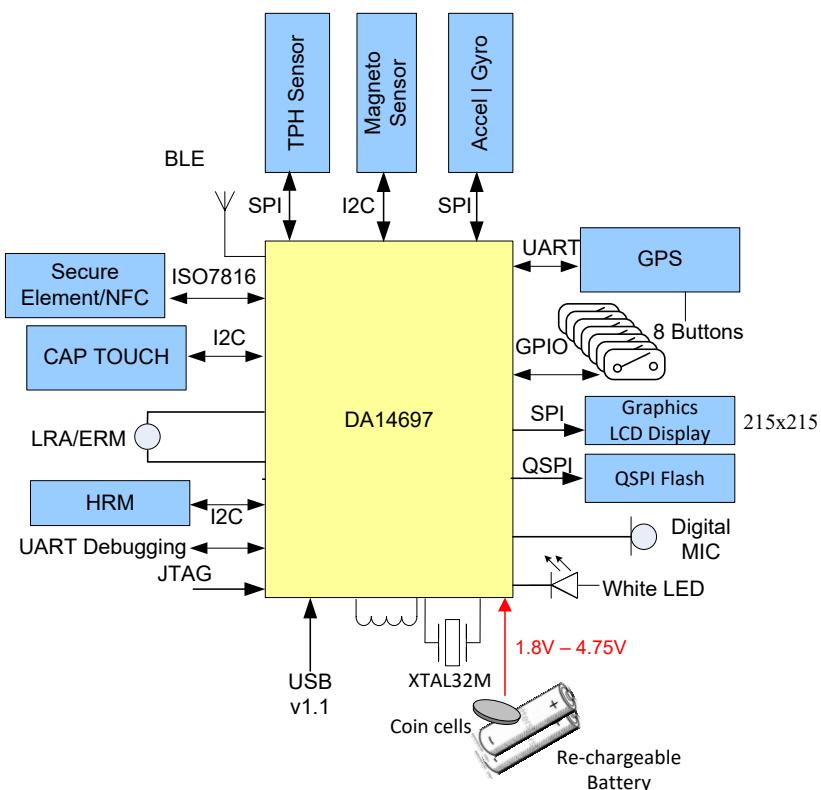


Figure 3: Analog Watch

Multi-core Bluetooth 5.2 SoC family with system PMU

Figure 4: Sport Watch

Note: All the external components are directly powered by the DA1469x Power Management Unit (PMU).

3 DA1469x Product Family

Table 1 presents the differentiation between the members of the DA1469x product family.

Table 1: DA1469x Product Family Differentiation

| Features | DA14691 | DA14695 | DA14697 | DA14699 |
|------------------------|---------|---------|----------|----------|
| Available RAM | 384kB | 512kB | 512kB | 512kB |
| Charger | ✗ | ✓ | ✓ | ✓ |
| LCD Controller | ✗ | ✓ | ✓ | ✓ |
| Haptics Controller | ✗ | ✗ | ✓ | ✓ |
| LEDs | ✗ | ✗ | ✓ | ✓ |
| QSPI RAM Controller | ✗ | ✓ | ✓ | ✓ |
| Motor Controller | ✗ | ✗ | ✗ | ✓ |
| GPIOs (P1_12 to P1_22) | ✗ | ✗ | ✓ | ✓ |
| Rest of the features | ✓ | ✓ | ✓ | ✓ |
| Package | VFBGA86 | VFBGA86 | VFBGA100 | VFBGA100 |

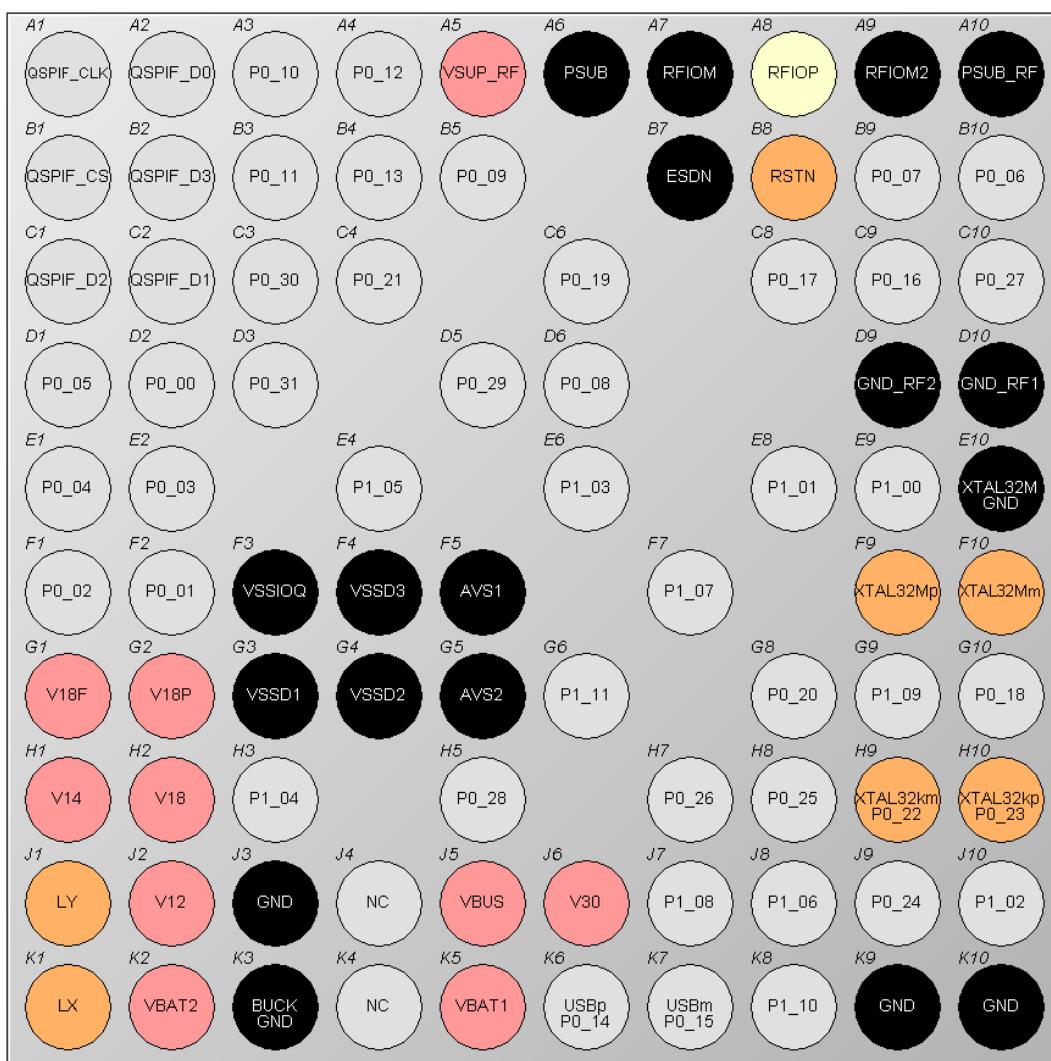
4 Pinout

The DA1469x comes in two packages:

- A 6 mm x 6mm VFBGA with 86 balls
- A 5 mm x 5mm VFBGA with 100 balls

The actual pin-ball assignment is depicted in the following sections:

4.1 VFBGA86 Pinout



DA14691/5

TOP VIEW

■ Digital ■ Analog ■ Radio ■ Power ■ GND

Figure 5: VFBGA86 Ball Assignment

Table 2: DA14691/5 Pin Description

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|---|--------------------|-------------------|-----------------------------|-------------|--|
| General Purpose I/Os (fixed pin assignment; additional functions are programmable via Px_xx_MODE_REG) | | | | | |
| D2 | P0_00 QSPIR_D0 | DIO DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 0. |
| F2 | P0_01 QSPIR_D1 | DIO DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 1. |
| F1 | P0_02 QSPIR_D2 | DIO DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 2. |
| E2 | P0_03 QSPIR_D3 | DIO DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 3. |
| E1 | P0_04 QSPIR_CS | DIO DO | 4.8 | I-PU | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) OUTPUT. QSPI RAM chip select (active LOW). |
| D1 | P0_05 QSPIR_CLK | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) OUTPUT. QSPI RAM clock. |
| B10 | P0_06 SDADC_GND | DIO/ RDS AI | 4.8/ 0.48 (Note 4) | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the ΣΔ ADC reference ground. |
| B9 | P0_07 NTC Input | DIO/ RDS AI | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for battery NTC (feedback). |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|-------------------------|-----------|------------|-------------|--|
| D6 | P0_08 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | GPADC_2 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 2. |
| | SDADC_2 UART Boot RX | AI DI | | | INPUT. Analog input for the ΣΔ ADC, channel 2. INPUT. UART Receive data input during boot. |
| B5 | P0_09 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | GPADC_3 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 3 |
| | SDADC_3 UART Boot TX | AI DO | | | INPUT. Analog input for the ΣΔ ADC, channel 3. OUTPUT. UART Transmit data output during boot. |
| A3 | P0_10 | DIO | 4.8 | I-PU | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | M33_SWDIO | DIO | | | INPUT / OUTPUT. Arm Cortex-M33 JTAG data I/O signal. |
| B3 | P0_11 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | M33_SWCLK | DI | | | INPUT. Arm Cortex-M33 JTAG clock signal. |
| A4 | P0_12 | DIO | 4.8 | I-PU | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | CMAC_SWDIO | DIO | | | INPUT / OUTPUT. Arm Cortex-M0+ JTAG data I/O signal. |
| | XTAL32M | DO | | | OUTPUT. XTAL32MHz oscillator clock |
| B4 | P0_13 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | CMAC_SWCLK RC32M | DI DO | | | INPUT. Arm Cortex-M0+ JTAG clock signal. OUTPUT. RC32M clock signal output (square wave). |
| K6 | P0_14 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Does not contain state retention mechanism during power down. |
| | USBp XTAL32k | AIO DO | | | INPUT/OUTPUT. Analog USB Full Speed D+ signal. OUTPUT. XTAL32k clock signal output (square wave). |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|------------------|-------------|--------------|-------------|--|
| K7 | P0_15 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Does not contain state retention mechanism during power down. |
| | USBm DIVN | AIO DO | | | INPUT/OUTPUT. Analog USB Full Speed D- signal. OUTPUT. DIVN clock signal output (square wave). |
| C9 | P0_16 | DIO/ RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | SDADC_REF RCX | AI DO | | | INPUT. Analog input for the $\Sigma\Delta$ ADC reference voltage. OUTPUT. RCX clock signal output (square wave). |
| C8 | P0_17 | DIO/ RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | RC32k | DO | | | OUTPUT. RC32k clock signal output (square wave). |
| G10 | P0_18 | DIO/ RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| C6 | P0_19 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| G8 | P0_20 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| C4 | P0_21 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| H9 | P0_22 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | XTAL32km | AO | | | OUTPUT. Analog output of the XTAL32K crystal oscillator. |
| H10 | P0_23 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | XTAL32kp | AI DI | | | INPUT. Analog input of the XTAL32K crystal oscillator. INPUT. Digital input for an external clock (square wave). |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|-----------------------|----------|--------------|-------------|--|
| J9 | P0_24 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_TE | DI | | | INPUT. LCD Tearing effect signal. Used to synchronize the CPU to the frame memory writing. |
| H8 | P0_25 | DIO/RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | GPADC_1 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 1. |
| | SDADC_1 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 1. |
| H7 | P0_26 | DIO/RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_VCK | DO | | | OUTPUT. LCD Shift clock for the vertical driver. |
| C10 | P0_27 | DIO/RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_ENB PLCD_ENAB | DO DO | | | OUTPUT. LCD Write enable signal for the pixel memory. OUTPUT. Parallel LCD Indication signal for valid data on the LCD data bus. |
| H5 | P0_28 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_VST PLCD_VSYNC | DO DO | | | OUTPUT. LCD Start signal for the vertical driver. OUTPUT. Parallel LCD Indication signal for resetting the LCD row pointer to top of the display. |
| D5 | P0_29 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_HCK PLCD_CLK | DO DO | | | OUTPUT. LCD Shift clock signal for the horizontal driver. OUTPUT. Parallel LCD Clock signal. |
| C3 | P0_30 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_HST PLCD_HSYNC | DO DO | | | OUTPUT. LCD Start signal for the horizontal driver. OUTPUT. Parallel LCD Indication signal for resetting the LCD column pointer to the edge of the display. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|------------|---------|--------------|-------------|--|
| D3 | P0_31 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_XRST | DO | | | OUTPUT. LCD Reset signal for the horizontal and vertical driver. |
| E9 | P1_00 | DIO/RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | NTC Supply | AO | | | OUTPUT. Power supply for the battery NTC sensor. |
| E8 | P1_01 | DIO/RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | Timer.PWM | DO | | | OUTPUT. Timer/PWM output (PWM) in Sleep mode. |
| J10 | P1_02 | DIO/RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_BLUE0 | DO | | | OUTPUT. LCD Blue image data output, bit 0. |
| E6 | P1_03 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_BLUE1 | DO | | | OUTPUT. LCD Blue image data output, bit 1. |
| H3 | P1_04 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_GREEN0 | DO | | | OUTPUT. LCD Green image data output, bit 0. |
| E4 | P1_05 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_GREEN1 | DO | | | OUTPUT. LCD Green image data output, bit 1. |
| J8 | P1_06 | DIO/RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | Timer2.PWM | DO | | | OUTPUT. Timer2/PWM output (PWM2) in Sleep mode. |
| F7 | P1_07 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | LCD_RED0 | DO | | | OUTPUT. LCD Red image data output, bit 0. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------------------|---------------------------------------|---------------------|--------------|-------------|---|
| J7 | P1_08 LCD_RED1 | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Red image data output, bit 1. |
| G9 | P1_09 GPADC_0 SDADC_0 | DIO/RDS AI AI | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 0. INPUT. Analog input for the ΣΔ ADC, channel 0. |
| K8 | P1_10 LCD_VCOM/FRP LCD_EXTCOMIN | DIO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Combined LCD signals: - VCOM: Common electrode driving signal (62.5Hz). - FRP: Liquid crystal driving signal (62.5Hz). OUTPUT. LCD COM Inversion Signal Input (1Hz). |
| G6 | P1_11 LCD_XFRP | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Liquid crystal driving signal (62.5Hz inverted). |
| Debug Interface | | | | | |
| A3 | M33_SWDIO | DIO | 4.8 | I-PU | INPUT / OUTPUT. Arm Cortex-M33 JTAG data I/O signal. |
| B3 | M33_SWCLK | DI | 4.8 | I-PD | INPUT. Arm Cortex-M33 JTAG clock signal. |
| A4 | CMAC_SWDIO | DIO | 4.8 | I-PU | INPUT / OUTPUT. Arm Cortex-M0+ JTAG data I/O signal. |
| B4 | CMAC_SWCLK | DI | 4.8 | I-PD | INPUT. Arm Cortex-M0+ JTAG clock signal. |
| Clocks | | | | | |
| H9 | XTAL32km | AO | | | OUTPUT. Analog output of the 32.768kHz XTAL oscillator. |
| H10 | XTAL32kp | AI DI | | | INPUT. Analog input of the 32.768kHz XTAL crystal oscillator. INPUT. Digital input for an external clock (square wave). |
| E10 | XTAL32M_GND | AI | | | INPUT. Analog input ground of the 32MHz XTAL oscillator. |
| F10 | XTAL32Mm | AO | | | OUTPUT. Crystal output for the 32 MHz XTAL oscillator. |
| F9 | XTAL32Mp | AI | | | INPUT. Crystal input for the 32 MHz XTAL oscillator. |
| QSPI Flash Interface | | | | | |
| A2 | QSPIF_D0 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 0. |
| C2 | QSPIF_D1 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 1. |
| C1 | QSPIF_D2 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 2. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|---|-----------|--------------|------------|-------------|--|
| B2 | QSPIF_D3 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 3. |
| A1 | QSPIF_CLK | DO | | | OUTPUT. QSPI Flash clock. |
| B1 | QSPIF_CS | DO | | | OUTPUT. QSPI Flash chip select (active LOW). This output is HIGH in the default and reset states. No pull-up resistor is required. |
| QSPI RAM Interface (mapped on port P0_yy) | | | | | |
| D2 | QSPIR_D0 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 0. Mapped on P0_00. |
| F2 | QSPIR_D1 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 1. Mapped on P0_01. |
| F1 | QSPIR_D2 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 2. Mapped on P0_02. |
| E2 | QSPIR_D3 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 3. Mapped on P0_03. |
| E1 | QSPIR_CS | DO | | | OUTPUT. QSPI RAM clock. Mapped on P0_04. |
| D1 | QSPIR_CLK | DO | | | OUTPUT. QSPI RAM chip select (active LOW). Mapped on P0_05. |
| SPI Bus Interface (mapped on port Px_yy) | | | | | |
| | SPI_DI | DI | | | INPUT. SPI data input. (Note 2) |
| | SPI_DO | DO | | | OUTPUT. SPI data output. (Note 3) |
| | SPI_CLK | DIO | | | INPUT/OUTPUT. SPI clock. |
| | SPI_EN | DI | | | INPUT. SPI clock enable (chip select). |
| | SPI2_DI | DI | | | INPUT. SPI 2 data input. (Note 2) |
| | SPI2_DO | DO | | | OUTPUT. SPI 2 data output. (Note 3) |
| | SPI2_CLK | DIO | | | INPUT/OUTPUT. SPI 2 clock. |
| | SPI2_EN | DI | | | INPUT. SPI 2 clock enable (chip select). |
| I2C Bus Interface (mapped on port Px_yy) | | | | | |
| | I2C_SCL | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus clock with open drain port. Supports bit stretching by a slave in open drain mode. |
| | I2C_SDA | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus data with open drain port. |
| | I2C2_SCL | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus 2 clock with open drain port. Supports bit stretching by a slave in open drain mode. |
| | I2C2_SDA | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus 2 data with open drain port. |
| UART Interface (mapped on port Px_yy) | | | | | |
| | UART_RX | DI | | | INPUT. UART receive data. |
| | UART_TX | DO | | | OUTPUT. UART transmit data. |
| | UART2_RX | DI | | | INPUT. UART 2 receive data. |
| | UART2_TX | DO | | | OUTPUT. UART 2 transmit data. |
| | UART2_CTS | DI | | | INPUT. UART 2 clear to send. |
| | UART2_RTS | DO | | | OUTPUT. UART 2 request to send. |
| | UART3_RX | DI | | | INPUT. UART 3 receive data. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|--|--|------|------------|-------------|---|
| | UART3_TX | DO | | | OUTPUT. UART 3 transmit data. |
| | UART3_CTS | DI | | | INPUT. UART 3 clear to send. |
| | UART3_RTS | DO | | | OUTPUT. UART 3 request to send. |
| ISO7816 Bus Interface (mapped on port Px_yy) | | | | | |
| | ISO7816_CLK | DO | | | OUTPUT. Smart card (ISO7816) clock signal |
| | ISO7816_DATA | DIO | | | INPUT / OUTPUT. Smart card (ISO7816) I/O data signal. |
| | ISO7816_RST | DO | | | OUTPUT. Smart card (ISO7816) reset signal. |
| | ISO7816_CI | DI | | | INPUT. Smart card (ISO7816) inserted signal. |
| PCM Interface (mapped on port Px_yy) | | | | | |
| | PCM_DI | DI | | | INPUT. PCM input data. |
| | PCM_DO | DO | | | OUTPUT. PCM output data. |
| | PCM_FSC | DIO | | | INPUT / OUTPUT. PCM Frame synchronization. |
| | PCM_CLK | DIO | | | INPUT / OUTPUT. PCM Clock |
| PDM Interface (mapped on port Px_yy) | | | | | |
| | PDM_DATA | DIO | | | INPUT/OUTPUT. PDM data. |
| | PDM_CLK | DO | | | OUTPUT. PDM clock output. |
| LCD Controller Interface | | | | | |
| | LCD_SPI_DC <small>(Note 1)</small> | DO | | | OUTPUT. LCD data/command select. |
| | LCD_SPI_DO <small>(Note 1)</small> | DO | | | OUTPUT. LCD SPI data output. |
| | LCD_SPI_CLK <small>(Note 1)</small> | DO | | | OUTPUT. LCD SPI clock. |
| | LCD_SPI_EN <small>(Note 1)</small> | DO | | | OUTPUT. LCD SPI clock enable (chip select). |
| F7 | LCD_RED0 <small>(Note 1)</small> | DO | | | LCD Red image data output, bit 0. |
| J7 | LCD_RED1 <small>(Note 1)</small> | DO | | | LCD Red image data output, bit 1. |
| H3 | LCD_GREEN0 <small>(Note 1)</small> | DO | | | LCD Green image data output, bit 0. |
| E4 | LCD_GREEN1 <small>(Note 1)</small> | DO | | | LCD Green image data output, bit 1. |
| J10 | LCD_BLUE0 <small>(Note 1)</small> | DO | | | LCD Blue image data output, bit 0. |
| E6 | LCD_BLUE1 <small>(Note 1)</small> | DO | | | LCD Blue image data output, bit 1. |
| C10 | PLCD_ENAB <small>(Note 1)</small> | DO | | | OUTPUT. Parallel LCD Indication signal for valid data on the LCD data bus. |
| D5 | PLCD_CLK <small>(Note 1)</small> | DO | | | OUTPUT. Parallel LCD Clock signal. |
| C3 | PLCD_HSYNC <small>(Note 1)</small> | DO | | | OUTPUT. Parallel LCD Indication signal for resetting the LCD column pointer to the edge of the display. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|------------------|--|------|------------|-------------|---|
| H5 | PLCD_VSYNC (Note 1) | DO | | | OUTPUT. Parallel LCD Indication signal for resetting the LCD row pointer to top of the display. |
| C10 | LCD_ENB (Note 1) | DO | | | OUTPUT. LCD Write enable signal for the pixel memory. |
| C3 | LCD_HST (Note 1) | DO | | | OUTPUT. LCD Start signal for the horizontal driver. |
| D5 | LCD_HCK (Note 1) | DO | | | OUTPUT. LCD Shift clock signal for the horizontal driver. |
| H5 | LCD_VST (Note 1) | DO | | | OUTPUT. LCD Start signal for the vertical driver. |
| D3 | LCD_XRST (Note 1) | DO | | | OUTPUT. LCD Reset signal for the horizontal and vertical driver. |
| H7 | LCD_VCK (Note 1) | DO | | | OUTPUT. LCD Shift clock for the vertical driver. |
| J9 | LCD_TE (Note 1) | DI | | | INPUT. LCD Tearing effect signal. Used to synchronize the CPU to the frame memory writing. |
| K8 | LCD_VCOM / LCD_FRP (Note 1) | DO | | | OUTPUT. Combined LCD signals into a single pin: - VCOM: Common electrode driving signal (62.5Hz). - FRP: Liquid crystal driving signal (62.5Hz) |
| K8 | LCD_EXTCOMIN (Note 1) | DO | | | OUTPUT. LCD COM Inversion Signal Input (1Hz). |
| G6 | LCD_XFRP (Note 1) | DO | | | OUTPUT. LCD Liquid crystal driving signal (62.5Hz inverted). |
| Analog Interface | | | | | |
| G9 | GPADC_0 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 0. |
| H8 | GPADC_1 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 1. |
| D6 | GPADC_2 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 2. |
| B5 | GPADC_3 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 3. |
| G9 | SDADC_0 | AI | | | INPUT. Analog input for the $\Sigma\Delta$ ADC, channel 0. |
| H8 | SDADC_1 | AI | | | INPUT. Analog input for the $\Sigma\Delta$ ADC, channel 1. |
| D6 | SDADC_2 | AI | | | INPUT. Analog input for the $\Sigma\Delta$ ADC, channel 2. |
| B5 | SDADC_3 | AI | | | INPUT. Analog input for the $\Sigma\Delta$ ADC, channel 3. |
| B10 | SDADC_GND | AI | | | INPUT. Analog input for the $\Sigma\Delta$ ADC reference ground. |
| C9 | SDADC_REF | AI | | | INPUT. Analog input for the $\Sigma\Delta$ ADC reference voltage. |
| USB FS Interface | | | | | |
| K6 | USBp | AIO | | | INPUT/OUTPUT. Analog USB Full Speed D+ signal. |
| K7 | USBm | AIO | | | INPUT/OUTPUT. Analog USB Full Speed D- signal. |
| Radio Interface | | | | | |
| A8 | RFIOP | AIO | | | RF input/output. Impedance 50 Ω . |
| A7 | RFIOM | AIO | | | RF Ground |

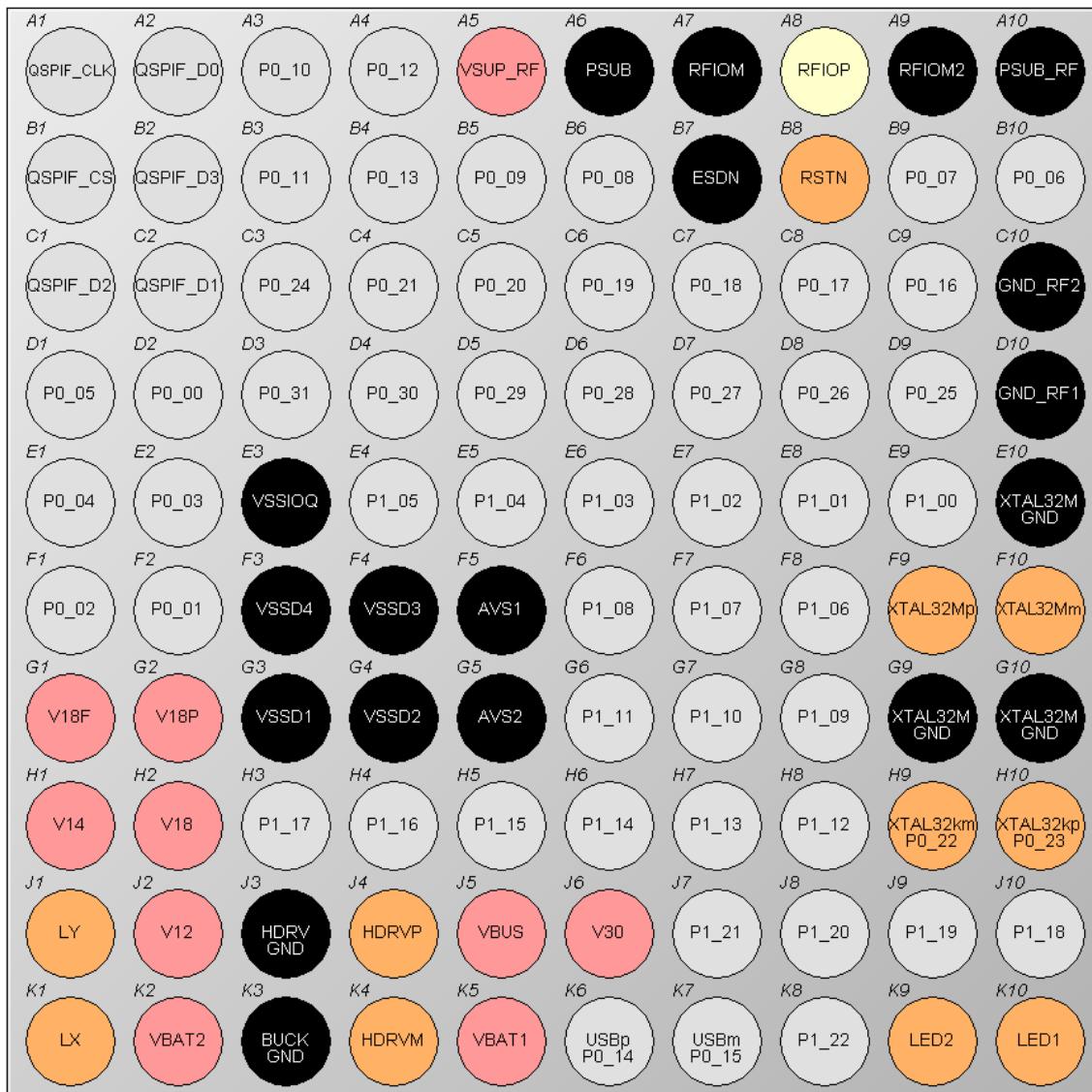
Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------------------|------------|----------|------------|-------------|---|
| Miscellaneous | | | | | |
| B9 | NTC Input | AI | | | INPUT. Analog input for battery NTC (feedback). |
| E9 | NTC Supply | AO | | | OUTPUT. Power supply for battery NTC sensor. |
| B8 | RSTn | AI | | | INPUT. Reset signal (active LOW). |
| Power Supply | | | | | |
| J2 | V12 | AO | | | OUTPUT. 1.2 V power rail. 10 µF decoupling capacitor required. |
| H1 | V14 | AIO | | | OUTPUT. 1.4 V power rail. 10 µF decoupling capacitor required. |
| A5 | VSUP_RF | AI | | | INPUT. Radio supply voltage. Connect to V14 externally. 10 µF decoupling capacitor required. |
| G1 | V18F | AIO | | | OUTPUT. 1.8 V power rail. Maximum current 50 mA. 0.1 µF decoupling capacitor required. Connect to V18P externally. |
| G2 | V18P | AIO | | | OUTPUT. 1.8 V power rail. Maximum current 50 mA. 22 µF decoupling capacitor required. |
| H2 | V18 | AIO | | | OUTPUT. 1.8 V power rail. Maximum current 50 mA. 22 µF decoupling capacitor required. |
| J6 | V30 | AIO | | | OUTPUT. 3.0 V power rail. Maximum current 150 mA. 4.7 µF decoupling capacitor required. |
| K1 | LX | AIO | | | INPUT/OUTPUT. Connection for the external DC-DC converter inductor. |
| J1 | LY | AIO | | | INPUT/OUTPUT. Connection for the external DC-DC converter inductor. |
| K5 | VBAT1 | AIO | | | INPUT. Battery connection 1 for LDO supply. |
| K2 | VBAT2 | AIO | | | INPUT. Battery connection 2 for DC-DC converter supply. |
| J5 | VBUS | AI AI | | | INPUT. USB bus voltage. INPUT. Battery charge voltage. |
| F5 | AVS1 | - | | | Analog Ground. |
| G5 | AVS2 | - | | | Analog Ground. |
| K3 | BUCK GND | - | | | DCDC Converter Ground. |
| G3 | VSSD1 | - | | | Digital Ground. |
| G4 | VSSD2 | - | | | Digital Ground. |
| F4 | VSSD3 | - | | | Digital Ground. |
| F3 | VSSIOQ | - | | | QSPI Ground. |
| B7 | ESDN | - | | | RF Ground. |
| A9 | RFIOM2 | - | | | RF Ground. |
| D10 | GND_RF1 | - | | | RF Ground. |
| D9 | GND_RF2 | - | | | RF Ground. |
| A10 | PSUB_RF | - | | | RF Ground. |
| A6 | PSUB | - | | | RF Ground. |
| J3 | GND | - | | | Connect to Ground. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|----------|------|------------|-------------|--------------------|
| J4 | NC | | | | Leave unconnected. |
| K4 | NC | | | | Leave unconnected. |
| K9 | GND | | | | Connect to Ground. |
| K10 | GND | | | | Connect to Ground. |

4.2 VFBGA100 Pinout


DA14697/9
TOP VIEW

| | | | | | | | | | |
|--|---------|--|--------|--|-------|--|-------|--|-----|
| | Digital | | Analog | | Radio | | Power | | GND |
|--|---------|--|--------|--|-------|--|-------|--|-----|

Figure 6: VFBGA100 Ball Assignment

Table 3: DA14697/9 Pin Description

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|---|---------------------------------|-----------------------|---------------------------------|-------------|--|
| General Purpose I/Os (fixed pin assignment; additional functions are programmable via Px_xx_MODE_REG) | | | | | |
| D2 | P0_00 QSPIR_D0 PG3_4 | DIO DIO DO | 4.8 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 0. OUTPUT. Pattern Generator 3, Output 4 |
| F2 | P0_01 QSPIR_D1 PG4_2 | DIO DIO DO | 4.8 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 1. OUTPUT. Pattern Generator 4, Output 2 |
| F1 | P0_02 QSPIR_D2 PG4_1 | DIO DIO DO | 4.8 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 2. OUTPUT. Pattern Generator 4, Output 1 |
| E2 | P0_03 QSPIR_D3 PG3_2 | DIO DIO DO | 4.8 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) INPUT / OUTPUT. QSPI RAM data I/O 3. OUTPUT. Pattern Generator 3, Output 2 |
| E1 | P0_04 QSPIR_CS PG3_1 | DIO DO DO | 4.8 4.8 | I-PU | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) OUTPUT. QSPI RAM chip select (active LOW). OUTPUT. Pattern Generator 3, Output 1 |
| D1 | P0_05 QSPIR_CLK PG3_3 | DIO DO DO | 4.8 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. (Note 5) OUTPUT. QSPI RAM clock. OUTPUT. Pattern Generator 3, Output 3 |
| B10 | P0_06 SDADC_GND | DIO/ RDS AI | 4.8/ 0.48 (Note 4) | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the ΣΔ ADC reference ground. |
| B9 | P0_07 NTC Input | DIO/ RDS AI | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for battery NTC (feedback). |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|--|-----------------------------|------------|-------------|--|
| B6 | P0_08 GPADC_2 SDADC_2 PG4_3 UART Boot RX | DIO AI AI DO DI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 2. INPUT. Analog input for the ΣΔ ADC, channel 2. OUTPUT. Pattern Generator 3, Output 3. INPUT. UART Receive data input during boot. |
| B5 | P0_09 GPADC_3 SDADC_3 PG4_4 UART Boot TX | DIO AI AI DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 3 INPUT. Analog input for the ΣΔ ADC, channel 3. OUTPUT. Pattern Generator 4, Output 4. OUTPUT. UART Transmit data output during boot. |
| A3 | P0_10 M33_SWDIO | DIO DIO | 4.8 | I-PU | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT / OUTPUT. Arm Cortex-M33 JTAG data I/O signal. |
| B3 | P0_11 M33_SWCLK | DIO DI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Arm Cortex-M33 JTAG clock signal. |
| A4 | P0_12 CMAC_SWDIO XTAL32M | DIO DIO DO | 4.8 | I-PU | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT / OUTPUT. Arm Cortex-M0+ JTAG data I/O signal. OUTPUT. LCD Start signal for the horizontal driver. |
| B4 | P0_13 CMAC_SWCLK RC32M | DIO DI DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Arm Cortex-M0+ JTAG clock signal. OUTPUT. RC32M clock signal output (square wave). |
| K6 | P0_14 USBp XTAL32k | DIO AIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Does not contain state retention mechanism during power down. INPUT/OUTPUT. Analog USB Full Speed D+ signal. OUTPUT. XTAL32k clock signal output (square wave). |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|------------------|-------------|--------------|-------------|--|
| K7 | P0_15 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Does not contain state retention mechanism during power down. |
| | USBm DIVN | AIO DO | | | INPUT/OUTPUT. Analog USB Full Speed D- signal. OUTPUT. DIVN clock signal output (square wave). |
| C9 | P0_16 | DIO/ RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | SDADC_REF RCX | AI DO | | | INPUT. Analog input for the $\Sigma\Delta$ ADC reference voltage. OUTPUT. RCX clock signal output (square wave). |
| C8 | P0_17 | DIO/ RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | PG2_1 RC32k | DO DO | | | OUTPUT. Pattern Generator 2, Output 1. OUTPUT. RC32k clock signal output (square wave). |
| C7 | P0_18 | DIO/ RDS | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | PG2_2 | DO | | | OUTPUT. Pattern Generator 2, Output 2 |
| C6 | P0_19 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | PG2_3 | DO | | | OUTPUT. Pattern Generator 2, Output 3. |
| C5 | P0_20 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | PG2_4 | DO | | | OUTPUT. Pattern Generator 2, Output 4. |
| C4 | P0_21 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | PG1_3 | DO | | | OUTPUT. Pattern Generator 1, Output 3. |
| H9 | P0_22 | DIO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. |
| | XTAL32km | AO | | | OUTPUT. Analog output of the XTAL32K crystal oscillator. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|---|-------------------------------|--------------|-------------|--|
| H10 | P0_23 XTAL32kp | DIO AI DI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input of the XTAL32K crystal oscillator. INPUT. Digital input for an external clock (square wave). |
| C3 | P0_24 PG1_4 LCD_TE | DIO DO DI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Pattern Generator 1, Output 4. INPUT. LCD Tearing effect signal. Used to synchronize the CPU to the frame memory writing. |
| D9 | P0_25 GPADC_1 SDADC_1 | DIO/ RDS AI AI | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 1. INPUT. Analog input for the ΣΔ ADC, channel 1. |
| D8 | P0_26 PG0_1 LCD_VCK | DIO/ RDS DO DO | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Pattern Generator 0, Output 1. OUTPUT. LCD Shift clock for the vertical driver. |
| D7 | P0_27 PG0_2 LCD_ENB PLCD_ENAB | DIO/ RDS DO DO DO | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Pattern Generator 0, Output 2. OUTPUT. LCD Write enable signal for the pixel memory. OUTPUT. Parallel LCD Indication signal for valid data on the LCD data bus. |
| D6 | P0_28 PG0_3 LCD_VST PLCD_VSYNC | DIO DO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Pattern Generator 0, Output 3. OUTPUT. LCD Start signal for the vertical driver. OUTPUT. Parallel LCD Indication signal for resetting the LCD row pointer to top of the display. |
| D5 | P0_29 PG0_4 LCD_HCK PLCD_CLK | DIO DO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Pattern Generator 0, Output 4. OUTPUT. LCD Shift clock signal for the horizontal driver. OUTPUT. Parallel LCD Clock signal. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|---|---------------------------|--------------|-------------|--|
| D4 | P0_30 PG1_1 LCD_HST PLCD_HSYNC | DIO DO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Pattern Generator 1, Output 1. OUTPUT. LCD Start signal for the horizontal driver. OUTPUT. Parallel LCD Indication signal for resetting the LCD column pointer to the edge of the display. |
| D3 | P0_31 PG1_2 LCD_XRST | DIO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Pattern Generator 1, Output 2. OUTPUT. LCD Reset signal for the horizontal and vertical driver. |
| E9 | P1_00 NTC Supply | DIO/RDS AO | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Power supply for the battery NTC sensor. |
| E8 | P1_01 Timer.PWM | DIO/RDS DO | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Timer/PWM output (PWM) in Sleep mode. |
| E7 | P1_02 LCD_BLUE0 | DIO/RDS DO | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Blue image data output, bit 0. |
| E6 | P1_03 LCD_BLUE1 | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Blue image data output, bit 1. |
| E5 | P1_04 LCD_GREEN0 | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Green image data output, bit 0. |
| E4 | P1_05 LCD_GREEN1 | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Green image data output, bit 1. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|---------------------------------------|-------------------------|--------------|-------------|---|
| F8 | P1_06 Timer2.PWM | DIO/ RDS DO | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Timer2/PWM output (PWM2) in Sleep mode. |
| F7 | P1_07 LCD_RED0 | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Red image data output, bit 0. |
| F6 | P1_08 LCD_RED1 | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Red image data output, bit 1. |
| G8 | P1_09 GPADC_0 SDADC_0 | DIO/ RDS AI AI | 4.8/ 0.48 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 0. INPUT. Analog input for the ΣΔ ADC, channel 0. |
| G7 | P1_10 LCD_VCOM/FRP LCD_EXTCOMIN | DIO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. Combined LCD signals: - VCOM: Common electrode driving signal (62.5Hz). - FRP: Liquid crystal driving signal (62.5Hz). OUTPUT. LCD COM Inversion Signal Input (1Hz). |
| G6 | P1_11 LCD_XFRP | DIO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Liquid crystal driving signal (62.5Hz inverted). |
| H8 | P1_12 GPADC_5 | DIO AI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 5. |
| H7 | P1_13 GPADC_4 LCD_VCK | DIO AI DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 4. OUTPUT. LCD Shift clock for the vertical driver. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|--|---------------------------|------------|-------------|---|
| H6 | P1_14 SDADC_4 LCD_ENB PLCD_ENAB | DIO AI DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the $\Sigma\Delta$ ADC, channel 4. OUTPUT. LCD Write enable signal for the pixel memory. OUTPUT. Parallel LCD Indication signal for valid data on the LCD data bus. |
| H5 | P1_15 LCD_VST PLCD_VSYNC | DIO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Start signal for the vertical driver. OUTPUT. Parallel LCD Indication signal for resetting the LCD row pointer to top of the display. |
| H4 | P1_16 LCD_HCK PLCD_CLK | DIO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. LCD Shift clock signal for the horizontal driver. OUTPUT. Parallel LCD Clock signal. |
| H3 | P1_17 LCD_HST PLCD_HSYNC | DIO DO DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. OUTPUT. XTAL32M clock signal output (square wave). OUTPUT. Parallel LCD Indication signal for resetting the LCD column pointer to the edge of the display. |
| J10 | P1_18 GPADC_6 | DIO AI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 6. |
| J9 | P1_19 GPADC_7 | DIO AI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the general-purpose ADC, channel 7. |
| J8 | P1_20 SDADC_5 | DIO AI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the $\Sigma\Delta$ ADC, channel 5. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|---|------------------------------|-----------------|------------|-------------|--|
| J7 | P1_21 SDADC_6 LCD_XRST | DIO AI DO | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the ΣΔ ADC, channel 6. OUTPUT. LCD Reset signal for the horizontal and vertical driver. |
| K8 | P1_22 SDADC_7 LCD_TE | DIO AI DI | 4.8 | I-PD | INPUT / OUTPUT with selectable pull up/down resistor and open drain functionality. Pull-down enabled during and after reset. General-purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. INPUT. Analog input for the ΣΔ ADC, channel 7. INPUT. LCD Tearing effect signal. Used to synchronize the CPU to the frame memory writing. |
| Debug Interface | | | | | |
| A3 | M33_SWDIO | DIO | 4.8 | I-PU | INPUT / OUTPUT. Arm Cortex-M33 JTAG data I/O signal. |
| B3 | M33_SWCLK | DI | 4.8 | I-PD | INPUT. Arm Cortex-M33 JTAG clock signal. |
| A4 | CMAC_SWDIO | DIO | 4.8 | I-PU | INPUT / OUTPUT. Arm Cortex-M0+ JTAG data I/O signal. |
| B4 | CMAC_SWCLK | DI | 4.8 | I-PD | INPUT. Arm Cortex-M0+ JTAG clock signal. |
| Clocks | | | | | |
| H9 | XTAL32km | AO | | | OUTPUT. Analog output of the 32.768kHz XTAL oscillator. |
| H10 | XTAL32kp | AI DI | | | INPUT. Analog input of the 32.768kHz XTAL crystal oscillator. INPUT. Digital input for an external clock (square wave). |
| E10, G9, G10 | XTAL32M_GND | AI | | | INPUT. Analog input ground of the 32MHz XTAL oscillator. |
| F10 | XTAL32Mm | AO | | | OUTPUT. Crystal output for the 32 MHz XTAL oscillator. |
| F9 | XTAL32Mp | AI | | | INPUT. Crystal input for the 32 MHz XTAL oscillator. |
| QSPI Flash Interface | | | | | |
| A2 | QSPIF_D0 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 0. |
| C2 | QSPIF_D1 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 1. |
| C1 | QSPIF_D2 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 2. |
| B2 | QSPIF_D3 | DIO | | | INPUT / OUTPUT. QSPI Flash I/O data 3. |
| A1 | QSPIF_CLK | DO | | | OUTPUT. QSPI Flash clock. |
| B1 | QSPIF_CS | DO | | | OUTPUT. QSPI Flash chip select (active LOW). This output is HIGH in the default and reset states. No pull-up resistor is required. |
| QSPI RAM Interface (mapped on port P0_yy) | | | | | |
| D2 | QSPIR_D0 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 0. Mapped on P0_00. |
| F2 | QSPIR_D1 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 1. Mapped on P0_01. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|--|--------------|--------------|------------|-------------|---|
| F1 | QSPIR_D2 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 2. Mapped on P0_02. |
| E2 | QSPIR_D3 | DIO | | | INPUT / OUTPUT. QSPI RAM I/O data 3. Mapped on P0_03. |
| E1 | QSPIR_CS | DO | | | OUTPUT. QSPI RAM clock. Mapped on P0_04. |
| D1 | QSPIR_CLK | DO | | | OUTPUT. QSPI RAM chip select (active LOW). Mapped on P0_05. |
| SPI Bus Interface (mapped on port Px_yy) | | | | | |
| | SPI_DI | DI | | | INPUT. SPI data input. (Note 2) |
| | SPI_DO | DO | | | OUTPUT. SPI data output. (Note 3) |
| | SPI_CLK | DIO | | | INPUT/OUTPUT. SPI clock. |
| | SPI_EN | DI | | | INPUT. SPI clock enable (chip select). |
| | SPI2_DI | DI | | | INPUT. SPI 2 data input. (Note 2) |
| | SPI2_DO | DO | | | OUTPUT. SPI 2 data output. (Note 3) |
| | SPI2_CLK | DIO | | | INPUT/OUTPUT. SPI 2 clock. |
| | SPI2_EN | DI | | | INPUT. SPI 2 clock enable (chip select). |
| I2C Bus Interface (mapped on port Px_yy) | | | | | |
| | I2C_SCL | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus clock with open drain port. Supports bit stretching by a slave in open drain mode. |
| | I2C_SDA | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus data with open drain port. |
| | I2C2_SCL | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus 2 clock with open drain port. Supports bit stretching by a slave in open drain mode. |
| | I2C2_SDA | DIO/ DIOD | | | INPUT/OUTPUT. I2C bus 2 data with open drain port. |
| UART Interface (mapped on port Px_yy) | | | | | |
| | UART_RX | DI | | | INPUT. UART receive data. |
| | UART_TX | DO | | | OUTPUT. UART transmit data. |
| | UART2_RX | DI | | | INPUT. UART 2 receive data. |
| | UART2_TX | DO | | | OUTPUT. UART 2 transmit data. |
| | UART2_CTS | DI | | | INPUT. UART 2 clear to send. |
| | UART2_RTS | DO | | | OUTPUT. UART 2 request to send. |
| | UART3_RX | DI | | | INPUT. UART 3 receive data. |
| | UART3_TX | DO | | | OUTPUT. UART 3 transmit data. |
| | UART3_CTS | DI | | | INPUT. UART 3 clear to send. |
| | UART3_RTS | DO | | | OUTPUT. UART 3 request to send. |
| ISO7816 Bus Interface (mapped on port Px_yy) | | | | | |
| | ISO7816_CLK | DO | | | OUTPUT. Smart card (ISO7816) clock signal |
| | ISO7816_DATA | DIO | | | INPUT / OUTPUT. Smart card (ISO7816) I/O data signal. |
| | ISO7816_RST | DO | | | OUTPUT. Smart card (ISO7816) reset signal. |
| | ISO7816_CI | DI | | | INPUT. Smart card (ISO7816) inserted signal. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|---|-------------------------|------|------------|-------------|---|
| PCM Interface (mapped on port Px_yy) | | | | | |
| | PCM_DI | DI | | | INPUT. PCM input data. |
| | PCM_DO | DO | | | OUTPUT. PCM output data. |
| | PCM_FSC | DIO | | | INPUT / OUTPUT. PCM Frame synchronization. |
| | PCM_CLK | DIO | | | INPUT / OUTPUT. PCM Clock |
| PDM Interface (mapped on port Px_yy) | | | | | |
| | PDM_DATA | DIO | | | INPUT/OUTPUT. PDM data. |
| | PDM_CLK | DO | | | OUTPUT. PDM clock output. |
| Motor Controller Interface (mapped on port P0_yy) | | | | | |
| D8 | PG0_1 | DO | | | OUTPUT. Pattern Generator 0, Output 1. |
| D7 | PG0_2 | DO | | | OUTPUT. Pattern Generator 0, Output 2. |
| D6 | PG0_3 | DO | | | OUTPUT. Pattern Generator 0, Output 3. |
| D5 | PG0_4 | DO | | | OUTPUT. Pattern Generator 0, Output 4. |
| D4 | PG1_1 | DO | | | OUTPUT. Pattern Generator 1, Output 1. |
| D3 | PG1_2 | DO | | | OUTPUT. Pattern Generator 1, Output 2. |
| C4 | PG1_3 | DO | | | OUTPUT. Pattern Generator 1, Output 3. |
| C3 | PG1_4 | DO | | | OUTPUT. Pattern Generator 1, Output 4. |
| C8 | PG2_1 | DO | | | OUTPUT. Pattern Generator 2, Output 1. |
| C7 | PG2_2 | DO | | | OUTPUT. Pattern Generator 2, Output 2. |
| C6 | PG2_3 | DO | | | OUTPUT. Pattern Generator 2, Output 3. |
| C5 | PG2_4 | DO | | | OUTPUT. Pattern Generator 2, Output 4. |
| E1 | PG3_1 | DO | | | OUTPUT. Pattern Generator 3, Output 1. |
| E2 | PG3_2 | DO | | | OUTPUT. Pattern Generator 3, Output 2. |
| D1 | PG3_3 | DO | | | OUTPUT. Pattern Generator 3, Output 3. |
| D2 | PG3_4 | DO | | | OUTPUT. Pattern Generator 3, Output 4. |
| F1 | PG4_1 | DO | | | OUTPUT. Pattern Generator 4, Output 1. |
| F2 | PG4_2 | DO | | | OUTPUT. Pattern Generator 4, Output 2. |
| B6 | PG4_3 | DO | | | OUTPUT. Pattern Generator 4, Output 3. |
| B5 | PG4_4 | DO | | | OUTPUT. Pattern Generator 4, Output 4. |
| LCD Controller Interface | | | | | |
| | LCD_SPI_DC (Note 1) | DO | | | OUTPUT. LCD data/command select. |
| | LCD_SPI_DO (Note 1) | DO | | | OUTPUT. LCD SPI data output. |
| | LCD_SPI_CLK (Note 1) | DO | | | OUTPUT. LCD SPI clock. |
| | LCD_SPI_EN (Note 1) | DO | | | OUTPUT. LCD SPI clock enable (chip select). |
| F7 | LCD_RED0 (Note 1) | DO | | | LCD Red image data output, bit 0. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|------------------|--|------|------------|-------------|---|
| F6 | LCD_RED1 (Note 1) | DO | | | LCD Red image data output, bit 1. |
| E5 | LCD_GREEN0 (Note 1) | DO | | | LCD Green image data output, bit 0. |
| E4 | LCD_GREEN1 (Note 1) | DO | | | LCD Green image data output, bit 1. |
| E7 | LCD_BLUE0 (Note 1) | DO | | | LCD Blue image data output, bit 0. |
| E6 | LCD_BLUE1 (Note 1) | DO | | | LCD Blue image data output, bit 1. |
| H6, D7 | PLCD_ENAB (Note 1) | DO | | | OUTPUT. Parallel LCD Indication signal for valid data on the LCD data bus. |
| H4, D5 | PLCD_CLK (Note 1) | DO | | | OUTPUT. Parallel LCD Clock signal. |
| H3, D4 | PLCD_HSYNC (Note 1) | DO | | | OUTPUT. Parallel LCD Indication signal for resetting the LCD column pointer to the edge of the display. |
| H5, D6 | PLCD_VSYNC (Note 1) | DO | | | OUTPUT. Parallel LCD Indication signal for resetting the LCD row pointer to top of the display. |
| H6, D7 | LCD_ENB (Note 1) | DO | | | OUTPUT. LCD Write enable signal for the pixel memory. |
| H3, D4 | LCD_HST (Note 1) | DO | | | OUTPUT. LCD Start signal for the horizontal driver. |
| H4, D5 | LCD_HCK (Note 1) | DO | | | OUTPUT. LCD Shift clock signal for the horizontal driver. |
| H5, D6 | LCD_VST (Note 1) | DO | | | OUTPUT. LCD Start signal for the vertical driver. |
| J7, D3 | LCD_XRST (Note 1) | DO | | | OUTPUT. LCD Reset signal for the horizontal and vertical driver. |
| H7, D8 | LCD_VCK (Note 1) | DO | | | OUTPUT. LCD Shift clock for the vertical driver. |
| K8, C3 | LCD_TE (Note 1) | DI | | | INPUT. LCD Tearing effect signal. Used to synchronize the CPU to the frame memory writing. |
| G7 | LCD_VCOM / LCD_FRP (Note 1) | DO | | | OUTPUT. Combined LCD signals into a single pin: - VCOM: Common electrode driving signal (62.5Hz). - FRP: Liquid crystal driving signal (62.5Hz) |
| G7 | LCD_EXTCOMIN (Note 1) | DO | | | OUTPUT. LCD COM Inversion Signal Input (1Hz). |
| G6 | LCD_XFRP (Note 1) | DO | | | OUTPUT. LCD Liquid crystal driving signal (62.5Hz inverted). |
| Analog Interface | | | | | |
| G8 | GPADC_0 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 0. |
| D9 | GPADC_1 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 1. |
| B6 | GPADC_2 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 2. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|-------------------------|------------|------|------------|-------------|---|
| B5 | GPADC_3 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 3. |
| H7 | GPADC_4 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 4. |
| H8 | GPADC_5 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 5. |
| J10 | GPADC_6 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 6. |
| J9 | GPADC_7 | AI | | | INPUT. Analog input for the general-purpose ADC, channel 7. |
| G8 | SDADC_0 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 0. |
| D9 | SDADC_1 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 1. |
| B6 | SDADC_2 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 2. |
| B5 | SDADC_3 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 3. |
| H6 | SDADC_4 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 4. |
| J8 | SDADC_5 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 5. |
| J7 | SDADC_6 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 6. |
| K8 | SDADC_7 | AI | | | INPUT. Analog input for the ΣΔ ADC, channel 7. |
| B10 | SDADC_GND | AI | | | INPUT. Analog input for the ΣΔ ADC reference ground. |
| C9 | SDADC_REF | AI | | | INPUT. Analog input for the ΣΔ ADC reference voltage. |
| USB FS Interface | | | | | |
| K6 | USBp | AIO | | | INPUT/OUTPUT. Analog USB Full Speed D+ signal. |
| K7 | USBm | AIO | | | INPUT/OUTPUT. Analog USB Full Speed D- signal. |
| Haptic Driver Interface | | | | | |
| K4 | HDRVNM | AO | | | Negative haptic driver differential output. |
| J4 | HDRVVP | AO | | | Positive haptic-driver differential output. |
| Radio Interface | | | | | |
| A8 | RFIOP | AIO | | | RF input/output. Impedance 50 Ω. |
| A7 | RFIOM | AIO | | | RF Ground |
| Miscellaneous | | | | | |
| K10 | LED1 | AO | | | OUTPUT. LED 1 driver output (open drain, 20mA maximum). |
| K9 | LED2 | AO | | | OUTPUT. LED 2 driver output (open drain, 20mA maximum). |
| B9 | NTC Input | AI | | | INPUT. Analog input for battery NTC (feedback). |
| E9 | NTC Supply | AO | | | OUTPUT. Power supply for battery NTC sensor. |
| B8 | RSTn | AI | | | INPUT. Reset signal (active LOW). |
| Power Supply | | | | | |
| J2 | V12 | AIO | | | OUTPUT. 1.2 V power rail. 10 µF decoupling capacitor required. |
| H1 | V14 | AIO | | | OUTPUT. 1.4 V power rail. 10 µF decoupling capacitor required. |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Ball No. | Pin Name | Type | Drive (mA) | Reset State | Description |
|----------|----------|----------|------------|-------------|---|
| A5 | VSUP_RF | AI | | | INPUT. Radio supply voltage. Connect to V14 externally. 10 µF decoupling capacitor required. |
| G1 | V18F | AIO | | | OUTPUT. 1.8 V power rail. Maximum current 50 mA. 0.1 µF decoupling capacitor required. Connect to V18P externally. |
| G2 | V18P | AIO | | | OUTPUT. 1.8 V power rail. Maximum current 50 mA. 22 µF decoupling capacitor required. |
| H2 | V18 | AIO | | | OUTPUT. 1.8 V power rail. Maximum current 50 mA. 22 µF decoupling capacitor required. |
| J6 | V30 | AIO | | | OUTPUT. 3.0 V power rail. Maximum current 150 mA. 4.7 µF decoupling capacitor required. |
| K1 | LX | AIO | | | INPUT/OUTPUT. Connection for the external DC-DC converter inductor. |
| J1 | LY | AIO | | | INPUT/OUTPUT. Connection for the external DC-DC converter inductor. |
| K5 | VBAT1 | AIO | | | INPUT. Battery connection 1 for LDO supply. |
| K2 | VBAT2 | AIO | | | INPUT. Battery connection 2 for DC-DC converter supply. |
| J5 | VBUS | AI AI | | | INPUT. USB bus voltage. INPUT. Battery charge voltage. |
| F5 | AVS1 | - | | | Analog Ground. |
| G5 | AVS2 | - | | | Analog Ground. |
| K3 | BUCK GND | - | | | DCDC Converter Ground. |
| G3 | VSSD1 | - | | | Digital Ground. |
| G4 | VSSD2 | - | | | Digital Ground. |
| F4 | VSSD3 | - | | | Digital Ground. |
| F3 | VSSD4 | - | | | Digital Ground. |
| E3 | VSSIOQ | - | | | QSPI Ground. |
| B7 | ESDN | - | | | RF Ground. |
| A9 | RFIOM2 | - | | | RF Ground. |
| D10 | GND_RF1 | - | | | RF Ground. |
| C10 | GND_RF2 | - | | | RF Ground. |
| A10 | PSUB_RF | - | | | RF Ground. |
| A6 | PSUB | - | | | RF Ground. |
| J3 | HDRV GND | - | | | Haptic Driver Ground |

Note 1 Not applicable to all supported LCD Interfaces

Note 2 Data input only. MOSI in SPI slave mode, MISO in SPI master mode

Note 3 Data output only. MISO in SPI slave mode, MOSI in SPI master mode

Note 4 RDS = Reduced Driving Strength functionality (PAD_WEAK_CTRL_REG).

Note 5 This pin can only be used as a GPIO in 1.8V since it is a QSPI type of pad

5 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_{BAT1} = V_{BAT2} = 3.0 V, T_A = 25 °C. All radio measurements are performed with standard RF measurement equipment providing a source/load impedance of 50 Ω.

The specified MIN and MAX capacitor values define the range of the effective capacitance, which may vary over the applied voltage due to voltage derating. Refer to the component manufacturer for the capacitor specifications.

Multi-core Bluetooth 5.2 SoC family with system PMU

5.1 Absolute Maximum Ratings

Table 4: Absolute Maximum Ratings

| Parameter | Description | Conditions | Min | Max | Unit |
|------------------------------|--|-------------------------------------|------|------|------|
| V _{PIN_LIM_DEF} | limiting voltage on a pin | default, unless otherwise specified | -0.1 | 3.6 | V |
| V _{BAT_LIM} | limiting battery supply voltage | pin V _{BAT} | 0 | 6 | V |
| V _{BUS_LIM} | limiting bus supply voltage | pin V _{BUS} | 0 | 6.5 | V |
| t _{R_SUP} | power supply rise time | | | 30 | ms |
| V _{PIN_LIM_3V0} | limiting voltage on a pin | 3.0 V I/O pins | 0 | 3.45 | V |
| V _{PIN_LIM_1V8} | limiting voltage on a pin | 1.8 V I/O pins | 0 | 1.98 | V |
| V _{ESD_HBM_BG_A100} | electrostatic discharge voltage (Human Body Model) | VFBGA100 package | | 2200 | V |
| V _{ESD_CDM_BG_A100} | electrostatic discharge voltage (Charged Device Model) | VFBGA100 package | | 500 | V |
| T _{STG} | storage temperature | | -50 | 150 | °C |

5.2 Recommended Operating Conditions

Table 5: Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------------------------|------------------------|---|-----|-----|------|------|
| V _{BAT} | battery supply voltage | pin V _{BAT1} and V _{BAT2} | 2.4 | | 4.75 | V |
| V _{BAT OTP(Prog ram)} | battery supply voltage | Voltage range for OTP programming. Required temperature for programming is between -20°C and 85°C | 2.4 | | 4.75 | V |
| V _{BAT OTP(Rea d)} | battery supply voltage | Voltage range for OTP reading | 2.4 | | 4.75 | V |
| V _{BUS} | bus supply voltage | pin V _{BUS} | 4.2 | | 5.75 | V |
| V _{PIN_3V0} | voltage on a pin | 3.0 V I/O pins | 0 | | 3 | V |
| V _{PIN_1V8} | voltage on a pin | 1.8 V I/O pins | 0 | | 1.8 | V |
| T _A | ambient temperature | | -40 | | 85 | °C |

5.3 DC Characteristics

Table 6: DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------------|---------------------------------------|--|-----|-----|-----|---------------|
| I_{BAT_IDLE} | battery supply current | CPU is idle (Wait for Interrupt - WFI); sys_clk = 32 MHz; pcik = 4MHz; DC-DC on; FLASH off; peripherals on; $V_{BAT} = 3\text{ V}$. | | 0.8 | | mA |
| $I_{BAT_RUN_32M\text{ Hz}}$ | average active battery supply current | CPU is executing code from RAM; sys_clk = 32 MHz; pcik=4 MHz; DC-DC on; Peripherals off; $V_{BAT} = 3\text{ V}$. | | 1.8 | | mA |
| $I_{BAT_RUN_48M\text{ Hz}}$ | average active battery supply current | CPU is executing code from RAM; hclk = 48 MHz; PLL on; pcik=6 MHz; DC-DC on; Peripherals off; $V_{BAT} = 3\text{ V}$. | | 5 | | mA |
| $I_{BAT_RUN_96M\text{ Hz}}$ | average active battery supply current | CPU is executing code from RAM; hclk = 96 MHz; PLL on; pcik=12 MHz; DC-DC on; Peripherals off; $V_{BAT} = 3\text{ V}$. | | 7.8 | | mA |
| I_{BAT_HIBERN} | battery supply current | Hibernation mode; no RAM retained; all clocks off; DC-DC off; V_{DD_clamp} setting 0xD at 25°C FLASH off; $V_{BAT} = 3\text{ V}$. Note 1 | | 2 | | μA |
| $I_{BAT_DP_SLP}$ | battery supply current | Deep Sleep mode; No RAM retained; XTAL32K on; RTC off; DC-DC off; $V_{DD_RET} = 0.75\text{ V}$; FLASH off; $V_{BAT} = 3\text{ V}$. Bandgap Refresh rate=0x80 | | 3 | | μA |
| $I_{BAT_EX_SLP_16K_64K}$ | battery supply current | Extended Sleep mode; 16 kB (Cache) and 64 kB (data) RAM retained; XTAL32K on; DC-DC off; $V_{DD_RET} = 0.75\text{ V}$; FLASH in Power Down mode; $V_{BAT} = 3\text{ V}$. Bandgap Refresh rate = 0x80; Note 2 | | 4.2 | | μA |
| $I_{BAT_EX_SLP_16K_128K}$ | battery supply current | Extended Sleep mode; 16 kB (cache) and 128 kB (data) RAM retained; XTAL32K on; DC-DC off; $V_{DD_RET} = 0.75\text{ V}$; FLASH in Power Down mode; $V_{BAT} = 3\text{ V}$. Bandgap Refresh rate = 0x80; Note 2 | | 4.7 | | μA |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------------------------------|--------------------------------|--|-----|-----|-----|------|
| I _{BAT_EX_SLP_1} 6K_256K | battery supply current | Extended Sleep mode; 16 kB (cache) and 256 kB (data) RAM retained; XTAL32K on; DC-DC off; V _{DD_RET} = 0.75 V; FLASH in Power Down mode; V _{BAT} = 3 V. Bandgap Refresh rate = 0x80; Note 2 | | 5.4 | | µA |
| I _{BAT_EX_SLP_1} 6K_384K | battery supply current | Extended Sleep mode; 16 kB (cache) and 384 kB (data) RAM retained; XTAL32K on; DC-DC off; V _{DD_RET} = 0.75 V; FLASH in Power Down mode; V _{BAT} = 3 V. Bandgap Refresh rate = 0x80; Note 2 | | 6.3 | | µA |
| I _{BAT_EX_SLP_1} 6K_512K | battery supply current | Extended Sleep mode; 16 kB (cache) and 512 kB (data) RAM retained; XTAL32K on; DC-DC off; V _{DD_RET} = 0.75 V; FLASH in Power Down mode; V _{BAT} = 3 V. Bandgap Refresh rate = 0x80; Note 2 | | 7.7 | | µA |
| I _{BAT_GearBox} | average battery supply current | Gear Box control mode for moving seconds (every 330ms) and minutes (every 20 sec) hands; RTC on; PD_PER on; RTC will wake up SNC.SNC will trigger Motor Controller to deliver 9 ms waveforms on GPIOs (at 1.8V) and go back to sleep. ARM M33 kept off constantly. | | 41 | | µA |
| I _{BAT_SensorNode} | average battery supply current | Sensor Node Controller reading from SPI at 4MHz, 8-bit at 1.8V; Running at RC32M; XTAL32M=off; Hclk=8MHz; Pclk=2MHz; DC-DC on; CPU off; FLASH in Power Down mode; V _{BAT} = 3 V. | | 1.1 | | mA |
| I _{BAT_BLE_RX_3} 2M | peak battery supply current | BLE receive mode; f _{CLK} = 32 MHz; CPU idle; DC-DC on; FLASH off; V _{BAT} = 3 V. | | 4.5 | | mA |
| I _{BAT_BLE_TX_3} 2M | peak battery supply current | BLE transmit mode @ 0dbm; f _{CLK} = 32 MHz; CPU idle; DC-DC on; FLASH off; V _{BAT} = 3 V. | | 5.6 | | mA |
| I _{BAT_BLE_RX_9} 6M | peak battery supply current | BLE receive mode; f _{CLK} = 96 MHz; CPU idle; DC-DC on; FLASH off; V _{BAT} = 3 V. | | 6.5 | | mA |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|-----------------------------|---|-----|-----|-----|------|
| I _{BAT_BLE_TX_96M} | peak battery supply current | BLE transmit mode @ 0dbm; f _{CLK} = 96 MHz; CPU idle; DC-DC on; FLASH off; V _{BAT} = 3 V. | | 7.6 | | mA |

Note 1 At temperatures higher than 55 °C, a higher voltage setting (0xC) should be applied at the clamp to ensure proper hibernation operation

Note 2 If RTC is on then 150 nA should be added

5.4 Timing Characteristics

Table 7: Timing Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|---|--|-----|-----|-----|-------|
| t _{STA_SLOW} | supply startup time | time from wake up GPIO toggle up to application SW execution start. All voltage levels are checked prior enabling the CPU | | | 600 | μs |
| t _{STA_FAST} | supply startup time | time from wake up GPIO toggle up to application SW execution start. Core (VDD) and Radio (V14) voltages checked prior enabling the CPU | | | 300 | μs |
| t _{STA_ULTRA_FAST} | supply startup time | time from wake up GPIO toggle up to application SW execution start. All voltage levels are assumed ok prior enabling the CPU | | | 120 | μs |
| t _{STA_BOOT} | Power up to booter executed, startup time | BootROM code execution time without authentication. Max. value when in "development mode", min. value when in "Normal" mode. | 8 | | 50 | ms |
| t _{CLF OTP} | cache line fetch time | from OTP; line size = 8 B | | | 6 | clock |
| t _{CLF FLA} | cache line fetch time | from FLASH; line size = 8 B | | | 40 | clock |

5.5 Thermal Characteristics

Table 8: Thermal Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------|--|---|-----|------|-----|------|
| R _{θJA_P1_HQ} | Thermal resistance (junction to ambient θ _{dJA}) | VFBGA86 package. JEDEC PCB, 0.1W+0.3W Power map | | 48.8 | | °C/W |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------|--|---|-----|------|-----|------|
| R _{θJA_P2_HR} | Thermal resistance (junction to ambient $\theta_{\text{dJA}} \text{n}$) | VFBGA100 package.JEDEC PCB, 0.1W+0.3W Power map | | 48.8 | | °C/W |

5.6 Reset Characteristics

Table 9: PAD_RESET - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|--------------------------------------|-----------------------------------|--------------------------|-----|--------------------------|------|
| I _{IL_PU_RSTN} | LOW level input current with pull-up | Pin RESETn, VI=0.0V | -75 | | -25 | μA |
| V _{IH_RSTN} | HIGH level input voltage | Pin RESETn, V ₁₂ =1.2V | 0.7*V ₁₂ 2 | | | V |
| V _{IL_RSTN} | LOW level input voltage | Pin RESETn, V ₁₂ =1.2V | | | 0.3*V ₁₂ 2 | V |

5.7 Bandgap Characteristics

Table 10: BG_REF - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------------|---|----------------|------|-----|------|------|
| V _{BG_REF} | bandgap reference voltage from GPADC reference buffer | | 1.14 | 1.2 | 1.26 | V |
| V _{ACC_TRIM} | voltage accuracy | after trimming | | | 1 | % |
| I _{BG_IREF_CUR_TRIM} | internal bandgap reference trim test current | | 13 | 15 | 16 | μA |

5.8 Brown-Out Detector Characteristics

Table 11: BOD - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------------|---------------|------------------------|------|------|------|------|
| V _{RST_VDD_sleep} | reset voltage | BOD_LVL_VDD_RET = 0x6F | 0.67 | 0.7 | 0.73 | V |
| V _{RST_VDD} | reset voltage | BOD_LVL_VDD_ON = 0xA7 | 1.02 | 1.05 | 1.08 | V |
| V _{RST_V14} | reset voltage | BOD_LVL_V14 = 0xC7 | 1.21 | 1.25 | 1.29 | V |

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------|---------------|----------------------|------|------|------|------|
| V _{RST_V18P} | reset voltage | BOD_LVL_V18P = 0x107 | 1.61 | 1.65 | 1.69 | V |
| V _{RST_V18F} | reset voltage | BOD_LVL_V18F = 0x107 | 1.61 | 1.65 | 1.69 | V |
| V _{RST_V18} | reset voltage | BOD_LVL_V18 = 0x107 | 1.61 | 1.65 | 1.69 | V |
| V _{RST_V30} | reset voltage | BOD_LVL_V30 = 0x137 | 1.91 | 1.95 | 1.99 | V |
| V _{RST_VBAT} | reset voltage | BOD_LVL_VBAT = 0xF4 | 2.24 | 2.3 | 2.36 | V |

5.9 General Purpose ADC Characteristics

Table 12: GP_ADC - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|-----------------------------|------------|-----|-----|-----|------|
| N _{BIT_ADC} | number of bits (resolution) | | | 10 | | bit |

Table 13: GP_ADC - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|-----------------------------------|--|------|------|-----|------|
| E _G | gain error without calibration | Trimmed bandgap | -3 | | 3 | % |
| E _{G_CALIBRATE_D} | gain error after calibration | Trimmed bandgap & Gain Error + Offset correction applied | -0.8 | | 0.8 | % |
| E _{ofs} | offset error without calibration | Trimmed bandgap LSB wrt 10bit accuracy | -20 | | 20 | LSB |
| E _{ofs_CALIBRATED} | offset error after calibration | Trimmed bandgap & Gain Error + Offset correction applied LSB wrt 10bit accuracy | -4 | | 4 | LSB |
| E _{G_ATTN3x} | gain error of the 3x attenuator | Trimmed bandgap & GPADC Gain Error + Offset correction applied | -4 | | 1 | % |
| E _{G_VBAT} | gain error of the VBAT attenuator | Trimmed bandgap & GPADC Gain Error + Offset correction applied | | -1.7 | | % |
| INL | integral non-linearity | LSB wrt 10bit accuracy | -2 | | 2 | LSB |
| DNL | differential non-linearity | LSB wrt 10bit accuracy | -2 | | 2 | LSB |
| ENOB | Effective Number Of Bits | no averaging, no chopping, Single-Ended: V _{IN,PP} = 1.1V | | 9 | | bit |
| ENOB _{AVG128} | Effective Number Of Bits | 128x averaging, Single-Ended: V _{IN,PP} = 1.1V | | 10.5 | | bit |

Table 14: GP_ADC - Timing Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------|-----------------|---|-----|-----|-----|------|
| t _{CONV_ADC} | conversion time | Single conversion (CONV_NRS=0, CHOP=0), maximum sampling time (SMPL_TIME=15), maximum store delay (STORE_DEL=15), and including initial LDO settling (20us) | | 55 | | μs |

5.10 ΣΔ ADC Characteristics

Table 15: SD_ADC - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------------------|--|---|-------|-----|------|------|
| C _{IN} | input capacitance | | | 192 | | fF |
| INL_ext_ref | integral non-linearity Note 1 | LSB wrt 14b accuracy | -4 | | 4 | LSB |
| DNL_ext_ref | differential non-linearity | LSB wrt 14b accuracy | -4 | | 4 | LSB |
| E _{G_ext_ref} | gain error | Using external voltage reference 1.2V | -0.25 | | 0.25 | % |
| E _{OFS_ext_ref} | offset error | LSB wrt 14b accuracy. Using external voltage reference 1.2V | -2 | | 2 | LSB |

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

Table 16: SD_ADC - Electrical performance

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------|-----------------------------|------------|-----|-----|-----|------|
| SFDR_ext_ref | Spurious-free dynamic range | | 74 | | | dB |
| SNR_ext_ref | Signal to noise ratio | | 74 | | | dB |

5.11 DC-DC Converter Characteristics

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 17: DCDC - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---------------------|--------------|------------|-----|-----|-----|------|
| I _{L_1V8} | Load Current | | | | 50 | mA |
| I _{L_1V8P} | Load Current | | | | 50 | mA |

Table 18: DCDC - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---------------------------------|-------------------------------------|---|-------|------|-------|------|
| I _Q | Quiescent Current | No load | | 180 | | µA |
| V _{O_1V2_0} | VDD1V2 Output Voltage | VDD_LEVEL = 0 | 0.875 | 0.9 | 0.925 | V |
| V _{O_1V2_1} | VDD1V2 Output Voltage | VDD_LEVEL = 1 | 0.975 | 1 | 1.025 | V |
| V _{O_1V2_2} | VDD1V2 Output Voltage | VDD_LEVEL = 2 | 1.075 | 1.1 | 1.125 | V |
| V _{O_1V2_3} | VDD1V2 Output Voltage | VDD_LEVEL = 3 | 1.175 | 1.2 | 1.225 | V |
| V _{RPL_1V2_TYP} | VDD1V2 Peak-to-Peak Ripple Voltage | V _{BAT} = 3.0 V, I _{VDD1V2} = 5 mA, I _{VDD1V4} = 5 mA, I _{VDD1V8P} = 2 mA, I _{VDD1V8} = 0 mA | | 10 | 50 | mV |
| V _{O_1V4_0} | VDD1V4 Output Voltage | V14_LEVEL = 0 | 1.15 | 1.2 | 1.25 | V |
| V _{O_1V4_1} | VDD1V4 Output Voltage | V14_LEVEL = 1 | 1.2 | 1.25 | 1.3 | V |
| V _{O_1V4_2} | VDD1V4 Output Voltage | V14_LEVEL = 2 | 1.25 | 1.3 | 1.35 | V |
| V _{O_1V4_3} | VDD1V4 Output Voltage | V14_LEVEL = 3 | 1.3 | 1.35 | 1.4 | V |
| V _{O_1V4_4} | VDD1V4 Output voltage | V14_LEVEL = 4 | 1.35 | 1.4 | 1.45 | V |
| V _{O_1V4_5} | VDD1V4 Output Voltage | V14_LEVEL = 5 | 1.4 | 1.45 | 1.5 | V |
| V _{O_1V4_6} | VDD1V4 Output Voltage | V14_LEVEL = 6 | 1.45 | 1.5 | 1.55 | V |
| V _{O_1V4_7} | VDD1V4 Output Voltage | V14_LEVEL = 7 | 1.5 | 1.55 | 1.6 | V |
| V _{RPL_1V4_TYP} | VDD1V4 Peak-to-Peak Ripple Voltage | V _{BAT} = 3.0 V, I _{VDD1V2} = 5 mA, I _{VDD1V4} = 5 mA, I _{VDD1V8P} = 2 mA, I _{VDD1V8} = 0 mA | | 5 | 20 | mV |
| V _{O_1V8_0} | VDD1V8 Output Voltage | V18_LEVEL = 0 | 1.1 | 1.2 | 1.25 | V |
| V _{O_1V8_1} | VDD1V8 Output Voltage | V18 LEVEL = 1 | 1.7 | 1.8 | 1.9 | V |
| V _{RPL_1V8_TYP} | VDD1V8P Peak-to-Peak Ripple Voltage | V _{BAT} = 3.0 V, I _{VDD1V2} = 5 mA, I _{VDD1V4} = 5 mA, I _{VDD1V8P} = 2 mA, I _{VDD1V8} = 0 mA | | 5 | 50 | mV |
| V _{RPL_1V8_EXT_LOAD_1} | VDD1V8P Peak-to-Peak Ripple Voltage | V _{BAT} = 3.0 V, I _{VDD1V2} = 5 mA, I _{VDD1V4} = 5 mA, I _{VDD1V8P} = 5 mA, I _{VDD1V8} = 5 mA | | 10 | 50 | mV |
| V _{RPL_1V8_EXT_LOAD_2} | VDD1V8P Peak-to-Peak Ripple Voltage | V _{BAT} = 3.0 V, I _{VDD1V2} = 5 mA, I _{VDD1V4} = 5 mA, I _{VDD1V8P} = 15 mA, I _{VDD1V8} = 15 mA | | 15 | 50 | mV |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------------|-------------------------------------|--|-----|-----|-----|------|
| $V_{RPL_1V8_EXT_LOAD_3}$ | VDD1V8P Peak-to-Peak Ripple Voltage | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 50 \text{ mA}$, $I_{VDD1V8} = 50 \text{ mA}$ | | 35 | 50 | mV |
| V_{O_1V8P} | VDD1V8P Output Voltage | | 1.7 | 1.8 | 1.9 | V |
| $V_{RPL_1V8P_TY_P}$ | VDD1V8 Peak-to-Peak Ripple Voltage | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 2 \text{ mA}$, $I_{VDD1V8} = 0 \text{ mA}$ | | 5 | 50 | mV |
| $V_{RPL_1V8P_EXT_LOAD_1}$ | VDD1V8 Peak-to-Peak Ripple Voltage | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 5 \text{ mA}$, $I_{VDD1V8} = 5 \text{ mA}$ | | 10 | 50 | mV |
| $V_{RPL_1V8P_EXT_LOAD_2}$ | VDD1V8 Peak-to-Peak Ripple Voltage | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 15 \text{ mA}$, $I_{VDD1V8} = 15 \text{ mA}$ | | 15 | 50 | mV |
| $V_{RPL_1V8P_EXT_LOAD_3}$ | VDD1V8 Peak-to-Peak Ripple Voltage | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 50 \text{ mA}$, $I_{VDD1V8} = 50 \text{ mA}$ | | 35 | 50 | mV |
| $\eta_{CONV_1V8_TY_P}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 2 \text{ mA}$, $I_{VDD1V8} = 0 \text{ mA}$ | | 80 | | % |
| $\eta_{CONV_1V8_EXT_LOAD_1}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 5 \text{ mA}$, $I_{VDD1V8} = 5 \text{ mA}$ | | 80 | | % |
| $\eta_{CONV_1V8_EXT_LOAD_2}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 15 \text{ mA}$, $I_{VDD1V8} = 15 \text{ mA}$ | | 85 | | % |
| $\eta_{CONV_1V8_EXT_LOAD_3}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 50 \text{ mA}$, $I_{VDD1V8} = 50 \text{ mA}$ | | 80 | | % |
| $\eta_{CONV_1V8P_T_YP}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 2 \text{ mA}$, $I_{VDD1V8} = 0 \text{ mA}$ | | 80 | | % |
| $\eta_{CONV_1V8P_EXT_LOAD_1}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 5 \text{ mA}$, $I_{VDD1V8} = 5 \text{ mA}$ | | 80 | | % |
| $\eta_{CONV_1V8P_EXT_LOAD_2}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 15 \text{ mA}$, $I_{VDD1V8} = 15 \text{ mA}$ | | 85 | | % |
| $\eta_{CONV_1V8P_EXT_LOAD_3}$ | Conversion Efficiency | $V_{BAT} = 3.0 \text{ V}$, $I_{VDD1V2} = 5 \text{ mA}$, $I_{VDD1V4} = 5 \text{ mA}$, $I_{VDD1V8P} = 50 \text{ mA}$, $I_{VDD1V8} = 50 \text{ mA}$ | | 80 | | % |

5.12 LDOs Characteristics

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 19: LDO_1v8 - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|------------------------------|---|-----|-----|-----|------------------|
| $I_{L_LDO_1v8}$ | load current | $(V_{IN} - V_{OUT}) > 400 \text{ mV}; I_L \text{ MAX} = 45 \text{ mA for } 200 \text{ mV} < (V_{IN} - V_{OUT}) < 400 \text{ mV.}$ | 0 | | 50 | mA |
| $C_{L_LDO_1v8}$ | effective load capacitance | | 1 | | 100 | μF |
| $ESR_{CL_LDO_1v8}$ | equivalent series resistance | | 0 | | 100 | $\text{m}\Omega$ |

Table 20: LDO_1v8 - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|--|---|-------|-----|------|------|
| $V_{LDO_1v8_0}$ | LDO output voltage | | 1.14 | 1.2 | 1.25 | V |
| $V_{LDO_1v8_1}$ | LDO output voltage | | 1.7 | 1.8 | 1.9 | V |
| REGLOAD | load regulation $((\Delta V_o/V_o) * 100\%) / \Delta I_L$ | $10 \text{ mA} < I_{LOAD} < 50 \text{ mA}$ when $(V_{IN} - V_{OUT}) > 400 \text{ mV}.$ $10 \text{ mA} < I_{LOAD} < 45 \text{ mA}$ when $200 \text{ mV} < (V_{IN} - V_{OUT}) < 400 \text{ mV}.$ | -0.03 | | 0.03 | %/mA |
| REGLINE | line regulation $((\Delta V_o/V_o) * 100\%) / \Delta V_i$ | $V_{IN} - V_{OUT} > 400 \text{ mV}$ | -0.5 | | 0.5 | %/V |
| $V_{DROP_1v8_MAX}$ | maximum dropout voltage | $I_L = 45 \text{ mA}; V_{DROP} > 400 \text{ mV}$ for $45 \text{ mA} < I_L < 50 \text{ mA}$ | | | 200 | mV |

Table 21: LDO_1v8 - AC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|------------------------------|--|-----|-----|-----|------|
| PSRR _{LDO_1v8} | power supply rejection ratio | $f \leq 200 \text{ kHz}.$ $5 \text{ mA} < I_{LOAD} < 50 \text{ mA}$ when $(V_{IN} - V_{OUT}) > 400 \text{ mV}.$ $5 \text{ mA} < I_{LOAD} < 45 \text{ mA}$ when $200 \text{ mV} < (V_{IN} - V_{OUT}) < 400 \text{ mV}.$ | 40 | | | dB |

Table 22: LDO_1v8_RET - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------|----------------------------|--|-----|-----|-----|---------------|
| $C_{L_LDO_1V8_RET}$ | effective load capacitance | | 2.2 | | 100 | μF |
| $I_{L_LDO_1V8_RET}$ | load current | $V30$ must be $> 2.2\text{V}$ for maximum output current | | | 10 | mA |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 23: LDO_1v8_RET - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------|---|--|-------|-----|------|------|
| $V_{LDO_1V8_RET_0}$ | LDO output voltage V_{OUT} nom = 1.2V | $V_{30} = 3.1V$ $I_{LOAD} = 1\text{ mA}$ | 1.15 | 1.2 | 1.25 | V |
| $V_{LDO_1V8_RET_1}$ | LDO output voltage V_{OUT} nom = 1.8V | $V_{30} = 3.1V$ $I_{LOAD} = 1\text{ mA}$ | 1.7 | 1.8 | 1.9 | V |
| REG_{LOAD} | load regulation $((\Delta V_o/V_o)^{*100\%})/\Delta I_L$ | $1\text{ mA} < I_{LOAD} < 10\text{ mA}$ $V_i \geq 2.2V$ | -0.12 | | 0.12 | %/mA |
| REG_{LINE} | line regulation $((\Delta V_o/V_o)^{*100\%})/\Delta V_i$ | $V_i \geq (V_o + 400\text{ mV})$ $V_{OUT} = 1.8\text{ or }1.2\text{ V}$, $I_{LOAD} = 10\text{ mA}$ $\Delta V_i = 0.8\text{ V}$ | -1.8 | | 1.8 | %/V |

Table 24: LDO_1V8P - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------|------------------------------|---|-----|-----|-----|-----------|
| $I_{L_LDO_1V8P}$ | load current | $(V_{IN} - V_{OUT}) > 400\text{ mV};$ $I_{L_MAX} = 45\text{ mA}$ for $200\text{ mV} < (V_{IN} - V_{OUT}) < 400\text{ mV}$. | 0 | | 50 | mA |
| $C_{L_LDO_1V8P}$ | effective load capacitance | | 1 | | 100 | μF |
| $ESR_{CL_LDO_1V8P}$ | equivalent series resistance | | 0 | | 100 | $m\Omega$ |

Table 25: LDO_1V8P - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------|---|--|-------|-----|------|------|
| V_{LDO_1V8P} | LDO output voltage | $V_{30} = 3.0\text{ V}$ V_{18P} = unloaded | 1.7 | 1.8 | 1.9 | V |
| REG_{LOAD} | load regulation $((\Delta V_o/V_o)^{*100\%})/\Delta I_L$ | $10\text{ mA} < I_{LOAD} < 50\text{ mA}$ when $(V_{IN} - V_{OUT}) > 400\text{ mV}.$ $10\text{ mA} < I_{LOAD} < 45\text{ mA}$ when $200\text{ mV} < (V_{IN} - V_{OUT}) < 400\text{ mV}.$ | -0.03 | | 0.03 | %/mA |
| REG_{LINE} | line regulation $((\Delta V_o/V_o)^{*100\%})/\Delta V_i$ | $(V_{IN} - V_{OUT}) > 400\text{ mV}$ | -0.5 | | 0.5 | %/V |
| $V_{DROP_1V8P_MAX}$ | maximum dropout voltage | $I_L = 45\text{ mA};$ $V_{DROP} > 400\text{ mV}$ for $45\text{ mA} < I_L < 50\text{ mA}$ | | | 200 | mV |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 26: LDO_1V8P - AC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------------------|------------------------------|---|-----|-----|-----|------|
| PSRR _{LDO_1V8P} | power supply rejection ratio | $f \leq 200 \text{ kHz}$. $5 \text{ mA} < I_{LOAD} < 50 \text{ mA}$ when $(V_{IN} - V_{OUT}) > 400 \text{ mV}$. $5 \text{ mA} < I_{LOAD} < 45 \text{ mA}$ when $200 \text{ mV} < (V_{IN}-V_{OUT}) < 400 \text{ mV}$. | 40 | | | dB |

Table 27: LDO_1V8P_RET - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------------|----------------------------|---|-----|-----|-----|------|
| C _L _LDO_1V8P_RET | effective load capacitance | | 2.2 | | 100 | μF |
| I _L _LDO_1V8P_RET | load current | V ₃₀ must be > 2.4V for maximum load current | | | 10 | mA |

Table 28: LDO_1V8P_RET - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---------------------------|---|--|-------|-----|------|------|
| V _{LDO_1V8P_RET} | LDO output voltage | $I_{LOAD} = 1 \text{ mA}$ $V_i \geq (V_o + 400 \text{ mV})$ | 1.7 | 1.8 | 1.9 | V |
| REG _{LOAD} | load regulation | $1 \text{ mA} < I_{LOAD} < 10 \text{ mA}$ $V_i \geq 2.2\text{V}$ | -0.12 | | 0.12 | %/mA |
| REG _{LINE} | line regulation $((\Delta V_o/V_o)^{*100\%})/\Delta V_i$ | $V_i \geq (V_o + 400 \text{ mV})$ $V_{OUT} = 1.8 \text{ V}, I_{LOAD} = 10 \text{ mA}$ | -1.8 | | 1.8 | %/V |

Table 29: LDO_CORE - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------------|------------------------------|------------|-----|-----|-----|------|
| I _L _LDO_CORE | load current | | 0 | | 50 | mA |
| C _L _LDO_CORE | load capacitance | | 1 | | 100 | μF |
| ESR _{CL_LDO_CORE} | equivalent series resistance | | 0 | | 100 | mΩ |

Table 30: LDO_CORE - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|--------------------|---|------|-----|------|------|
| V _{LDO_CORE_0} | LDO output voltage | $V_{30} = 3.0 \text{ V}$ $I_{LOAD} = 1 \text{ mA}$ | 0.85 | 0.9 | 0.95 | V |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------------|--|--|-------|-----|------|------|
| V _{LDO_CORE_1} | LDO output voltage | V ₃₀ = 3.0 V I _{LOAD} = 1 mA | 0.95 | 1 | 1.05 | V |
| V _{LDO_CORE_2} | LDO output voltage | V ₃₀ = 3.0 V I _{LOAD} = 1 mA | 1.04 | 1.1 | 1.16 | V |
| V _{LDO_CORE_3} | LDO output voltage | V ₃₀ = 3.0 V I _{LOAD} = 1 mA | 1.14 | 1.2 | 1.26 | V |
| REG _{LOAD} | load regulation (((ΔV _O /V _O) [*] 100%)/ΔI _L) | V ₃₀ = 3.0V 1 mA ≤ I _{LOAD} ≤ 50 mA | -0.06 | | 0.06 | %/mA |
| REG _{LINE} | line regulation (((ΔV _O /V _O) [*] 100%)/ΔV _I | 1.75V < V ₃₀ < 3V 1 mA ≤ I _{LOAD} ≤ 50 mA | -0.5 | | 0.5 | %/V |
| V _{DROP_LDO_CORE} | Dropout voltage (headroom) | I _L = 50 mA | | | 450 | mV |

Table 31: LDO_CORE_RET - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|----------------------------|------------|-----|-----|-----|------|
| C _{L_CORE_RET} | effective load capacitance | | 1 | | 100 | μF |

Table 32: LDO_CORE_RET - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---------------------------------|--|---|------|------|------|------|
| REG _{LINE} | line regulation (((ΔV _O /V _O) [*] 100%)/ΔV _I | I _{LOAD} = 3 mA 1.75 V < V _{IN} < 3.45 V V _{OUT} = 0.75 V or 1.0 V | -0.5 | | 0.5 | %/V |
| REG _{LOAD} | load regulation (((ΔV _O /V _O) [*] 100%)/ΔI _L) | V ₃₀ = 3.45V 1 mA < I _{LOAD} < 3 mA V _{OUT} is 0.75V or 1V | -0.1 | | 0.1 | %/mA |
| V _{LDO_CORE_R ET_0v75} | LDO output voltage Vout = 0.75 V | I _{LOAD} = 1 mA Vin = 3.45 V | 0.7 | 0.75 | 0.8 | V |
| V _{LDO_CORE_R ET_0v8} | LDO output voltage | I _{LOAD} = 1 mA Vin = 3.45 V | 0.75 | 0.8 | 0.85 | V |
| V _{LDO_CORE_R ET_0v85} | LDO output voltage | I _{LOAD} = 1 mA Vin = 3.45 V | 0.8 | 0.85 | 0.9 | V |
| V _{LDO_CORE_R ET_0v9} | LDO output voltage Vout = 0.9 V | I _{LOAD} = 1 mA Vin = 3.45 V | 0.84 | 0.9 | 0.96 | V |
| V _{LDO_CORE_R ET_0v95} | LDO output voltage | I _{LOAD} = 1 mA Vin = 3.45 V | 0.89 | 0.95 | 1.01 | V |
| V _{LDO_CORE_R ET_1v0} | LDO output voltage | I _{LOAD} = 1 mA Vin = 3.45 V | 0.94 | 1 | 1.06 | V |

Table 33: LDO_RADIO - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|------------------------------|------------|-----|-----|-----|------|
| I _{L_LDO_RADIO} | load current | | 0 | | 20 | mA |
| C _{L_LDO_RADIO} | load capacitance | | 1 | | 100 | μF |
| ESR _{CL_LDO_RADIO} | equivalent series resistance | | 0 | | 100 | mΩ |

Table 34: LDO_RADIO - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|---|---|-------|------|------|------|
| V _{LDO_V14_0} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.14 | 1.2 | 1.26 | V |
| V _{LDO_V14_1} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.18 | 1.25 | 1.32 | V |
| V _{LDO_V14_2} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.23 | 1.3 | 1.37 | V |
| V _{LDO_V14_3} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.28 | 1.35 | 1.42 | V |
| V _{LDO_V14_4} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.33 | 1.4 | 1.47 | V |
| V _{LDO_V14_5} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.38 | 1.45 | 1.53 | V |
| V _{LDO_V14_6} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.42 | 1.5 | 1.58 | V |
| V _{LDO_V14_7} | LDO output voltage | V ₃₀ = 3.0V I _{LOAD} = 1 mA | 1.47 | 1.55 | 1.63 | V |
| REG _{LOAD} | load regulation (((ΔV _O /V _O)*100%)/ΔI _L) | V ₃₀ = 3.0V 1 mA ≤ I _{LOAD} ≤ 20 mA | -0.06 | | 0.06 | %/mA |
| REG _{LINE} | line regulation (((ΔV _O /V _O)*100%)/ΔV _I | 1 mA < I _{LOAD} < 20 mA. 1.75V < V ₃₀ < 3.0V | -0.5 | | 0.5 | %/V |
| V _{DROP_LDO_RADIO} | Dropout voltage (headroom) | I _L = 20 mA | | | 200 | mV |

Multi-core Bluetooth 5.2 SoC family with system PMU
Table 35: LDO_RADIO - AC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------|------------------------------|--|-----|-----|-----|------|
| PSRR_LDO_RADIO | power supply rejection ratio | 2 mA ≤ I _L ≤ 20 mA; f ≤ 200 kHz | 40 | | | dB |

Table 36: LDO_USB - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------------|------------------------------|--|-----|-----|-----|------|
| I _L _LDO_USB | load current | | | | 150 | mA |
| C _L _LDO_USB | load capacitance | Shared output pin with retention LDO (which needs a minimum cap of 2.2 uF) | 2.2 | | 100 | μF |
| ESR _{CL} _LDO_USB | equivalent series resistance | | 0 | | 100 | mΩ |

Table 37: LDO_USB - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------|---|---|-------|------|------|------|
| V _{LDO_USB_0} | LDO output voltage | | 2.81 | 3 | 3.15 | V |
| V _{LDO_USB_1} | LDO output voltage | | 3.2 | 3.45 | 3.6 | V |
| V _{LDO_USB_2} | LDO output voltage | | 3.05 | 3.3 | 3.47 | V |
| REG _{LOAD} | load regulation (((ΔV _O /V _O)*100%)/ΔI _L) | 10 mA < I _{LOAD} < 150 mA | -0.03 | | 0.03 | %/mA |
| REG _{LINE} | line regulation (((ΔV _O /V _O)*100%)/ΔV _I) | I _{LOAD} = 150 mA V _{IN} > V _{OUT} + 800 mV | -2 | | 2 | %/V |

Table 38: LDO_VBAT - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|------------------------------|--|-----|-----|-----|------|
| I _L _LDO_VBAT | load current | | | | 150 | mA |
| C _L _LDO_VBAT | effective load capacitance | Shared output pin with retention LDO (which needs a minimum cap of 2.2 uF) | 1 | | 100 | μF |
| ESR _{CL} _LDO_VBAT | equivalent series resistance | | 0 | | 100 | mΩ |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 39: LDO_VBAT - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------------|--|---|-------|------|------|------|
| I _{Q_LDO_VBAT} | quiescent current | No Load | | 65 | | µA |
| V _{LDO_VBAT_0} | output voltage | V _{BAT} = 4.2V V30 = Unloaded | 2.8 | 3 | 3.15 | V |
| V _{LDO_VBAT_1} | output voltage | V _{BAT} = 4.2V V30 = Unloaded | 3.2 | 3.45 | 3.6 | V |
| V _{LDO_VBAT_2} | output voltage | V _{BAT} = 4.2V V30 = Unloaded | 3.05 | 3.3 | 3.47 | V |
| REG _{LOAD} | load regulation (((ΔV _O /V _O) [*] 100%)/ΔI _L) | 10 mA < I _{LOAD} < 150 mA | -0.03 | | 0.03 | %/mA |
| REG _{LINE} | line regulation (((ΔV _O /V _O) [*] 100%)/ΔV _I) | I _{LOAD} 150 mA V _i ≥ (V _O + 400 mV) | -0.5 | | 0.5 | %/V |
| V _{DROP_VBAT_MAX} | maximum dropout voltage | I _L = 150 mA | | | 400 | mV |

Table 40: LDO_VBAT_RET - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------------|----------------------------|------------|-----|-----|-----|------|
| I _{L_LDO_VBAT_R ET} | load current | | | | 10 | mA |
| C _{L_LDO_VBAT} | effective load capacitance | | 2.2 | | 100 | µF |

Table 41: LDO_VBAT_RET - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------------|--|--|-------|------|------|------|
| V _{LDO_VBAT_R ET_0} | output voltage | V _{IN} 4.2V; V _{OUT} = Unloaded | 2.85 | 3 | 3.15 | V |
| V _{LDO_VBAT_R ET_1} | output voltage | V _{IN} = 4.2V; V _{OUT} = Unloaded | 3.25 | 3.45 | 3.6 | V |
| V _{LDO_VBAT_R ET_2} | output voltage | V _{IN} = 4.2V; V _{OUT} = Unloaded | 3.02 | 3.3 | 3.56 | V |
| I _{Q_LDO_VBAT_RET} | quiescent current | Unloaded | | 80 | | nA |
| REG _{LOAD} | load regulation (((ΔV _O /V _O) [*] 100%)/ΔI _L) | 1 mA < I _{LOAD} < 10 mA | -0.07 | | 0.07 | %/mA |
| REG _{LINE} | line regulation (((ΔV _O /V _O) [*] 100%)/ΔV _I) | V _{IN} - V _{OUT} > 450 mV I _{LOAD} = (I _{_int} +3 mA) | -0.5 | | 0.5 | %/V |
| V _{DROP} | maximum dropout voltage | I _{LOAD} = 10 mA | | | 450 | mV |

5.13 32 kHz Crystal Oscillator Characteristics

Table 42: OSC_XTAL32K - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------------------|---|--|------|--------|-----|------|
| fCLK_EXT_32K | external clock frequency | at pin XTAL32KP/P0_23 in GPIO mode | 31 | | 33 | kHz |
| fXTAL_32K | crystal oscillator frequency | | | 32.768 | | kHz |
| ESR _{32K} | equivalent series resistance | | | | 100 | kΩ |
| C _{L_32K} | load capacitance | No external capacitors are required for a 6 pF or 7 pF crystal. | 6 | 7 | 9 | pF |
| C _{0_32K} | shunt capacitance | | | 1 | 2 | pF |
| Δf _{XTAL_32K} | crystal frequency tolerance (including aging) | Timing accuracy is dominated by crystal accuracy. A much smaller value is preferred. | -250 | | 250 | ppm |
| P _{DRV_MAX_32K} | 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 43: OSC_XTAL32K - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|--------------------------|--|------|-----|------|------|
| V _{IH_EXT_CLK_32K} | HIGH level input voltage | at pin XTAL32KP/P0_23 in GPIO input mode; XTAL32K disabled Note 1 | 0.84 | | | V |
| V _{IL_EXT_CLK_32K} | LOW level input voltage | at pin XTAL32KP/P0_23 in GPIO input mode; XTAL32K disabled Note 1 | | | 0.36 | V |

Note 1 Maximum input voltage of GPIO pins applies.

Table 44: OSC_XTAL32K - Timing Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---------------------------|---------------------------------|---|-----|-----|-----|------|
| t _{STA_XTAL_32K} | crystal oscillator startup time | Typical application, time until 1000 clocks are detected. | | 400 | | ms |

5.14 32 MHz Crystal Oscillator Characteristics

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 45: OSC_XTAL32MEG - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------------|------------------------------|---|-----|-----|-----|----------|
| $f_{XTAL}(32M)$ | crystal oscillator frequency | | | 32 | | MHz |
| ESR(32M) | equivalent series resistance | | | | 100 | Ω |
| $C_L(32M)$ | load capacitance | No external capacitors are required | 4 | 6 | 8 | pF |
| $C_0(32M)$ | shunt capacitance | | | | 7 | pF |
| $\Delta f_{XTAL}(32M)$ | crystal frequency tolerance | After optional trimming; including aging and temperature drift Note 1 | -20 | | 20 | ppm |
| $\Delta f_{XTAL}(32M)$ UNT | crystal frequency tolerance | Untrimmed; including aging and temperature drift Note 2 | -40 | | 40 | ppm |

Note 1 Using the internal varicaps a wide range of crystals can be trimmed to the required tolerance.

Note 2 Maximum allowed frequency tolerance for compensation by the internal varicap trimming mechanism.

5.15 RCX Oscillator Characteristics

Table 46: OSC_RCX - Timing Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------------------|-------------------------|--|------|-----|------|----------|
| $\Delta f_{RC}/\Delta T$ | | For temperature ranging from -40°C to 85°C | -100 | | 100 | ppm/d eg |
| $\Delta f_{RC}/\Delta V$ | | Battery Voltage | 0 | | 1100 | ppm/V |
| f_{RCX} | RC oscillator frequency | at target fixed trim setting | 13 | 15 | 17 | kHz |

5.16 32MHz RC Oscillator Characteristics

Table 47: OSC_RC32MEG - Timing Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|-------------------------|--------------------|------|------|------|------|
| $f_{RC32M_TRIMMED}$ | RC oscillator frequency | at target trimming | 30.6 | 31.5 | 32.6 | MHz |

5.17 PLL Characteristics

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 48: PLL_SYS - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------|-----------------------|--|-----|-----|-----|------|
| I _{SUP} | PLL operating current | Calculation: I _{SUP} = I _{PLL_TOT} - I _{PLL_REF} | 0.2 | 0.4 | 0.5 | mA |

Table 49: PLL_SYS - Timing Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------|-------------------------|-----------------|-----|-----|-----|------|
| t _{LOCK} | frequency settling time | 200ppm accuracy | | 30 | 100 | μs |

5.18 Charger Characteristics

Table 50: Charger - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--|--|------------------------|-----|-----|-----|------|
| V _{DROP} | dropout voltage (VBUS - VBAT) | Max. charge current | 800 | | | mV |
| I _{CHARGE_RAN} GE1 | Range of typical programmable charge currents | Charger in CC-mode | 5 | | 80 | mA |
| I _{CHARGE_RAN} GE1_stepsize | Stepsize between programmable charge currents. | Charger in CC-mode | | 5 | | mA |
| I _{CHARGE_RAN} GE2 | Range of typical programmable charge currents | Charger in CC-mode | 90 | | 240 | mA |
| I _{CHARGE_RAN} GE2_stepsize | Stepsize between programmable charge currents. | Charger in CC-mode | | 10 | | mA |
| I _{CHARGE_RAN} GE3 | Range of typical programmable charge currents | Charger in CC-mode | 260 | | 560 | mA |
| I _{CHARGE_RAN} GE3_stepsize | Stepsize between programmable charge currents. | Charger in CC-mode | | 20 | | mA |
| I _{CHARGE_accur} acy | Charge current accuracy of all typical values for Tdie from 0 to 95C | Charger in CC-mode | -10 | | 10 | % |
| I _{PRECHARGE_R} ANGE1 | Precharge range is normal range / 10 | Vbat < precharge level | 0.5 | | 8 | mA |
| I _{PRECHARGE_R} ANGE1_stepsize | Precharge current steps are normal steps / 10 | Vbat < precharge level | | 0.5 | | mA |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------------|---|---|-----|-----|-----|------|
| I_PRECHARGE_R_ANGE2 | Precharge range is normal range / 10 | Vbat < precharge level | 9 | | 24 | mA |
| I_PRECHARGE_R_ANGE2_stepsize | Precharge current steps are normal steps / 10 | Vbat < precharge level | | 1 | | mA |
| I_PRECHARGE_R_ANGE3 | Precharge range is normal range / 10 | Vbat < precharge level | 26 | | 56 | mA |
| I_PRECHARGE_R_ANGE3_stepsize | Precharge current steps are normal steps / 10 | Vbat < precharge level | | 2 | | mA |
| I_PRECHARGE_accuracy | Charge current accuracy of all typical values for Tdie from 0 to 95C | Vbat < precharge level | -15 | | 22 | % |
| I_EOCratio_RANGE1 | Ratio [%] between EOC current and normal charge current | For normal Charge current range (no pre-charge) | 4 | | 8.5 | % |
| I_EOCratio_RANGE1_stepsize | Stepsize between Ratio settings of EOC current and normal charge current | For normal Charge current range (no pre-charge) | | 1.5 | | % |
| I_EOCratio_RANGE1_accuracy | Absolute accuracy of ratio between EOC current and normal charge current | For normal Charge current range (no pre-charge) | -2 | | 2 | % |
| I_EOCratio_RANGE2 | Ratio [%] between EOC current and normal charge current | For normal Charge current range (no pre-charge) | 10 | | 16 | % |
| I_EOCratio_RANGE2_stepsize | Stepsize between Ratio settings of EOC current and normal charge current | For normal Charge current range (no pre-charge) | | 2 | | % |
| I_EOCratio_RANGE2_accuracy | Absolute accuracy of ratio between EOC current and normal charge current | For normal Charge current range (no pre-charge) | -4 | | 4 | % |
| I_EOCratio_RANGE_multiplier | Multiplication factor [in %] of IEOCratio_RANGE1 and IEOCratio_RANGE2 Multiplication involves range, stepsize and accuracy | Range multiplier is set by bit "I_EOC_DOUBLE_RANGE" = 1 | | 220 | | % |
| V_CHARGE_RANGE1 | Range of typical programmable charge voltages | Forced charge current = 1 mA | 2.8 | | 3.8 | V |
| V_CHARGE_RANGE1_step_size | Stepsize between programmable charge voltages | Forced charge current = 1 mA | | 50 | | mV |
| V_CHARGE_RANGE2 | Range of typical programmable charge voltages | Forced charge current = 1 mA | 3.8 | | 4.6 | V |
| V_CHARGE_RANGE2_step_size | Stepsize between programmable charge voltages | Forced charge current = 1 mA | | 20 | | mV |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------------|---|--|------|------|------|------|
| V_CHARGE_RAN GE3 | Range of typical programmable charge voltages | Forced charge current = 1 mA | 4.6 | | 4.8 | V |
| V_CHARGE_RAN GE3_step size | Stepsize between programmable charge voltages | Forced charge current = 1 mA | | 100 | | mV |
| V_CHARGE_ACC URACY | Charge voltage accuracy [%] | Forced charge current = 1 mA | -1.5 | | 1.5 | % |
| V_REPLENISH_r ange | Replenish voltage (where charging starts again) | Range is equal to whole "Vcharge" range | 2.8 | | 4.8 | V |
| V_REPLENISH_A CC | Accuracy of programmable replenish voltages | | -2 | | 2 | % |
| V_PRECHARGE_ range | Precharge voltage threshold | Range is equal to whole "Vcharge" range | 2.8 | | 4.8 | V |
| V_PRE_CHG_AC C | Accuracy of programmable precharge voltages | | -2 | | 2 | % |
| V_OVP_range | Overvoltage Protection level | Range is equal to whole "Vcharge" range, added with 4.9V | 2.8 | | 4.9 | V |
| V_OVP_ACC | Accuracy of programmable OVP voltages | | -2 | | 2 | % |
| NTC_ratio_0 | Voltage ratio between NTC-tap and ladder-top | Tab setting = 0; THOT comp. from 0 to 1 | 73.4 | 74.4 | 75.4 | % |
| NTC_ratio_10 | Voltage ratio between NTC-tap and ladder-top | Tab setting = 10; THOT comp. from 0 to 1 | 63.7 | 64.7 | 65.7 | % |
| NTC_ratio_20 | Voltage ratio between NTC-tap and ladder-top | Tab setting = 20; THOT comp. from 0 to 1 | 53.4 | 54.4 | 55.4 | % |
| NTC_ratio_45 | Voltage ratio between NTC-tap and ladder-top | Tab setting = 45; THOT comp. from 0 to 1 | 30.3 | 31.3 | 32.3 | % |
| NTC_ratio_55 | Voltage ratio between NTC-tap and ladder-top | Tab setting = 55; THOT comp. from 0 to 1 | 23.5 | 24.5 | 25.5 | % |
| NTC_ratio_63 | Voltage ratio between NTC-tap and ladder-top | Tab setting = 63; THOT comp. from 0 to 1 | 19 | 20 | 21 | % |
| T_DIETEMP_PR OT_RANGE | Programmable range of the die-temperature protection. | All charge modes | 80 | | 130 | °C |
| T_DIETEMP_PR OT_STEPSIZE | Stepsize of the die-temperature protection selection. | All charge modes | | 10 | | °C |
| T_DIETEMP_PR OT_accuracy | accuracy of the die-temperature protection. | All charge modes | -13 | | 13 | °C |
| T_SENSE_RANG E | Temperature Sensor range | Readout via GPADC | -40 | | 100 | °C |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---|--|--|-----|-----|-------|------|
| T _{SENSE_ACC_} _{uncalibrated} | Temperature Sensor accuracy over the whole temperature range without using calibration. | Minimum internal power dissipation Tx = (ADCx - 734)/2.3 where ADCx is 10 bit representation of GPADC | -24 | | 24 | °C |
| T _{SENSE_ACC_} _{one_point_cal} | Temperature Sensor accuracy over the whole temperature range when using one-point calibration. | Equal internal power dissipation at calibration and measurement Tx=TCAL + (ADCx - ADCcal)/2.3 Where TCAL = temperature at calibration; ADCx = 10bit representation of Tx measurement; ADCcal = 10bit representation of calibration measurement | -6 | | 6 | °C |
| T _{SENSE_TC_A} _{cc} | sense temperature coefficient accuracy [%] | Temp.Coeff = 2.3 [LSB/°C] (typ) | -5 | | 5 | °C |
| t _{Charge} _{_prot_PRECHAR} _{GE} | Precharge Time Protection[15 bits] | Timer runs in precharge mode | 1 | | 32767 | s |
| t _{Charge} _{_prot_CC-MODE} | CC-mode Charge Time Protection [15 bits] | Timer runs in CC-mode | 1 | | 32767 | s |
| t _{Charge} _{_prot_CV-MODE} | CV-mode Charge Time Protection[15 bits] | Timer runs in CV-mode | 1 | | 32767 | s |
| t _{Charge} _{_prot_TOTAL_TI} _{ME} | Total Charge Time Protection[16 bits] | Timer runs in all active charge-modes | 1 | | 65535 | s |

5.19 Battery Check Characteristics

Table 51: BATCHECK - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---------------------------------------|------------------|--|-----|-----|-----|------|
| I _{ACC_BATCHEC} _K | current accuracy | Battery Check, Load current (BATCHECK_ILOAD + 1)mA | -5 | | 5 | % |

5.20 Digital I/O Characteristics

Table 52: GPIO - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------|--------------------------|---------------------------------------|-----|--------|-----|------|
| I _{IH} | HIGH level input current | V _i =V ₃₀ =3.0V | -10 | 0.0005 | 10 | μA |

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|---------------------------|---|--------------|------|--------------|------|
| I _{IL} | LOW level input current | V _i =VSS=0V | -10 | | 10 | µA |
| I _{IH_PU_3V0} | HIGH level input current | V _i =V30=3.0V | 60 | | 180 | µA |
| I _{IL_PU_3V0} | LOW level input current | V _i =VSS=0 V, V30=3.0V | -180 | | -60 | µA |
| I _{IL_PU_1V8} | LOW level input current | V _i =VSS=0 V, V18P=1.8V | -110 | | -35 | µA |
| V _{IH} | HIGH level input voltage | V12=1.2V | 0.7*V1 2 | | | V |
| V _{IL} | LOW level input voltage | V12=1.2V | | | 0.3*V1 2 | V |
| V _{OH_1V8} | HIGH level output voltage | I _o =4.8mA, V18P=1.8V | 0.8*V1 8p | | | V |
| V _{OL_1V8} | LOW level output voltage | I _o =4.8mA, V18P=1.8V | | | 0.2*V1 8P | V |
| V _{OH_3V0} | HIGH level output voltage | I _o =4.8mA, V30=3V | 0.8*V3 0 | | | V |
| V _{OL_3V0} | LOW level output voltage | I _o =4.8mA, V30=3V | | | 0.2*V3 0 | V |
| S _R | rising slew rate | C _L =15 pF; I _L =4.8mA; | 0.4 | | 3.2 | V/ns |
| S _R _F | falling slew rate | C _L =15 pF; I _L =4.8mA; | 0.4 | | 3.3 | V/ns |
| C _{IN} | input capacitance | | | 0.75 | | pF |

Table 53: GPIO_LOWDRV - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------------|--|--|--------------|-----|--------------|------|
| V _{OH_1V8_LOW_DRV} | HIGH level output voltage, limited drive | I _o =150uA, V18P=1.8V, low drive enabled Note 1 | 0.8*V1 8P | | | V |
| V _{OL_1V8_LOW_DRV} | LOW level output voltage, limited drive | I _o =150uA, V18P=1.8V, low drive enabled | | | 0.2*V1 8P | V |

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

5.21 QSPI Characteristics

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 54: QSPIF pad - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|---|---------------------------------|--------------|------|--------------|------|
| V _{IH} | HIGH level input voltage | V18F=1.8V | 0.7*V1 8F | | | V |
| V _{IL} | LOW level input voltage | V18F=1.8V | | | 0.3*V1 8F | V |
| V _{OH_16mA} | HIGH level output voltage | I _O =16mA, V18F=1.8V | 0.8*V1 8F | | | V |
| V _{OL_16mA} | LOW level output voltage | I _O =16mA, V18F=1.8V | | | 0.2*V1 8F | V |
| I _{IH_PD} | HIGH level input current with pull-down | V _I =V18F, V18F=1.8V | 25 | | 75 | µA |
| C _{IN} | input capacitance | | | 0.87 | | pF |

Table 55: QSPIR pad - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|---|---------------------------------|--------------|------|--------------|------|
| V _{IH} | HIGH level input voltage | V18P=1.8V | 0.7*V1 8P | | | V |
| V _{IL} | LOW level input voltage | V18P=1.8V | | | 0.3*V1 8P | V |
| V _{OH_16mA} | HIGH level output voltage | I _O =16mA, V18P=1.8V | 0.8*V1 8P | | | V |
| V _{OL_16mA} | LOW level output voltage | I _O =16mA, V18P=1.8V | | | 0.2*V1 8P | V |
| I _{IH} | HIGH level input current | V _I =V18P=1.8V | -10 | | 10 | µA |
| I _{IH_PD} | HIGH level input current with pull-down | V _I =V18P=1.8V | 25 | | 75 | µA |
| I _{IL} | LOW level input current | V _I =VSS, V18P=1.8V | -10 | | 10 | µA |
| I _{IL_PU} | LOW level input current with pull-up | V _I =VSS, V18P=1.8V | -75 | | -25 | µA |
| C _{IN} | input capacitance | | | 0.87 | | pF |

5.22 LED Characteristics

Table 56: LED_DRIVER - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|----------------------|---|-----|-----|-----|------|
| I _{O_MAX_LED1} | maximum sink current | V _{LED1} =200mV, 100% duty cycle | 19 | | 21 | mA |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|----------------------|---|-----|-----|-----|------|
| I _{O_MAX_LED2} | maximum sink current | V _{LED2} =200mV, 100% duty cycle | 19 | | 21 | mA |
| I _{MATCH_LED} | current matching | relative to average LED sink current | -2 | | 2 | % |
| I _{ACC_LED1} | current accuracy | PWM accuracy at 10% duty cycle and above relative to I _{O_MAX_LED} (100% duty cycle) | -2 | | 2 | % |
| I _{ACC_LED2} | current accuracy | PWM accuracy at 10% duty cycle and above relative to I _{O_MAX_LED} (100% duty cycle) | -2 | | 2 | % |
| I _{OFF_LED1} | off-state current | driver disabled | -1 | | 10 | µA |
| I _{OFF_LED2} | off-state current | driver disabled | -1 | | 10 | µA |

Table 57: LED_DRIVER - AC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|---------------|------------|-----|-----|-------|------|
| f _{PWM_LED} | PWM frequency | | 125 | | 32000 | Hz |

5.23 LRA Characteristics

Table 58: LRA - Recommended Operating Conditions

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|--------------------|----------------------|------------|-----|-----|-----|------|
| I _{L_MAX} | maximum load current | | | 250 | | mA |

Table 59: LRA - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------|-------------------|------------|-----|-----|-----|------|
| I _{ACC_calib} | current accuracy | | | | 1 | % |
| R _{ON_HSN} | on resistance | | | | 5 | Ω |
| R _{ON_HSP} | on resistance | | | | 5 | Ω |
| R _{ON_LSN} | on resistance | | | | 5 | Ω |
| R _{ON_LSP} | on resistance | | | | 5 | Ω |
| R _{S_LSN} | series resistance | | | | 1.5 | Ω |
| R _{S_LSP} | series resistance | | | | 1.5 | Ω |
| V _{REF_OCPN} | reference voltage | | 0.4 | 0.6 | 0.8 | V |

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------|-------------------|------------|-----|-----|-----|------|
| V _{REF_OCPP} | reference voltage | | 0.4 | 0.6 | 0.8 | V |

Table 60: LRA - AC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|----------------------------------|------------|-----|-----|-----|------|
| f _{PWM_OUT} | PWM frequency of H-bridge output | | | 250 | | kHz |

5.24 USB Characteristics

Table 61: GPIO_USB - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------|---|--------------------------|-----|-----|-----|------|
| I _{IH} | HIGH level input current Note 1 | V _i =V30=3.0V | -20 | | 20 | µA |
| I _{IL} | LOW level input current | V _i =VSS=0V | -20 | | 20 | µA |

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

5.25 USB Charger Detection Characteristics

Table 62: USB_CHRG_DET - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|--------------------------|---|-----|-----|------|------|
| V _{IH_CHG_DET} | HIGH level input voltage | USB_CHARGER_CTRL_RE G[VDP_SRC_ON]=1 | 0.4 | | | V |
| V _{IL_CHG_DET} | LOW level input voltage | USB_CHARGER_CTRL_RE G[VDP_SRC_ON]=1; | | | 0.25 | V |
| V _{IH_DCP_DET} | HIGH level input voltage | USB_CHARGER_CTRL_RE G[VDM_SRC_ON]=1; V30 = 3.3V | 0.4 | | | V |
| V _{IL_DCP_DET} | LOW level input voltage | USB_CHARGER_CTRL_RE G[VDM_SRC_ON]=1; V30 = 3.3V | | | 0.25 | V |
| V _{IH_DM_VAL} | HIGH level input voltage | V30 = 3.3V | 1.5 | | | V |
| V _{IL_DM_VAL} | LOW level input voltage | V30 = 3.3V | | | 0.8 | V |
| V _{IH_DP_VAL} | HIGH level input voltage | V30 = 3.3V | 1.5 | | | V |
| V _{IL_DP_VAL} | LOW level input voltage | V30 = 3.3V | | | 0.8 | V |
| V _{IH_DM_VAL2} | HIGH level input voltage | V30 = 3.3V | 2.5 | | | V |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------|--------------------------|--|------------|------------|------------|-------------|
| V _{IL_DM_VAL2} | LOW level input voltage | V ₃₀ = 3.3V | | | 2.3 | V |
| V _{IH_DP_VAL2} | HIGH level input voltage | V ₃₀ = 3.3V | 2.5 | | | V |
| V _{IL_DP_VAL2} | LOW level input voltage | V ₃₀ = 3.3V | | | 2.3 | V |
| V _{O_DM_SRC} | output voltage | V ₃₀ = 3.3V | 0.5 | | 0.7 | V |
| V _{O_DP_SRC} | output voltage | V ₃₀ = 3.3V | 0.5 | | 0.7 | V |
| I _{DM_SINK} | D- sink current | V ₃₀ = 3.3V | 25 | | 175 | µA |
| I _{DP_SINK} | D+ sink current | V ₃₀ = 3.3V | 25 | | 175 | µA |
| I _{DP_SRC} | D+ source current | V ₃₀ = 3.3V | 5 | | 13 | µA |
| R _{DM_DWN} | D- resistance to ground | USB_CHARGER_CTRL_REG[1DP_SRC_ON] = 1; V ₃₀ = 3.3V | 14.25 | | 24.8 | kΩ |

5.26 Radio Characteristics

Table 63: Radio BLE 1M - 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 64: Radio BLE 1M - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------------|--------------------------------|--|------------|------------|------------|-------------|
| I _{BAT_RF_RX} | RX only battery supply current | radio receiver and synthesizer active; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.8 | | mA |
| I _{BAT_RF_TX_+6 dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 15; ideal DC-DC converter; T _A = 25 °C Note 1 | | 5.2 | | mA |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------------|--------------------------------|---|-----|-----|-----|------|
| I _{BAT_RF_TX_0dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 8; ideal DC-DC converter; T _A = 25 °C Note 1 | | 3 | | mA |
| I _{BAT_RF_TX_-3dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 5; ideal DC-DC converter; T _A = 25 °C Note 1 | | 2.2 | | mA |
| I _{BAT_RF_TX_-6dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 4; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.9 | | mA |
| I _{BAT_RF_TX_-12dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 2; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.4 | | mA |
| I _{BAT_RF_TX_-18dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 1; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.1 | | mA |

Note 1 The DC-DC converter efficiency is assumed to be 100 % to enable benchmarking of the radio currents at battery supply domain.

Table 65: Radio BLE 1M - AC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------------|-------------------|--|-----|-------|-----|------|
| P _{SENS_CLEAN} | sensitivity level | Dirty Transmitter disabled; DC-DC converter disabled; PER = 30.8 %; Note 1 | | -97 | | dBm |
| P _{SENS} | sensitivity level | Dirty Transmitter enabled; DC-DC converter disabled; PER = 30.8 %; Note 1 | | -96.5 | | dBm |
| P _{SENS_EPKT_CLEAN} | sensitivity level | Dirty Transmitter disabled; DC-DC converter disabled; Extended packet size (255 octets) Note 1 | | -95.5 | | dBm |
| P _{SENS_EPKT} | sensitivity level | Dirty Transmitter enabled; DC-DC converter disabled; Extended packet size (255 octets) | | -94.5 | | dBm |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|----------------------|--|---|-----|-----|-----|------|
| P _{INT_IMD} | intermodulation distortion interferer power level | 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 1 | | -28 | | dBm |
| CIR ₀ | carrier to interferer ratio | n = 0; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | 7 | | dB |
| CIR _{P1} | carrier to interferer ratio | n = +1; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -3 | | dB |
| CIR _{M1} | carrier to interferer ratio | n = -1; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -5 | | dB |
| CIR _{P2} | carrier to interferer ratio | n = +2 (image frequency); interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -30 | | dB |
| CIR _{M2} | carrier to interferer ratio | n = -2; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -37 | | dB |
| CIR _{P3} | carrier to interferer ratio | n = +3 (image frequency + 1 MHz); interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -42 | | dB |
| CIR _{M3} | carrier to interferer ratio | n = -3; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -47 | | dB |
| CIR _{P4} | carrier to interferer ratio | n = +4; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -48 | | dB |
| CIR _{M4} | carrier to interferer ratio | n = -4; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -51 | | dB |
| CIR ₅ | carrier to interferer ratio | n ≥ 5; interferer @ f ₁ = f ₀ + n*1 MHz; Note 1 | | -52 | | dB |
| P _{BL_I} | blocker power level | 30 MHz ≤ f _{BL} ≤ 2000 MHz; P _{WANTED} = -67 dBm; Note 1 | | 5 | | dBm |
| P _{BL_II} | blocker power level | 2003 MHz ≤ f _{BL} ≤ 2399 MHz; P _{WANTED} = -67 dBm; Note 2 | | 0 | | dBm |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------|------------------------------|---|-----|-------|-----|--------|
| 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 1 | | 5 | | dBm |
| L _{ACC_RSSI} | RSSI level accuracy | tolerance at 5 % to 95 % confidence interval of P _{RF} (in -90dBm to -20dBm range); burst mode, 1500 packets; | | 2 | | dB |
| L _{RES_RSSI} | RSSI level resolution | gradient of monotonous range (-90dBm to -20dBm); burst mode, 1500 packets; | | 0.5 | | dB/LSB |
| ACP _{2M} | adjacent channel power level | Output power set to 6dBm; f _{OFFS} = 2 MHz; Note 1 | | -51 | | dBm |
| ACP _{3M} | adjacent channel power level | Output power set to 6dBm; f _{OFFS} ≥ 3 MHz; Note 1 | | -56 | | dBm |
| P _{O_15} | output power level | power set to 6 dBm | | 6 | | dBm |
| P _{O_14} | output power level | power set to 5 dBm | | 5 | | dBm |
| P _{O_13} | output power level | power set to 4.5 dBm | | 4.5 | | dBm |
| P _{O_12} | output power level | power set to 4 dBm | | 4 | | dBm |
| P _{O_11} | output power level | power set to 3 dBm | | 3 | | dBm |
| P _{O_10} | output power level | power set to 2 dBm | | 2 | | dBm |
| P _{O_09} | output power level | power set to 1.5 dBm | | 1.5 | | dBm |
| P _{O_08} | output power level | power set to 0 dBm | | 0.5 | | dBm |
| P _{O_07} | output power level | power set to -1 dBm | | -1 | | dBm |
| P _{O_06} | output power level | power set to -2 dBm | | -2 | | dBm |
| P _{O_05} | output power level | power set to -3 dBm | | -3.5 | | dBm |
| P _{O_04} | output power level | power set to -6 dBm | | -5.5 | | dBm |
| P _{O_03} | output power level | power set to -8 dBm | | -8 | | dBm |
| P _{O_02} | output power level | power set to -12 dBm | | -11.5 | | dBm |
| P _{O_01} | output power level | power set to -18 dBm | | -17.5 | | dBm |
| P _{O_01A1} | output power level | power set to -22 dBm | | -22 | | dBm |
| P _{O_01A2} | output power level | power set to -26 dBm | | -26 | | dBm |
| P _{O_ULP} | output power level | power set to -50 dBm | | -51 | | dBm |

Multi-core Bluetooth 5.2 SoC family with system PMU

Note 1 Measured according to Bluetooth® Low Energy Test Specification RF-PHY.TS

Note 2 Measured according to Bluetooth® Low Energy Test Specification RF-PHY.TS. Frequencies close to the ISM band can show slightly worse performance

Table 66: Radio BLE 2M - 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 67: Radio BLE 2M - DC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-------------------------------|--------------------------------|--|-----|-----|-----|------|
| I _{BAT_RF_RX} | RX only battery supply current | radio receiver and synthesizer active; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.9 | | mA |
| I _{BAT_RF_TX_+6 dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 15; ideal DC-DC converter; T _A = 25 °C Note 1 | | 5.2 | | mA |
| I _{BAT_RF_TX_0dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 8; ideal DC-DC converter; T _A = 25 °C Note 1 | | 3 | | mA |
| I _{BAT_RF_TX_-3dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 5; ideal DC-DC converter; T _A = 25 °C Note 1 | | 2.2 | | mA |
| I _{BAT_RF_TX_-6dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 4; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.9 | | mA |
| I _{BAT_RF_TX_-12dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 2; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.4 | | mA |
| I _{BAT_RF_TX_-18dBm} | TX only battery supply current | radio transmitter and synthesizer active; power setting = 1; ideal DC-DC converter; T _A = 25 °C Note 1 | | 1.1 | | mA |

Multi-core Bluetooth 5.2 SoC family with system PMU

Note 1 The DC-DC converter efficiency is assumed to be 100 % to enable benchmarking of the radio currents at battery supply domain.

Table 68: Radio BLE 2M - AC Characteristics

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|------------------------------|--|--|-----|-------|-----|------|
| P _{SENS_CLEAN} | sensitivity level | Dirty Transmitter disabled; DC-DC converter disabled; PER = 30.8 %; Note 1 | | -94.5 | | dBm |
| P _{SENS} | sensitivity level | Dirty Transmitter enabled; DC-DC converter disabled; PER = 30.8 %; Note 1 | | -94 | | dBm |
| P _{SENS_EPKT_CLEAN} | sensitivity level | Dirty Transmitter disabled; DC-DC converter disabled; Extended packet size (255 octets) Note 1 | | -93 | | dBm |
| P _{SENS_EPKT} | sensitivity level | Dirty Transmitter enabled; DC-DC converter disabled; Extended packet size (255 octets) | | -92.5 | | dBm |
| P _{INT_IMD} | intermodulation distortion interferer power level | interferer level @ f_1, f_2 with $2*f_1 - f_2 = f_0, f_1 - f_2 = n \times 2\text{MHz}$ and $n = 3, 4, 5$; P _{WANTED} = -64 dBm @ f_0 ; PER = 30.8 %; Note 1 | | -27 | | dBm |
| CIR ₀ | carrier to interferer ratio | $n = 0$; interferer @ $f_1 = f_0 + n*2\text{MHz}$; Note 1 | | 6 | | dB |
| CIR _{P1} | carrier to interferer ratio | $n = +1$; interferer @ $f_1 = f_0 + n*2\text{MHz}$; Note 1 | | -4 | | dB |
| CIR _{M1} | carrier to interferer ratio | $n = -1$; interferer @ $f_1 = f_0 + n*2\text{MHz}$; Note 1 | | -4 | | dB |
| CIR _{P2} | carrier to interferer ratio | $n = +2$ (image frequency); interferer @ $f_1 = f_0 + n*2\text{MHz}$; Note 1 | | -31 | | dB |
| CIR _{M2} | carrier to interferer ratio | $n = -2$; interferer @ $f_1 = f_0 + n*2\text{MHz}$; Note 1 | | -36 | | dB |
| CIR _{P3} | carrier to interferer ratio | $n = +3$ (image frequency + 1 MHz); interferer @ $f_1 = f_0 + n*2\text{MHz}$; Note 1 | | -41 | | dB |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|-----------------------|------------------------------|--|-----|-----|-----|--------|
| CIR _{M3} | carrier to interferer ratio | n = -3; interferer @ $f_1 = f_0 + n \cdot 2 \text{ MHz}$; Note 1 | | -47 | | dB |
| CIR _{P4} | carrier to interferer ratio | n = +4; interferer @ $f_1 = f_0 + n \cdot 2 \text{ MHz}$; Note 1 | | -41 | | dB |
| CIR _{M4} | carrier to interferer ratio | n = -4; interferer @ $f_1 = f_0 + n \cdot 2 \text{ MHz}$; Note 1 | | -47 | | dB |
| CIR ₅ | carrier to interferer ratio | n ≥ 5; interferer @ $f_1 = f_0 + n \cdot 2 \text{ MHz}$; Note 1 | | -53 | | dB |
| P _{BL_I} | blocker power level | 30 MHz ≤ f _{BL} ≤ 2000 MHz; P _{WANTED} = -67 dBm; Note 1 | | 5 | | dBm |
| P _{BL_II} | blocker power level | 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 1 | | 5 | | dBm |
| L _{ACC_RSSI} | RSSI level accuracy | tolerance at 5 % to 95 % confidence interval of PRF (in -90dBm to -20dBm range); burst mode, 1500 packets; | | 2 | | dB |
| L _{RES_RSSI} | RSSI level resolution | gradient of monotonous range (-90dBm to -20dBm); burst mode, 1500 packets; | | 0.5 | | dB/LSB |
| ACP _{4M} | adjacent channel power level | Output power set to 6dBm; f _{OFS} = 4 MHz; Note 1 | | -57 | | dBm |
| ACP _{5M} | adjacent channel power level | Output power set to 6dBm; f _{OFS} = 5 MHz; Note 1 | | -61 | | dBm |
| ACP _{6M} | adjacent channel power level | Output power set to 6dBm; f _{OFS} ≥ 6 MHz; Note 1 | | -60 | | dBm |
| P _{O_15} | output power level | power set to 6 dBm | | 6 | | dBm |
| P _{O_14} | output power level | power set to 5 dBm | | 5 | | dBm |
| P _{O_13} | output power level | power set to 4.5 dBm | | 4.5 | | dBm |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Parameter | Description | Conditions | Min | Typ | Max | Unit |
|---------------------|--------------------|----------------------|-----|-------|-----|------|
| P _{O_12} | output power level | power set to 4 dBm | | 4 | | dBm |
| P _{O_11} | output power level | power set to 3 dBm | | 3 | | dBm |
| P _{O_10} | output power level | power set to 2 dBm | | 2 | | dBm |
| P _{O_09} | output power level | power set to 1.5 dBm | | 1.5 | | dBm |
| P _{O_08} | output power level | power set to 0 dBm | | 0.5 | | dBm |
| P _{O_07} | output power level | power set to -1 dBm | | -1 | | dBm |
| P _{O_06} | output power level | power set to -2 dBm | | -2 | | dBm |
| P _{O_05} | output power level | power set to -3 dBm | | -3.5 | | dBm |
| P _{O_04} | output power level | power set to -6 dBm | | -5.5 | | dBm |
| P _{O_03} | output power level | power set to -8 dBm | | -8 | | dBm |
| P _{O_02} | output power level | power set to -12 dBm | | -11.5 | | dBm |
| P _{O_01} | output power level | power set to -18 dBm | | -17.5 | | dBm |
| P _{O_01A1} | output power level | power set to -22 dBm | | -22 | | dBm |
| P _{O_01A2} | output power level | power set to -26 dBm | | -26 | | dBm |
| P _{O_ULP} | output power level | power set to -50 dBm | | -51 | | dBm |

Note 1 Measured according to Bluetooth® Low Energy Test Specification RF-PHY.TS

Note 2 Measured according to Bluetooth® Low Energy Test Specification RF-PHY.TS. Frequencies close to the ISM band can show slightly worse performance

6 System Overview

6.1 Internal Blocks

The DA1469x family contains the following blocks:

Arm® Cortex®-M33 CPU: This processor provides 1.5 dMIPS/MHz and is used for implementing the higher layers of the Bluetooth Low Energy (BLE) protocol (Host). It is also used for the application requirements. This includes controlling the system's power scheme for up to 144 dMIPs, if required. It has a powerful cache controller with configurable associativity, cache line size and RAM size. The CPU executes code from FLASH using a 16 kB cache controller. Code in the FLASH might be encrypted; so, decryption will happen while in progress without extra wait states.

BLE 5.x MAC: This is a Configurable MAC (CMAC) based on the Cortex-M0+ CPU and hardware accelerators implementing all timing critical tasks of the BLE Controller stack.

ROM: This is a 128 kB ROM containing the booter code as well as the routines for implementing authentication of the FLASH image (using Elliptic Curves).

OTP: This is a 4 kB One Time Programmable memory array which contains the symmetric keys for the FLASH image decryption, the symmetric keys for the application AES operations, and the public keys for the authentication of the FLASH image during boot. It also contains trim values programmed during production testing. It allows for a small secondary bootloader (if required) by the application.

Data RAM: Up to 512 kB Data RAM (DataRAM) which is shared between all masters of the system. It is used for storing code and data of the BLE MAC, the Sensor Node Controller and application data (Cortex-M33). It comprises RAM cells of 64 kB each, all with content retaining as well as complete power switch off capability.

QSPI Controllers: There are 2 QSPI controllers. One dedicated to FLASH communication (supporting XiP), the other to PSRAM/FLASH communication used to extend the embedded RAM or to store data into non-volatile while executing code. The dedicated FLASH controller supports decrypting on-the-fly while reading from the FLASH, using a dedicated AES-256-bit decryption unit, without increasing the fetch number of clock cycles.

LCD Controller: This controller supports a number of parallel and serial LCD interfaces. It incorporates a DMA that allows for autonomous operation without CPU intervention.

Cryptography Controllers: They consist of an AES 256 bits block and a HASH controller implementing SHA-1, SHA-2. This accelerates any application security requirements. A True Random Number Generator (TRNG) is also provided, that enables secure key generation.

Motor Controller: This is an autonomous gear box driver. It can be configured to control a number of different coil-based lave motors: while the rest of the system is in sleep mode. It can be used for efficiently implementing smart watches with analog hands.

UART, UART2 and UART3: Asynchronous serial interfaces. UART2 implements hardware flow control while UART3 is amended with ISO7816 functionality for connecting to a secure element. All UARTs are equipped with a FIFO of 16 bytes depth supporting. UART supports up to 1 Mbps while UART2 and UART3 can reach up to 3 Mbps.

SPI and SPI2: These are the serial peripheral interfaces with master/slave capability. They have an 8-byte/4-byte RX/TX FIFO respectively.

I2C and I2C2: These are Master/Slave I2C interfaces used for sensors and/or host MCU communication. Each controller includes a 32 locations deep FIFO (8-bits Rx, 10-bits Tx). They can both achieve up to 2.9 Mbps with 32MHz system clock.

Audio blocks: This part enables audio streaming by means of a Pulse Density Modulation (PDM), a Sample Rate Converter (SRC) and a Pulse Code Modulation (PCM) interface. It can support up to 2 digital microphones or 2 digital loudspeakers using the PDM interface or connect an external CoDec at the PCM/I2S interface.

General Purpose (GP) ADC: This is a 10-bit analog to digital converter with 8 external input channels and averaging circuitry, which increase the effective number of bits (ENOB) to 11 using oversampling up to 64 times.

Multi-core Bluetooth 5.2 SoC family with system PMU

Sigma-Delta (SD) ADC: This is a 14-bit analog to digital converter with 8 external input channels and the possibility of an external voltage reference.

Radio Transceiver: This block implements the digital and analog PHY of the Bluetooth Low Energy protocol at 2.4 GHz.

General Purpose Timers: Four general purpose timers of 24-bit width each are available for the user, 2 of them in the system power domain and 2 in their own power domain. They provide a number of features like PWM generation, 2 capture channels that save a snapshot of the timer, up/down counting with free-running mode, selectable clock source and one-shot pulse generation with configurable width.

Real Time Clock: This is a hardware controller that supports the complete time of day clock: 12/24 hours, minutes, seconds, milliseconds and hundredths of milliseconds. It comprises 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 milliseconds.

Watchdog Timers: The system comprises 2 watchdog timers, 13-bit wide each. One for CMAC SW monitoring (CMAC Wdog) and another for the System CPU (System Wdog). The System watchdog is constantly counting down, automatically started right after POWERUP, it is powered by the always on domain and will generate an NMI and a HW Reset when 0 and -16 is reached respectively. Its maximum counting time is 84 seconds or 3 minutes depending on the clock used (RC32K or RCX). The CMAC Watchdog resides in the Radio power domain and will generate an interrupt to the CMAC CPU when 0 is reached. It will also generate a HW reset if -16 is reached. Both watchdogs are automatically frozen when either of the two CPUs is in debug mode.

Wakeup Controller: This is a controller for capturing external events that can be used as a wake-up trigger on any of the GPIO ports with programmable polarity. It comprises a single debouncing structure for generating a wake-up interrupt upon a button press.

LRA/ERM Driver: This block implements the haptic driver for external linear resonant actuators (LRA) or eccentric rotating mass (ERM) motors. It comprises an H-Bridge and automatically adapts to the resonant frequency of the haptic feedback actuator. It has a configurable supply current that can be used to set the force of the haptic feedback.

White LED Drivers: There are 2 white LED drivers that are able to sink up to 20 mA current. Their intensity is controlled by a dedicated configurable PWM signal. It provides programmability regarding the amount of sinking current while sustaining an accuracy of +/- 5%.

USB FS Device: This is a 12 Mbit/s USB device controller, which is mainly used for software upgrades. It is also used for recharging the system's battery. It supports 7 endpoints.

DMA Engine: This is a general-purpose DMA engine with 8 channels that can be multiplexed to support data transfers between memory resources but also between memory and peripherals in single or burst modes (where applicable). It is also used when secure features are enabled to perform key transfers from OTP to registers without CPU having access.

Sensor Node Controller: This is a seven instructions state machine that acts as a smart DMA controller and can be programmed to manipulate all serial interfaces. It is primarily designed to program, serve and fetch data from external sensors to RAM, where it has a direct connection to improve speed.

6.2 Digital Power Domains

The DA1469x supports a number of digital power domains that can be turned ON and OFF independently from one another:

Multi-core Bluetooth 5.2 SoC family with system PMU

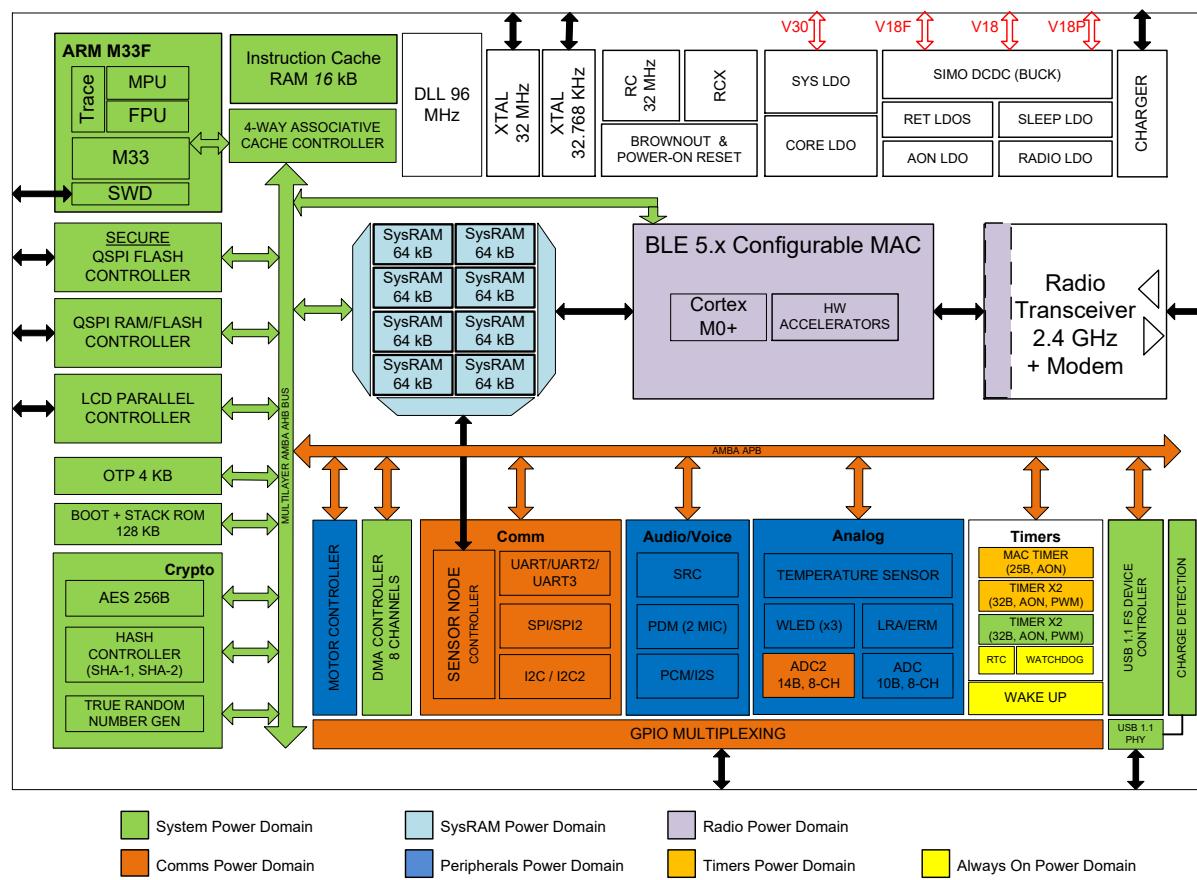


Figure 7: Digital Power Domains and Blocks Mapping

Description and domain names used throughout this document is shown in the following table:

Table 69: Power Domains Description

| Domain Name | Description |
|-------------|--|
| PD_SYS | System Power Domain. It comprises the Cortex-M33 CPU, the QSPI controllers, the LDC controller, OTP, ROM, USB, DMA, Crypto blocks and 2 general purpose Timers. It also contains the AHB multilayer bus. |
| PD_AON | Always-On Power Domain. It comprises the wake-up controller, the system watchdog and power management as well as clock generation circuitry. It also contains the RTC. |
| PD_COM | Communications Power Domain. It comprises the Sensor Node Controller, all serial interfaces and SD ADC. It also supplies the GPIO multiplexing. |
| PD_MEM | Memory Power Domain. It comprises the memory controller, the DCDC digital FSM and the RAM cells. |
| PD_TMR | Timer Power Domain. It comprises the MAC timer, 2 general purpose Timers and the XTAL32M digital state machine. |
| PD_PER | Peripherals Power Domain. It comprises the Audio blocks, the Motor Controller, the GP ADC, the WLEDs and LRA and the on-chip temperature sensors. |
| PD_RAD | Radio Power Domain. It comprises the CMAC and the digital PHY of the Radio. |

6.3 HW FSM (POWERUP, WAKEUP, GOTO Sleep)

6.3.1 HW FSM

The HW FSM responsible for the POWERUP, WAKEUP and GOTO sleep process of the system is presented in the following flow chart:

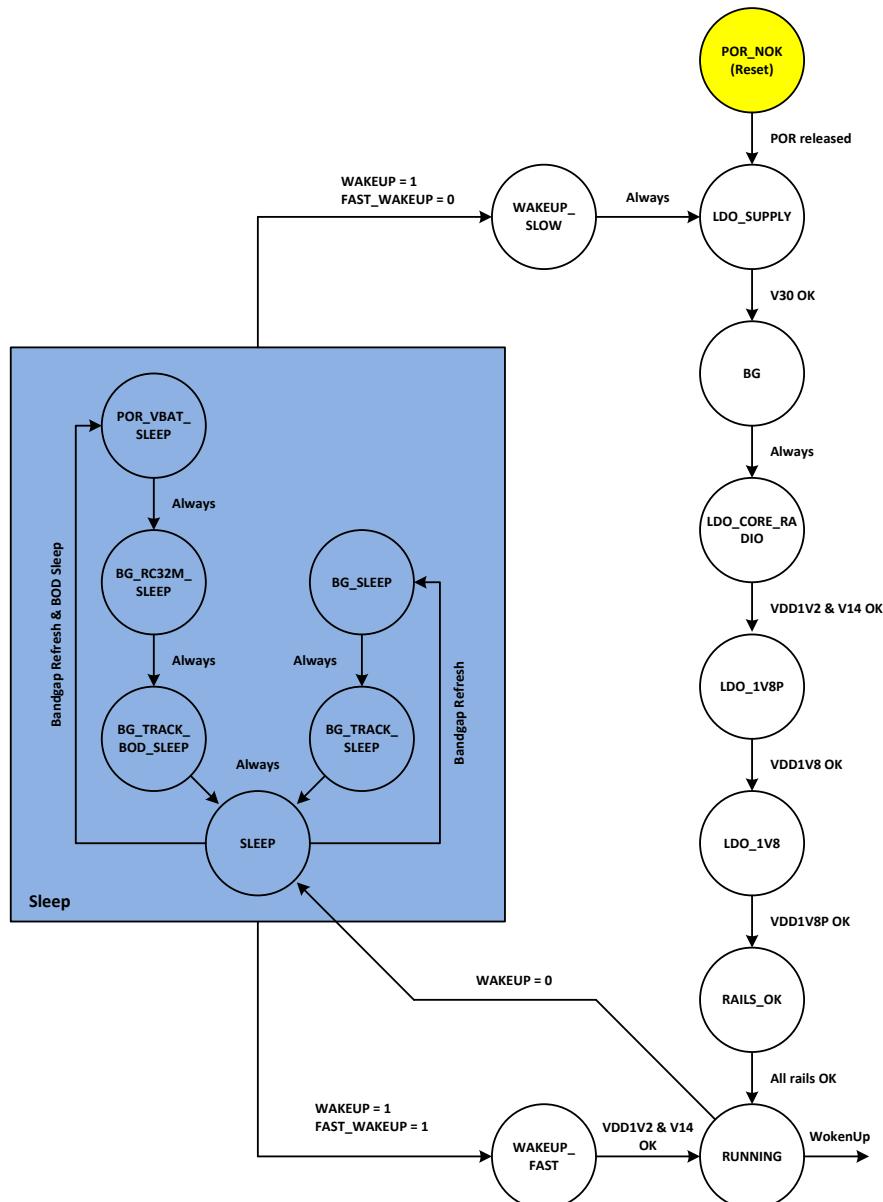


Figure 8: POWERUP, WAKEUP, GOTO Sleep HW FSM

The WAKEUP signal indicates that the hardware FSM has finished, and the digital power domains will be initialized. This signal can be monitored at register bit field SYS_STAT_REG[POWER_IS_UP].

6.3.2 POWERUP

After POR is released, the FSM enters state LDO_SUPPLY where the following actions are taken

- The COLD_BOOT bit is asserted indicating that the system recovers from a POR (previous state has to be POR_NOK)

Multi-core Bluetooth 5.2 SoC family with system PMU

- The LDO_VBAT will be enabled. The LDO requires some time to settle so the voltage is stable at its output. This time is always <300 µs. A timeout mechanism is set for 400 µs.

The next state (BG) triggers the following operations:

- Bandgap is enabled

The latency of this state is always one 32 kHz clock cycle.

In the next state (LDO_CORE_RADIO), the

- LDO_CORE is enabled depending on the programmed register bit¹
- LDO_RADIO is enabled depending on the programmed register bit

It should never exceed 150 µs, so a timeout is set for this value. When this state is done, both the LDO_CORE is regulating supplying the core with 0.9V and the LDO_RADIO with 1.4V the V14 rail.

The next 2 states will enable the V18P and V18 (provided that the respective bits are enabled). A timeout of 300 µs is set for both of these power rails.

A final state of one clock cycle is used for checking if all Rails are powered (RAILS_OK) before releasing the system to RUNNING state, and assert the WOKENUP signal (which can be mapped on a GPIO or monitored at a register bit), allowing SW to start running.

The following timing diagram summarizes the aforementioned description:

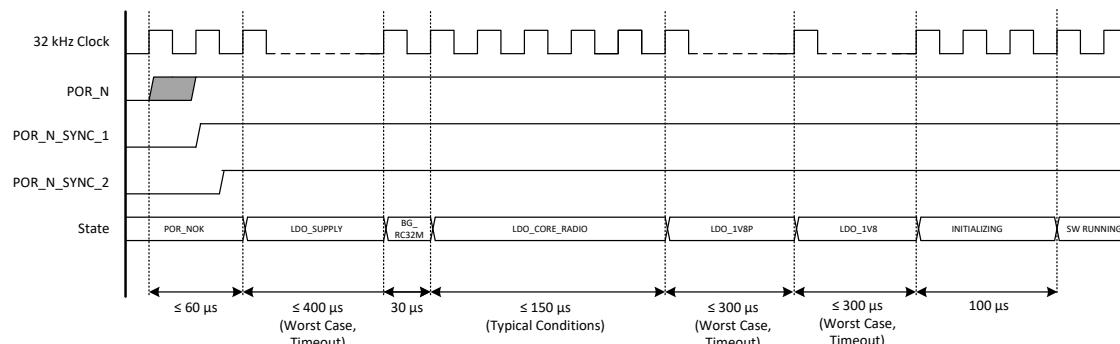


Figure 9: POWERUP Timing Diagram

6.3.3 WAKEUP Options

The system supports 3 different WAKEUP options. It can be configured which one to select depending on the application requirements and constraints. [Table 70](#) summarizes the WAKEUP modes characteristics.

Table 70: Wake Up Modes

| Wake up Mode | Latency | Recommended | Description | Constraints |
|--------------|---------|---|--|-------------|
| Slow | 600 µs | If system sleeps for a long time (>8 sec) | All LDOs of the Power Management are powered sequentially. Software is released to run after all LDO OK signals are evaluated | None |

¹ These bits can be programmed in the POWER_CTRL_REG

| Wake up Mode | Latency | Recommended | Description | Constraints |
|--------------|-------------|------------------------------|---|--|
| Fast | 300 μ s | If system sleeps for <8 sec | All LDOs of the Power Management are powered simultaneously. Software is released to run after Core and Radio LDO OK signals are evaluated. | None |
| Ultra-Fast | ~100 us | If system sleeps for <100 ms | All LDOs of the Power Management are powered simultaneously. Software is released 2 low power clocks after wake-up trigger without evaluating any LDO ok signal | No heavy load (i.e. >0.5 mA) should be applied for the first 100 us. The VDD_SLEEP_LEVEL voltage must be 0.9 V. |

6.3.3.1 Slow WAKEUP

The slow WAKEUP is following the states of the cold boot. It will, however, de-assert the COLD_BOOT flag. The timing diagram of the slow WAKEUP is presented in the following figure.

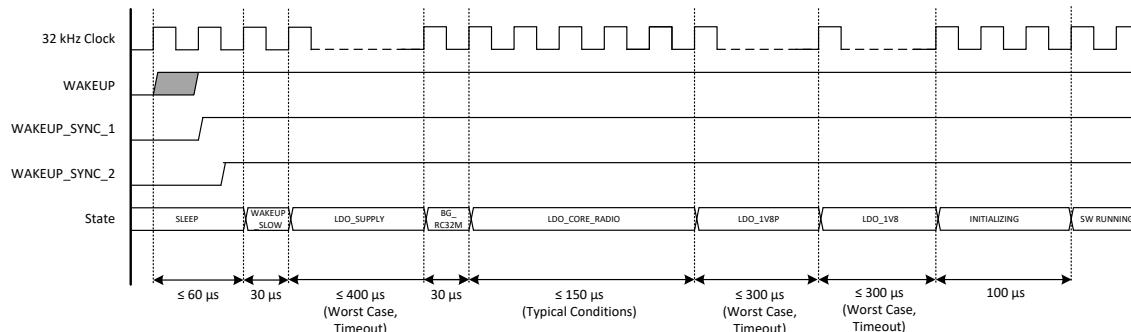


Figure 10: Slow Wake-Up Timing

A typical latency of the slow wake up, is in the range of 1 ms, while maximum (assuming time outs) can reach 1.5 ms. This WAKEUP mode is recommended in cases where the device has been in sleep mode for over 4 seconds.

6.3.3.2 Fast WAKEUP

Enabling Fast WAKEUP mode will really reduce the wake-up time. This is done by enabling all LDOs at the same state, and checking for just core and radio voltage levels before proceeding to the next state. Waking up in this mode will also de-assert the COLD_BOOT flag.

Latency of this mode should be no more than 300 μ s. The timing diagram is presented in the following figure.

Multi-core Bluetooth 5.2 SoC family with system PMU

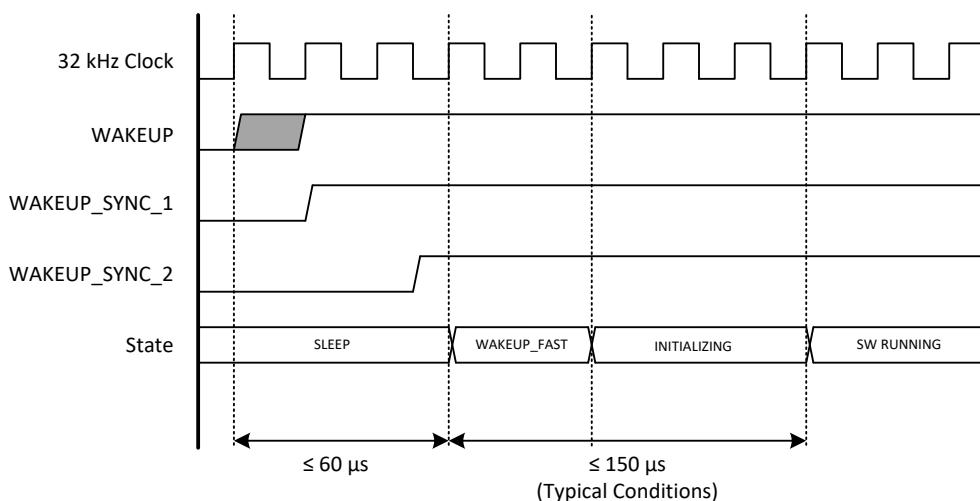


Figure 11: Fast WAKEUP timing

This mode should be selected if the system is in sleep mode for time intervals less than 4 seconds.

6.3.3.3 Ultra-Fast WAKEUP

An ultra-fast WAKEUP mode has been implemented to further reduce wake up time down to ~100 us. If this mode is activated, no checking will be performed on LDO feedback signals. It will be assumed that core voltage level is ok and thus CPU is allowed to run right after a couple of low power clock cycles required for sampling the wakeup trigger event whether it comes from a GPIO or an internal timer. The timing diagram of the ultra-fast WAKEUP, is presented in the following figure:

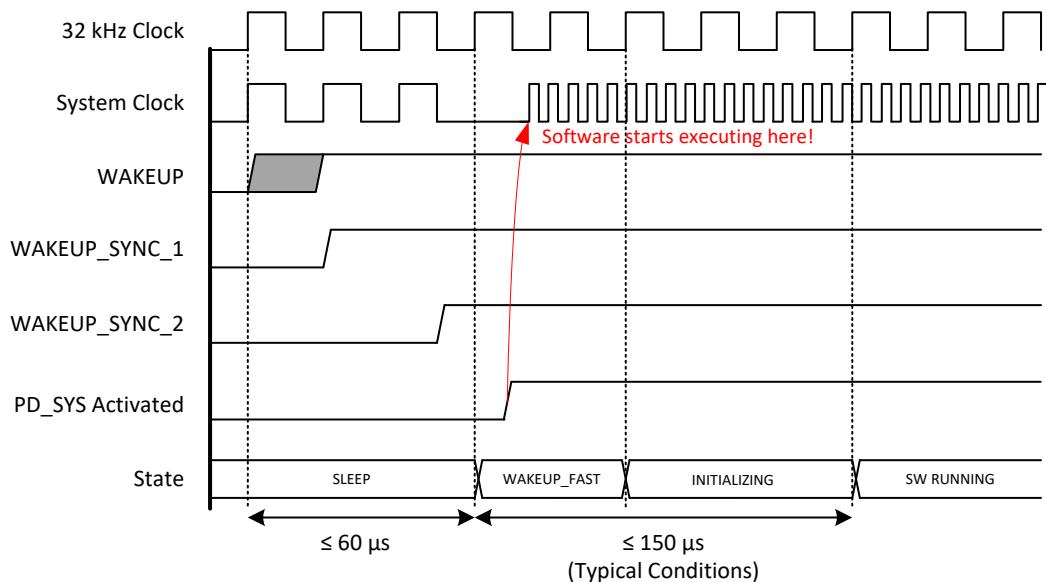


Figure 12: Ultra Fast Wake-Up timing

Note that, the HW FSM is still implementing a “Fast WAKEUP” in parallel as shown in the timing diagram under the “State” signal. Until the RUNNING state is entered (which designates that power is indeed ok and more current can be dissipated), it is recommended that the CPU runs at 32 MHz (using the internal RC32MHz) and not at higher frequencies (PLL). It is also recommended not to enable external devices powered by the DA1469x while the internal LDOs are not properly activated (i.e. State has reached “SW Running”).

6.3.4 Go-To-Sleep

The DA1469x can go to deep sleep only if the Power Domain Controller (see [PDC chapter](#)) has no pending activity from any of the three masters of the system. If this is the case, then the system enters the sleep state where two operations can be configured to happen on a periodic basis:

1. Refresh the reference of the LDO_CORE_RET which provides the sleep voltage by activating the bandgap every PMU_SLEEP_REG[BG_REFRESH_INTERVAL] low power clocks.
2. Check on the voltage levels by allowing the BOD to sense each voltage rail while the bandgap is enabled. BOD is then running on a 1MHz clock derived by the RC32M oscillator. Detailed timing diagrams can be found at the [BOD chapter](#).

The exact configuration of LDOs while getting from RUNNING into SLEEP can be found in Table 71.

6.4 Power Modes and Rails

There are 4 main power modes in the DA1469x family:

Hibernation mode. This is the “shipping mode”. There is no RAM retained, no clocks running (so no RTC), all domains are off, the system can only be woken up by POR, HW reset or a GPIO trigger. To enter this mode, RC32K has to be selected as the low power clock. This is the oscillator started after a GPIO trigger that wakes up the complete system. Note that, when woken up, the system resets.

Deep Sleep mode. No RAM is retained, XTAL32K is running (so RTC is on), all domains are off, the system can only be woken up by POR, RTC alarm or a GPIO trigger.

Extended Sleep mode. Programmable RAM can be retained, XTAL32K/RCX is running (so RTC is ON or OFF depending on the sleep clock), all domains are OFF, the system can only be woken up by POR, RTC alarm (if XTAL32K is used), MAC timer, other Timer or a GPIO trigger.

Active/Idle. System is up and running, with a number of power domains enabled depending on the use case. For Sensor Node, for instance, PD_COM and the PD_MEM is enabled. If Cortex- M33 processing power is needed, for example, then all domains might be enabled.

Note that there is no state retention at the CPU, so it will always wake up in reset state. Required registers will have to be saved and restored to/from RAM upon waking up.

A number of different power rails are supplied in the DA1469x family power management. A detailed description of the architecture can be found in the [power architecture diagram](#). The actual rail supply, driving elements, and configuration capabilities are presented in the following table:

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 71: Power Rails Configuration

| Rail | Supplied by | V_OUT (V) | I_LOAD (mA) | Activation Register(s) | HW Wake-UP FSM | HW Go-to-Sleep FSM | Remarks |
|-------|--------------|---------------|-------------|---|----------------|--------------------|--|
| V30 | LDO_VBAT_RET | 3 / 3,3 /3,45 | 10 | LDO_3V0_RET_ENABLE_ACTIVE LDO_3V0_RET_ENABLE_SLEEP | Programmable | Programmable | Precise voltage: 3V and approximate voltage: 3V45 available. If LDO_3V0_RET_ENABLE_SLEEP=1 is set, the LDO will be used in sleep. If LDO_3V0_RET_ENABLE_ACTIVE=1 is set, it will be used in active. Reset value is enabled in sleep, disabled in active |
| | LDO_VBAT | 3 / 3,3 /3,45 | 150 | LDO_3V0_MOD_E | Programmable | OFF | Only used in active. Four settings: 0: Disabled 1: Always use LDO_VBAT 2: Always use LDO_VBUS 3: Automatically select depending on VBUS level |
| | LDO_VBUS | 3 / 3,3 /3,45 | 150 | LDO_3V0_MOD_E | Programmable | OFF | Same as above |
| | VSYS_CLAMP | ~2,4 | 1 | CLAMP_3V0_VBAT_ENABLE | Programmable | Programmable | This clamp could be used for minimum current dissipation, provided that there are no external components powered by the PMU, both at V30, as well as V18 or V18P. Clamp so voltage depends on temperature, processing, load, etc. This clamp cannot supply appreciable loads |
| Vcont | VCNT_CLAMP | ~2,4 | 0,01 | | Always on | Always on | This is an internal rail supplying the RCX |
| V18 | LDO_IO | 1,2 / 1,8 | 50 | LDO_1V8_ENABLE | Programmable | OFF | |
| | LDO_IO_RET | 1,2 / 1,8 | 10 | LDO_1V8_RET_ENABLE_SLEEP LDO_1V8_RET_ENABLE_ACTIVE | Programmable | Programmable | Same functionality as with LDO_VBAT_RET |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Rail | Supplied by | V_OUT (V) | I_LOAD (mA) | Activation Register(s) | HW Wake-UP FSM | HW Go-to-Sleep FSM | Remarks |
|------|--------------|--------------|-------------|---|----------------|--------------------|---|
| | DCDC_V18 | 1,2 / 1,8 | 50 | DCDC_V18_EN ABLE_HV/LV | N/A | N/A | DCDC output is always handled by the application SW |
| V18P | LDO_IO2 | 1,8 | 50 | LDO_1V8P_EN ABLE | Programmable | OFF | |
| | LDO_IO_RET2 | 1,8 | 10 | LDO_1V8P_RET ENABLE_SLEEP LDO_1V8P_RET ENABLE_ACTIVE | Programmable | Programmable | Same functionality as with LDO_VBAT_RET |
| | DCDC_V18P | 1,8 | 50 | DCDC_V18P_E NABLE_HV/LV | N/A | N/A | DCDC output is always handled by the application SW |
| V18F | Follows V18P | Follows V18P | 50 | SW_1V8F_ENA BLE | Programmable | OFF | This is a different pin connected to the V18P rail via a switch. The <i>switch</i> is controlled by SW_1V8F_ENABLE |
| V12 | LDO_CORE | 0,9 / 1,2 | 50 | LDO_CORE_EN ABLE | Programmable | OFF | For 32 MHz operation, a 0.9V setting should be used. If 96 MHz is required, then LDO_CORE should be switched to 1.2 V prior to enabling the PLL |
| | LDO_CORE_RET | 0,75 / 0,9 | 1 | LDO_CORE_RET ENABLE_SLEEP LDO_CORE_RET ENABLE_ACTIVE | Programmable | Programmable | Same functionality as LDO_VBAT_RET. Core can run on this LDO when it is programmed at 0.9 V and load is low (ultrafast WAKEUP) |
| | VDD_CLAMP | 0,8 - 1,1 | 0,01 | | Always on | Always on | This is only used at STARTUP to supply the AON power domain. Voltage level can be trimmed by CLAMP_VDD_LEVEL[3:0] |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Rail | Supplied by | V_OUT (V) | I_LOAD (mA) | Activation Register(s) | HW Wake-UP FSM | HW Go-to-Sleep FSM | Remarks |
|------|-------------|---------------------------|-------------|---------------------------|----------------|--------------------|--|
| | DCDC_V12 | 0,9 / 1,2 | 50 | DCDC_VDD_EN ABLE_HV/LV | N/A | N/A | DCDC output is always controlled by the application SW |
| V14 | LDO_RADIO | 1,2 – 1,55 50 mV steps | 20 | LDO_RADIO_ENABLE | Programmable | OFF | Activated by the <i>booter</i> (if cold boot) or by the application SW (if WAKEUP). Turned off by FSM in sleep |
| | DCDC_V14 | 1,2 – 1,55 50 mV steps | 20 | DCDC_V14_EN ABLE_HV/LV | N/A | N/A | DCDC output is always controlled by the application SW |

6.5 OTP

6.5.1 OTP Segments

The OTP consists of several different segments which are described in the following table.

Table 72: OTP Layout

| Segment | Bytes | Description | OTP Address |
|---------|-------|---|-------------|
| 1 | 1024 | Configuration Script ~100 registers write operations | 0x000000C00 |
| 2 | 256 | QSPI FW Decryption Keys Area – Payload write/read protected when secure mode enabled in CS Secure mode connects those (8 * 256-bits) keys to QSPI Controller | 0x000000B00 |
| 3 | 256 | User Data Encryption Keys – Payload Write/Read protected when secure mode enabled in CS. Secure mode connects those (8 * 256-bits) keys to AES engine | 0x000000A00 |
| 4 | 32 | QSPI FW Decryption Keys Area – Index Eight entries for eight 256-bit keys | 0x0000009E0 |
| 5 | 32 | User Data Encryption Keys – Index 8 entries for 8 256-bit keys | 0x0000009C0 |
| 6 | 256 | Signature Keys Area – Payload | 0x0000008C0 |
| 7 | 32 | Signature Keys Area – Index | 0x0000008A0 |
| 8 | 2208 | Customer Application Area (Secondary bootloader, binaries, e.t.c.) | 0x000000000 |

Segment 1 contains the “Configuration Script” of the system. This is a script with a certain format, which is mainly executed by the booter.

6.5.2 Configuration Script

It is a table of 32-bit entries, X spaces deep. X cannot be longer than 256 places (It means that the CS can use 1Kbyte of total space).

The script is used for programming registers with values that are defined during production testing, storing a trim value for the application software, and defining UART time-out time during booting. The booter will execute the script to prepare and initialize the system, before the CPU starts running application code from the FLASH.

The format of the commands in the script is one of the seven cases that follow:

Table 73: Configuration Script 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. These are always in pairs with the address sitting in even memory addresses |

Multi-core Bluetooth 5.2 SoC family with system PMU

| # | Command Type | Description |
|---|--------------------------|--|
| 3 | Trim Value | <ul style="list-style-type: none"> One 32-bit word which is equal to 0x9000YYXX indicating that the next word is a value stored during production testing. More specifically: <ul style="list-style-type: none"> 9: indicates that the following word(s) are not to be stored to registers but will be used by the SDK SW YY: indicates that YY amount of words follow XX: is an increasing value and can be used for indexing by the SW application. If YY>1 then this number will not be increased for the words that belong to the same value One or more 32-bit words which represent the value |
| 4 | Booter Value | One 32-bit word which is equal to 0x6XXXXXXXXX indicating this is a value pointing to the Flash product header in flash at address 0XXXXXXXXX |
| 5 | Development Mode Disable | One 32-bit word which is equal to 0x70000000, disabling the development mode. Development Mode is enabled by default at the initialization phase of the booter |
| 6 | UART STX Timeout | One 32-bit word which is equal to 0x8XXXXXXXXX. The XXXXXXXX is used to program the selected STX timeout in multiples of 100 us. So i.e. 0x80000040 is 40x100 µs = 4 ms |
| 7 | Stop Command | One 32-bit word containing 0x00000000 designating that the configuration script has reached the end and execution should be terminated |

The above-mentioned cases are summarized in the following table, illustrating an example.

Table 74: Configuration Script example

| Words | Even Words | Odd Words | Description |
|-------|------------|------------|---|
| 0-1 | 0xA5A5A5A5 | 0x80000028 | Start command of the CS Script, followed by STX timeout value of 4mS (40x 100µs) |
| 2-3 | <Address> | <Value> | Booter will automatically write to <Address>, the <Value> |
| 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 will automatically write to <Address>, the <Value> |
| 10-11 | <Address> | <Value> | Booter will automatically write to <Address>, the <Value> |
| 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> | 0x60001000 | Booter value: address of the product header in flash |
| 18-19 | 0x70000000 | 0x00000000 | Disable development mode command, followed by Stop command. Booter will stop running script after stop command so anything after this is don't care |

6.5.3 Keys and Indexing

There are three different groups of keys in the OTP, namely:

1. The QSPI FW decryption keys group.
2. The user application symmetric keys group.
3. The signature keys group.

Multi-core Bluetooth 5.2 SoC family with system PMU

The first contains the keys that can be used for decryption-on-the-fly while the CPU is executing code in place from the FLASH with help of the cache controller. The second, contains user defined keys that can/will be used by the application with help of the crypto block (AES accelerator). The last, contains public keys used for authentication of the FLASH image while booting (Secure Boot).

Each group has its own index section. There are eight entries in every index section, each entry is initially 0xFF. Every index entry corresponds to a 256-bit key. If an entry is written with 0x00, then the respective key is revoked and hence not used anymore. Revocation is only done through the booter, as explained in section [6.7.4](#).

6.5.4 Customer Application Area

A 2 kBytes space can be used for a small secondary bootloader that runs a limited amount of software. The system can be programmed to remap address zero to the OTP base address. So right after booting, the CPU will start executing code from the OTP base address.

6.6 FLASH

The FLASH is partitioned in regions so that it can support multiple images in one FLASH device, and let the cache controller remap to them individually. By default, the FLASH region size is set at 0.5 Mbytes, but this can be defined in the configuration script by programming the CACHE_FLASH_REG[FLASH_REGION_SIZE].

All different firmware images start at a region aligned address. The first region is not used for a FLASH image, but reserved for the product header. This contains important information about the system and FLASH type.

Multi-core Bluetooth 5.2 SoC family with system PMU

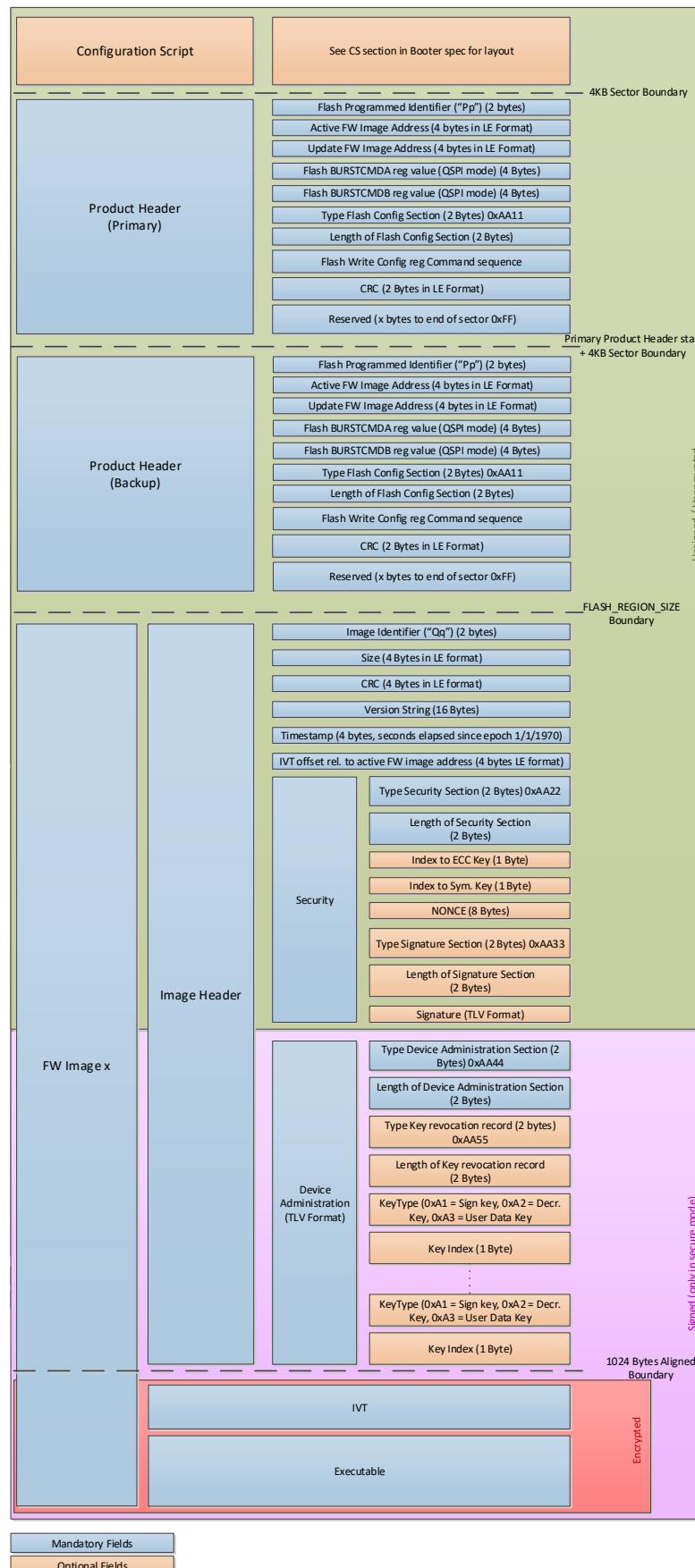


Figure 13: FLASH Regions and Layout

Multi-core Bluetooth 5.2 SoC family with system PMU

In Figure 13, orange stands for optional while blue for mandatory fields.

Padding of 0xFF is used between the end of the Image header and the start of the Image itself.
Image is always 1 Kbyte aligned.

In case the image header contains a Device Administration section, the signature will be calculated over this section, the padding and the image. On the other hand, if Device Administration section is not included, the signature will be calculated over the image only.

6.7 Booting

The booter will always be executed when a POR, a HW Reset or the RESET_ON_WAKEUP feature is configured. Different booting flavors are supported:

- Boot from cached QSPI FLASH without secure features, configuration script in OTP
- Boot from cached QSPI FLASH without secure features, configuration script in FLASH
- Boot from cached QSPI FLASH with secure features, configuration script in OTP
- Boot from UART without FLASH or secure features
- Boot from OTP without FLASH or secure features

The booter will also detect available software updates and apply them according to the FLASH header. Note that the booter cannot boot a flash image if the Product Header or the active image partition is beyond the 128Mbit address (0x1000000) in the flash. This happens because the flash opcode that allows for reading addresses larger than 24 bits is not unified across all flash vendors.

The Boot flow is divided into five separate phases:

1. Initialization
2. Run Configuration Script
3. Retrieve application code
4. Device administration
5. Load image

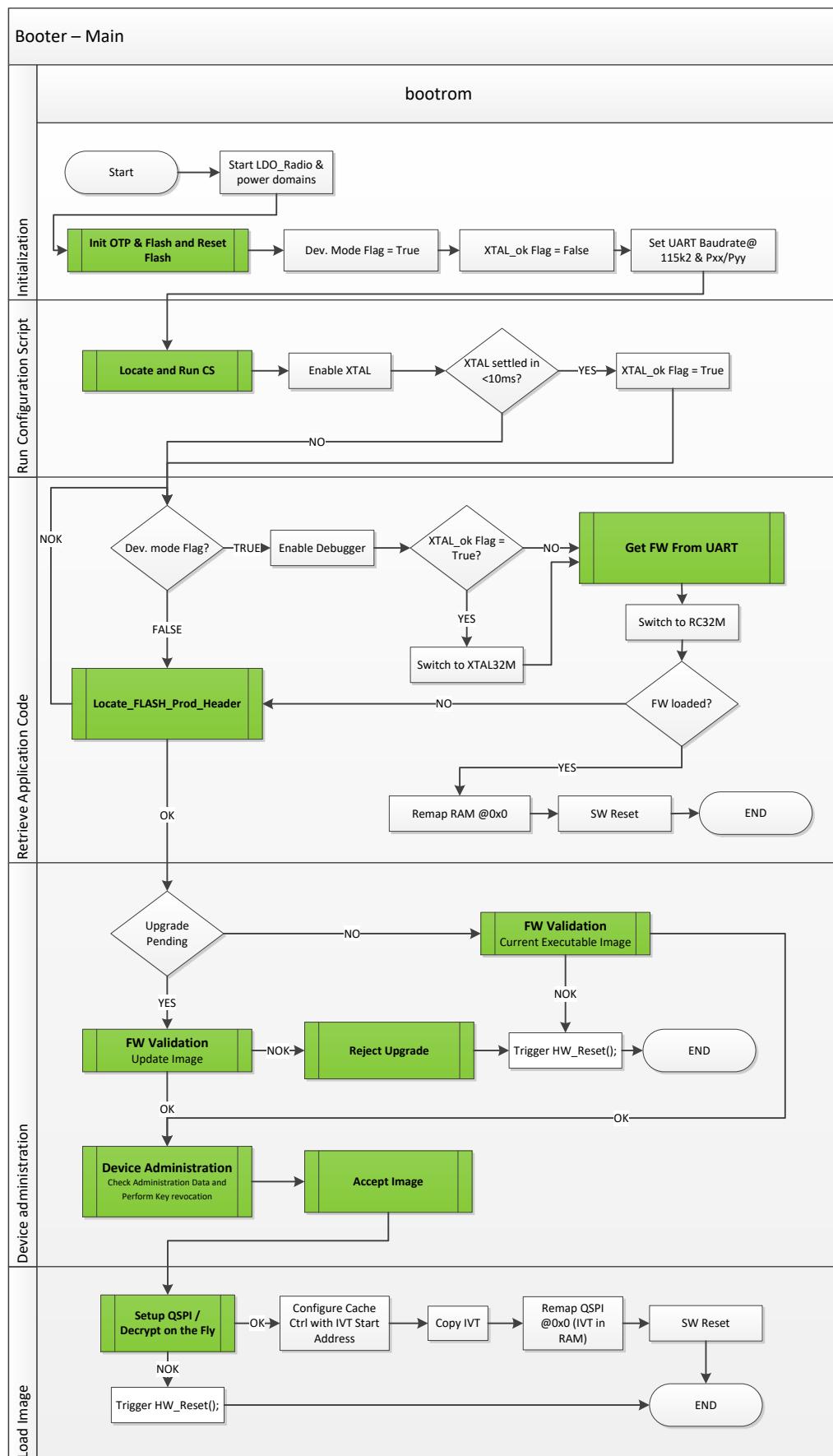


Figure 14: BootROM Flowchart

6.7.1 Initialization

The initialization phase takes care of starting the LDO_radio and enabling the Power domains. Then it will initialize the OTP Controller, QSPI interface and Clocks. Following this, it will enable Development mode. This is done so that it can be disabled by the CS in the next phase if desired. Development mode can be used for having the debugger on and update the device firmware using the UART.

Next it will set the XTAL_ok flag to “False” (it will be set to “True” if the XTAL settled correctly at a later phase).

Finally, the booter will initialize the UART (but not enable it yet) with the default settings, which can be still be overwritten by the configuration script residing in the OTP.

6.7.2 Configuration Script

This phase will try to locate and execute the Configuration Script (CS) from OTP or FLASH. It is expected to be at address 0x00000C00 in OTP, or at the start of the QSPI FLASH. If none is found the booter will just continue to the next step assuming that there is no configuration script in place.

The way a CS is detected is by looking at the location where it is expected and verify if the CS Start command is found. If a CS is found, either in OTP or FLASH, it will be parsed and executed. It will stop parsing when any of the following happens:

1. It reaches the stop command.
2. It reaches the first empty entry.
3. It reaches the maximum length.

After the CS is processed, XTAL32M is enabled and the booter checks if the settle and trim ready bits are set within 10 ms. If not, the XTAL_ok flag remains “False”. If settled correctly it sets to “True”. This flag will later be used to detect if it is safe to switch to XTAL32M, or to continue using the default system clock: RC32M.

6.7.3 Retrieve Application Code

During this phase, the booter scans to check if the development mode flag is disabled or not. If it is still in development mode, then it will enable the Debug interface and try to boot from UART. If booting from UART is successful, then it will issue a SW reset after having remapped address zero to RAM. If not, then it will try to locate a valid FLASH header, trying to boot from FLASH.

If not in development mode, the booter will continue to the next phase after having identified a valid product header in the FLASH. A valid product header is identified by a programmed “FLASH programmed identifier” and an “Active FW Image Address” as shown in [Figure 13](#).

6.7.4 Device Administration

The device administration phase is used to check for pending updates and validate the FLASH images. It also, processes corresponding image headers and revoke keys, if needed.

The booter will check if the “Active FW Image address” and the “Upgrade Image address” fields of the FLASH product header are the same. If not, an upgrade image is available. The FW validation will be executed if the secure boot bit has already been set by the Configuration Script (CS). Validation is done by identifying the public key, checking if the key is already revoked and if not, proceed with invoking the Ed25519 verification algorithm. Note that, this happens with the PLL being enabled and the system CPU running at 96 MHz.

After the image is authenticated, the booter checks for any “key revocation record” in the FLASH image. This will trigger revocation of the key that is currently in use. Note that, the key revocation requires an OTP write as explained in section [6.5.3](#).

Multi-core Bluetooth 5.2 SoC family with system PMU

6.7.5 Load Image

In this final phase, the actual FW image is loaded. This is done by setting up the QSPI & cache controllers and executing the QSPI Loader, which is located in the Product header of the FLASH. If secure boot is enabled, the QSPI controller will initialize to decrypt on the fly. The cache controller is then configured to point to the interrupt vector table (copied in RAM) and address zero will be remapped to QSPI FLASH. Any error condition during this or previous phases, will result in a HW reset.

6.8 Memory Map

The mapping of the system's internal resources is presented in the following table. Note that *_C defines a Controller (registers) while *_M defines Memory (RAM) space. All resources are 32-bit aligned.

Table 75: Memory Map

| Resource | Start Address | End Address | Size (KBytes) | PD | AMBA | Comments |
|------------------|---------------|-------------|---------------|--------|------|---|
| Remapped Devices | 0 | 800000 | 8192 | PD_SYS | AHB | Remap IVT into SYSRAM |
| SYSRAM (code) | 800000 | 880000 | 512 | PD_MEM | AHB | Remapped at 0x0. DA14691 end address: 0x860000 |
| Reserved | | | | | | |
| ROM | 900000 | 920000 | 128 | PD_SYS | AHB | Remapped at 0x0 |
| Reserved | | | | | | |
| CACHE_RAM | 10060000 | 10064000 | 16 | PD_SYS | AHB | |
| Reserved | | | | | | |
| OTPC_C | 10070000 | 10080000 | 64 | PD_SYS | AHB | |
| OTPC_M | 10080000 | 10090000 | 64 | PD_SYS | AHB | First 4 Kbytes Remapped at 0x0 |
| Reserved | | | | | | |
| CACHE_C | 100C0000 | 100C0100 | 0,25 | PD_SYS | AHB | |
| Reserved | | | | | | |
| QSPIF_M | 16000000 | 18000000 | 32768 | PD_SYS | AHB | First 8 Mbytes Remapped at 0x0 |
| QSPIF_C | 18000000 | 1A000000 | 32768 | PD_SYS | AHB | |
| Reserved | | | | | | |
| SYSRAM (data) | 20000000 | 20080000 | 512 | PD_MEM | AHB | Same physical address as SYSRAM(code) |
| Reserved | | | | | | |
| AHB_DMA_B | 30020000 | 30020400 | 1 | PD_SYS | AHB | |
| Reserved | | | | | | |
| LCD_C | 30030000 | 30040000 | 64 | PD_SYS | AHB | |
| AES_HASH_C | 30040000 | 30050000 | 64 | PD_MEM | AHB | |
| TRNG_M | 30050000 | 30060000 | 64 | PD_SYS | AHB | |
| Reserved | | | | | | |
| OTPC_C | 30070000 | 30080000 | 64 | PD_SYS | AHB | Same physical address as |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Resource | Start Address | End Address | Size (KBytes) | PD | AMBA | Comments |
|------------|---------------|-------------|---------------|--------|-------|--|
| | | | | | | 0x10070000 |
| OTPC_M | 30080000 | 30090000 | 64 | PD_SYS | AHB | Same physical address as 0x10080000 (Note 3) |
| PATCH | 30090000 | 30090800 | 2 | PD_SYS | AHB | |
| Reserved | | | | | | |
| QSPIR_M | 32000000 | 34000000 | 32768 | PD_SYS | AHB | |
| QSPIR_C | 34000000 | 36000000 | 32768 | PD_SYS | AHB | |
| QSPIF_M | 36000000 | 38000000 | 32768 | PD_SYS | AHB | Same physical address as 0x16000000 (Note 3) |
| QSPIF_C | 38000000 | 3A000000 | 32768 | PD_SYS | AHB | Same physical address as 0x18000000 |
| Reserved | | | | | | |
| CMAC | 40000000 | 40020000 | 128 | PD_RAD | AHB | |
| RFCU | 40020000 | 40020200 | 0,5 | PD_RAD | AHB | |
| RFCU_POWER | 40020200 | 40021000 | 3,5 | PD_RAD | AHB | |
| DEMOD | 40021000 | 40022000 | 4 | PD_RAD | AHB | |
| SYNTH | 40022000 | 40023000 | 4 | PD_RAD | AHB | |
| Reserved | | | | | | |
| CRG_AON | 50000000 | 50000100 | 0,25 | PD_AON | APB32 | |
| WKUP | 50000100 | 50000200 | 0,25 | PD_AON | APB32 | |
| PDC | 50000200 | 50000300 | 0,25 | PD_AON | APB32 | |
| DCDC | 50000300 | 50000400 | 0,25 | PD_AON | APB32 | |
| RTC | 50000400 | 50000500 | 0,25 | PD_AON | APB32 | |
| Reserved | | | | | | |
| WDOG | 50000700 | 50000800 | 0,25 | PD_TIM | APB32 | |
| Reserved | | | | | | |
| XTAL32M_C | 50010000 | 50010200 | 0,5 | PD_TIM | APB32 | |
| TIMER | 50010200 | 50010300 | 0,25 | PD_TIM | APB32 | |
| TIMER2 | 50010300 | 50010400 | 0,25 | PD_TIM | APB32 | |
| MAC_TIM | 50010400 | 50010500 | 0,25 | PD_TIM | APB32 | |
| Reserved | | | | | | |
| UART | 50020000 | 50020100 | 0,25 | PD_COM | APB32 | |
| UAR2 | 50020100 | 50020200 | 0,25 | PD_COM | APB32 | |
| UART3 | 50020200 | 50020300 | 0,25 | PD_COM | APB32 | |
| SPI | 50020300 | 50020400 | 0,25 | PD_COM | APB32 | |
| SPI2 | 50020400 | 50020500 | 0,25 | PD_COM | APB32 | |
| Reserved | | | | | | |
| I2C | 50020600 | 50020700 | 0,25 | PD_COM | APB32 | |
| I2C2 | 50020700 | 50020800 | 0,25 | PD_COM | APB32 | |
| SDADC | 50020800 | 50020900 | 0,25 | PD_COM | APB32 | |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Resource | Start Address | End Address | Size (KBytes) | PD | AMBA | Comments |
|------------------|---------------|-------------|---------------|--------|-------|----------|
| CRG_COMM | 50020900 | 50020A00 | 0,25 | PD_COM | APB32 | |
| GPIOMUX | 50020A00 | 50020C00 | 0,5 | PD_COM | APB32 | |
| SENSORN_C | 50020C00 | 50020D00 | 0,25 | PD_COM | APB32 | |
| SENSORN_M | 50020D00 | 50020E00 | 0,25 | PD_COM | APB32 | |
| Reserved | | | | | | |
| PWMWLED | 50030500 | 50030600 | 0,25 | PD_PER | APB32 | |
| SRC/PDM | 50030600 | 50030700 | 0,25 | PD_PER | APB32 | |
| PCM | 50030700 | 50030800 | 0,25 | PD_PER | APB32 | |
| TEMPSENSE | 50030800 | 50030900 | 0,25 | PD_PER | APB32 | |
| GPADC | 50030900 | 50030A00 | 0,25 | PD_PER | APB32 | |
| LRA | 50030A00 | 50030B00 | 0,25 | PD_PER | APB32 | |
| ANAMISC | 50030B00 | 50030C00 | 0,25 | PD_PER | APB32 | |
| CRG_PER | 50030C00 | 50030E00 | 0,5 | PD_PER | APB32 | |
| SMOTOR | 50030E00 | 50030F00 | 0,25 | PD_PER | APB32 | |
| Reserved | | | | | | |
| USB_C | 50040000 | 50040200 | 0,5 | PD_SYS | APB32 | |
| VERSION | 50040200 | 50040300 | 0,25 | PD_SYS | APB32 | |
| GPREG | 50040300 | 50040400 | 0,25 | PD_SYS | APB32 | |
| CHARGER | 50040400 | 50040500 | 0,25 | PD_SYS | APB32 | |
| CRG_2 | 50040500 | 50040600 | 0,25 | PD_SYS | APB32 | |
| RFMON | 50040600 | 50040700 | 0,25 | PD_SYS | APB32 | |
| Reserved | | | | | | |
| DMA_C | 50040800 | 50040A00 | 0,5 | PD_SYS | APB32 | |
| TIMER3 | 50040A00 | 50040B00 | 0,25 | PD_SYS | APB32 | |
| TIMER4 | 50040B00 | 50040C00 | 0,25 | PD_SYS | APB32 | |
| TRNG_C | 50040C00 | 50040D00 | 0,25 | PD_SYS | APB32 | |
| Reserved | | | | | | |
| MEMCTRL_C | 50050000 | 50050100 | 0,25 | PD_MEM | APB32 | |
| Reserved | | | | | | |
| Arm Internal Bus | E0000000 | FFFFFFFFFF | | PD_SYS | | |

Note 3 Access to QSPI Flash memory from peripherals (e.g. DMA), is done through 0x36000000 memory space and access to OTP memory through 0x30080000

6.9 Resource Sharing

The DA1469x has three processing units that might request access to one or more of the system's peripheral controllers. The application software can map certain resources to one of the processing units. But there are times, for example, when CMAC needs to use the General-Purpose ADC to read the die temperature and trigger a radio calibration, while the same ADC is required by the application software running a State of Charge algorithm by checking the Battery Voltage on a regular basis.

To avoid race conditions and provide all three processing units with a robust way of identifying who the owner of the peripheral is, a hardware mutex is implemented. This is a 32-bit register

Multi-core Bluetooth 5.2 SoC family with system PMU

(BUSY_STAT_REG) which can be set/reset by using write only registers (BUSY_SET_REG) and (BUSY_RESET_REG). This stops race conditions happen because two processing units are writing at the same time. Two bits are reserved per resource, so that the value represents the owner of the resource:

- 0x0: resource is available for use
- 0x1: resource is busy, controlled by the Sensor Node Controller
- 0x2: resource is busy, controlled by the Cortex-M33
- 0x3: resource is busy, controlled by the CMAC

Such a mutex register, which decides resources that require sharing, can be defined by application software. [Table 76](#) shows a possible configuration of such a register.

Table 76: Busy Status Register

| 31-30 | 28-29 | 26-27 | 24-25 | 22-23 | 20-21 | 18-19 | 16-17 | 14-15 | 12-13 | 10-11 | 8-9 | 6-7 | 4-5 | 2-3 | 0-1 | Bit |
|----------|------------------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-----|------|-----|-------|------|-----|
| RESERVED | Motor Controller | Timer2 | Timer | UART3 | ADC | PDM | SRC | PCM | ADC2 | I2C2 | I2C | SPI2 | SPI | UART2 | UART | |

Notice that the register is in the PD_MEM, which is automatically activated if one of the three masters is alive. However, if the system is in deep sleep, then this register will not be retained.

6.10 Remapping

Remapping options are explained in the following table:

Table 77: Remapping Options

| Remap Field in the SYS_CTRL_REG – 3 bits | |
|--|--------------------------------------|
| 0x0 | Remap address 0 to ROM |
| 0x1 | Remap address 0 to OTP |
| 0x2 | Remap address 0 to FLASH cached area |
| 0x3 | Remap address 0 to SysRAM1 |
| 0x4 | RESERVED |
| 0x5 | Remap address 0 to SysRAM2 |
| 0x6 | Remap address 0 to Cache RAM |
| 0x7 | RESERVED |

In the case of FLASH cached (default use case), CACHE_FLASH_REG contains the base address and offset of the FLASH image.

6.11 Security Features

The DA1469x supports a number of security features that can be configured. This is done using the configuration script to program the following write-one-only (sticky) register bits (Note that these bits can only be reset by HW or PORReset):

Table 78: Security Configuration Options

| Bit field | Description |
|-------------------------|--|
| FORCE_DEBUGGER_OFF | This bit will permanently disable the M33 debugger |
| FORCE_CMAC_DEBUGGER_OFF | This bit will permanently disable the CMAC debugger. |
| PROT_QSPI_KEY_READ | This bit will permanently disable CPU read capability at OTP offset 0x00000B00 and for the complete segment |
| PROT_QSPI_KEY_WRITE | This bit will permanently disable ANY write capability at OTP offset 0x00000B00 and for the complete segment |
| PROT_AES_KEY_READ | This bit will permanently disable CPU read capability at OTP offset 0x00000A00 and for the complete segment. The AES sections are only used by the application SW, but protecting the key area from read/write makes it secure after leaving the manufacturing facilities |
| PROT_AES_KEY_WRITE | This bit will permanently disable ANY write capability at OTP offset 0x00000A00 and for the complete segment. The AES sections are only used by the application SW, but protecting the key area from read/write makes it secure after leaving the manufacturing facilities |
| PROT_SIG_KEY_WRITE | This bit will permanently disable ANY write capability at OTP offset 0x000008C0 and for the complete segment. This is for protecting public keys from being written (used by ECC only) |
| SECURE_BOOT | This bit will enable authentication of the image in the FLASH while the system is booting |

6.11.1 Secure Keys Manipulation

This feature allows for programming up to 8 different 256-bit symmetric keys for each of the encrypted image or the user application cases and up to 8 different 256-bit ECC keys for the authentication of the FLASH image. A revocation mechanism is supported through the booter as explained in previous sections allowing for changing the current key of any of the 3 operations while the product is in the field.

6.11.2 Secure Boot

The feature is enabled by programming the SECURE_BOOT_REG[SECURE_BOOT] bit in the configuration script in the OTP.

This forces authentication of the FLASH image before booting is finished. The booter code will start the PLL switch the system clock to 96 MHz and then execute the Ed25519 verification algorithm. If the generated signature and the signature stored in the FLASH match, authentication is successful and booting is continued. If not, a HW reset is issued.

The aforementioned process represents the “FW Validation” state in Figure 14.

6.11.3 Secure Access

Permanently disabling the JTAG interface, prevents unwanted access to the DA1469x. This is done by programming the SECURE_BOOT_REG[FORCE_DEBUGGER_OFF] in the configuration script. This disconnects the SWD signals from the CPU's SWD controller.

Except for the sticky bit, the debugger has its own enable bit; namely, the SYS_CTRL_REG[DEBUGGER_ENABLE] which is by default disabled. This bit is enabled during booting, at the “Retrieve Application Code” phase (see [Figure 14](#)).

If nothing is programmed in the configuration script, JTAG will be enabled a few microseconds after POWERUP. If the sticky bit is programmed, JTAG will be permanently disabled.

6.11.4 Validation

Every device can be uniquely identified using the Position, Package, and Time Stamp information put in the configuration script during the production test. This is a 64-bit word, which contains information about the position of the die, the wafer number, the package, and the time stamp of the production testing that compared to the Tester ID and site.

6.11.5 Cryptography Operations

The DA1469x is equipped with HW acceleration for supporting all modern cryptography operations. More specifically, it comprises:

- A 256-bit capable AES encryption/decryption and key expansion engine that implements ECB/CBC/CTR modes covering all symmetric key application needs
- A complete HASH block supporting up to SHA 512 bits
- A real hardware True Random Number Generation, capable of generating 1024 random bits in 64k clock cycles

7 Power

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

7.1 Introduction

The DA1469x has a complete integrated Power Management Unit (PMU). This includes a Single Inductance Multiple Output (SIMO), a DC-DC converter with 4 outputs, a number of LDOs for the different power rails of the system, a Constant-Current-Constant-Voltage (CCCV) charger for battery recharging, and a charge detection circuit. The PMU is capable of supplying external devices even when the DA1469x is in sleep mode. Figure 15 shows the system diagram of the analog Power Management Unit (PMU).

Features

- Synchronous Single Inductance Multiple Output Buck DC-DC converter with 4 output power rails
- Programmable DC-DC converter output charging sequence
- Two DC-DC converter outputs at 1.8 V with 50 mA load capability for powering external devices
- One LDO output up to 3.45 V with up to 150 mA drive capability
- DC-DC converter ON/OFF control per output
- Active and Sleep mode current limited LDOs
- Use of small external components
- Supply of external rails (V30, VDD1V8, VDD1V8P) while DA1469x is in Sleep mode
- CC/CV Charger with battery/die-temperature protection
- Interrupt line for the DCDC converter

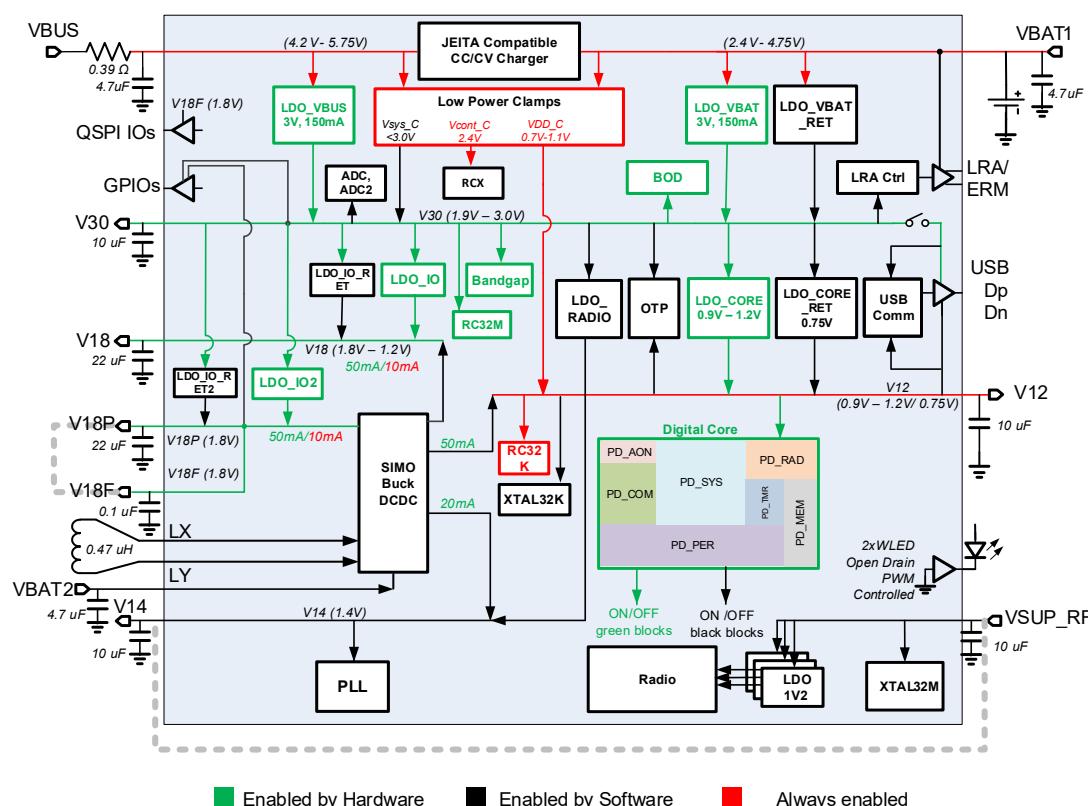


Figure 15: Power Management Unit Architecture

Note that, all decoupling capacitor values listed in Figure 15 are nominal.

7.2 Architecture

There are three main power inputs, namely VBUS, VBAT1, and VBAT2. The VBUS is connected when charging the battery through the USB connector. VBAT1 should be shorted with VBAT2 externally; it supplies the LDOs, while VBAT2 supplies the SIMO DCDC converter. There are certain parts of the PMU which are always powered. They are shown in red in Figure 15. The always-ON power circuitry has two clamps and LDO_SLEEP. This supplies the voltage needed when the system is in extended sleep, deep sleep or hibernation mode. When the system wakes up, then many of the blocks of the PMU are activated automatically (shown in green). This is explained in the power/wake up sequence flow charts. Finally, SW is responsible for activating the SIMO DCDC and any other block shown in black in Figure 15.

The digital core is further divided into power domains as explained in the “System Overview” section. These domains are controlled by the Power Domains Controller.

7.2.1 SIMO DC-DC Converter

The heart of the PMU is the SIMO Buck DC-DC converter. The block diagram is displayed in the Figure 16:

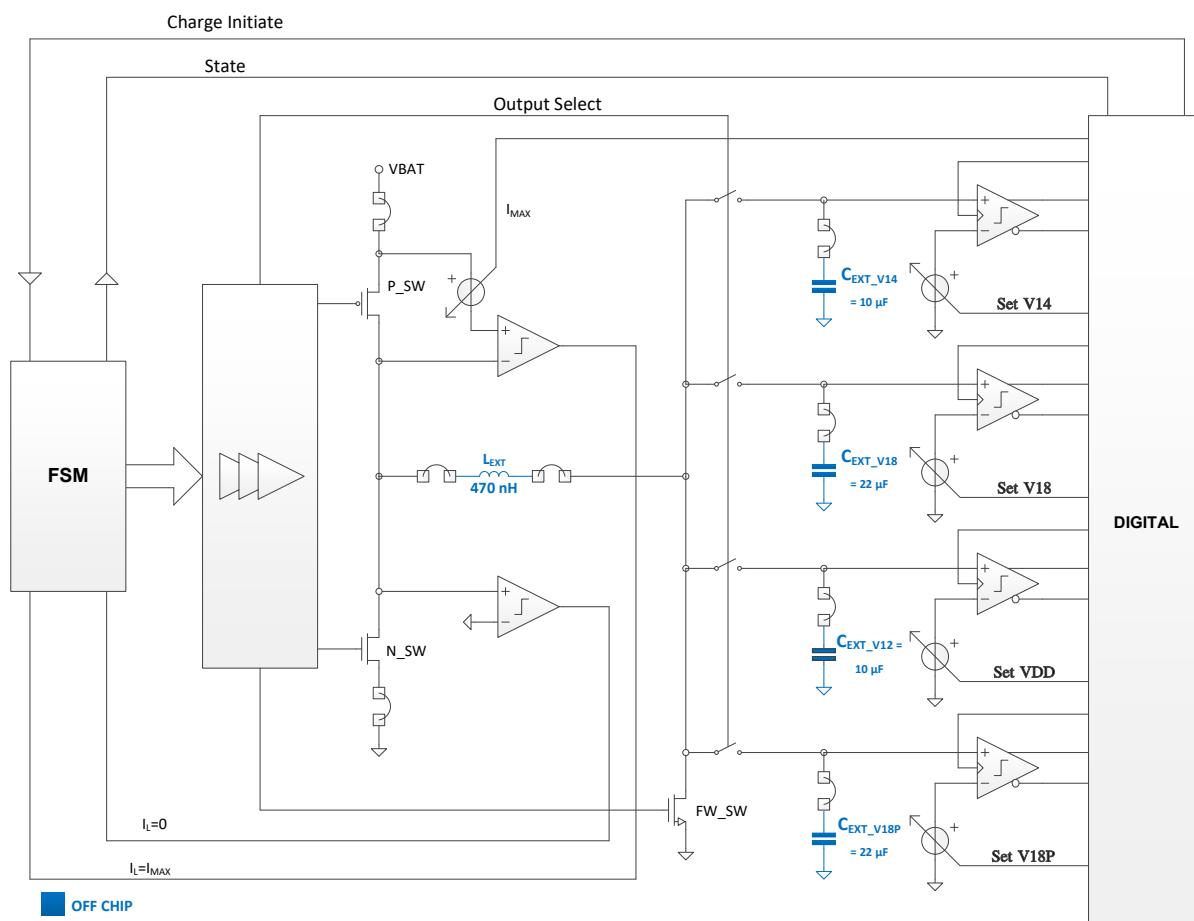


Figure 16: SIMO DCDC Block Diagram

The DCDC converter has four outputs:

- V18P which delivers 50 mA when DA1469x is in active. The voltage range of this power rail is 1.8 V +/- 5%. External devices or sensors that need to be constantly powered ON, should be connected to this power rail
- V18F which is used for supplying the external Quad SPI FLASH. This rail's characteristics are identical to the V18P

Multi-core Bluetooth 5.2 SoC family with system PMU

- V14 which is connected externally on the PCB with VSUP_RF and delivers up to 20 mA at 1.4 V and should not be used to supply external devices
- V12 that supplies the digital core of the DA1469x, and delivers up to 50 mA at 1.2 V when in Active mode. This rail should not be used to supply external devices

The converter has an asynchronous architecture, i.e. the ON-time of the switches is not determined by an external clock. Instead, the ON-time is determined by a (dynamically varying) current limit in the external inductor. If one of the output voltages is too low (determined using clocked comparators), a charge cycle is triggered.

However, it is possible for more than one output to be below minimum at the same time. In such a case, the system has to decide which output to charge first. This is done using a priority select register, which holds the sequence in which the outputs will be charged. Each time one or more outputs require a charge cycle, the system sorts the outputs based on this priority register and loads this sequence in a four tab shift register.

To minimize the ripple voltage on the outputs, the current limit is dynamically set during operation in Active mode. This is done by measuring how long each output is above its minimum value after a charge cycle. If this time is too long, more charge than required (given the load current) was stored on the output capacitor, so the current limit is reduced by one bit (LSB). However, when the output voltage drops too quickly, not enough charge was delivered to the output capacitor, so the current limit is increased by one bit (LSB).

7.2.2 LDOs

Several LDOs are used to provide a stable power supply to all rails, when the SIMO DCDC is not active (e.g. in Sleep mode or during start up) or when the device is plugged onto a USB charger. Furthermore, bypassing the DCDC is also considered, when the external voltage on pin VBAT2 is at the edge of enabling an efficient step-down activity (i.e. < 2.3 V). This is done automatically by HW or manually by SW.

Two low-power LDOs (LDO_ret) one connected to VBUS and one to VBAT, provide power to the Vcont power line and hence to the LDO_sleep (which is a clamp actually). This LDO is responsible for providing the VDD supply during Sleep mode, which can be configured down to 0.75 V. This is basically the supply of the Always ON power domain (PD_AON) which is constantly powered, independently of active or any sleep mode.

In Sleep modes the retention LDOs might take over and make sure that the system is properly powered without the need of the DC-DC converter. The LDO_VBAT_RET provides power to the System supply line the LDO_SLEEP at Vcore, and LDO_IO_RET/LDO_IO_RET2 to the external 1.8 V power rails. There is no need to power the Vradio since it is not enabled in any of the sleep mode. The LDO_VBAT_RET, LDO_CORE_RET and LDO_IO_RETx circuits, are identical and operate in a sample & hold manner: They contain a reference voltage capacitance which is used to regulate the output voltage. However, due to leakage, this internal reference capacitor is discharged. To keep a stable voltage reference, a mechanism is built to start the Bandgap, sample the voltage reference in the LDOs, and shut it down again. This periodic operation is programmable in terms of timing with use of the BG_REFRESH_INTERVAL, which counts sleep clock ticks.

In active mode, when external supply is between 1.7 V and 2.4 V and the DC-DC converter is bypassed (stepdown conversion not feasible due to low voltage), the LDO_VBAT provides power to the Vsys line and the LDO_IO/LDO_IO2, LDO_Core and LDO_radio to the 1.8 V rails, the Vcore and the Vradio, respectively.

Finally, when the system is connected to a USB charger, pin VBUS is the source of the power instead of pin VBAT1/VBAT2. The same path is used as with VBAT2, but the LDO_USB is responsible for providing the System supply line with power. This LDO is automatically switched on as soon as a VBUS>VBAT1 voltage is sensed.

7.2.3 Switching between DC-DC and LDOs

In general, when one of the masters of the system (M33, M0+ or SNC) needs to turn ON, the DCDC will be allowed to do so. DCDC will not be turned off by any master. When the system is allowed to

Multi-core Bluetooth 5.2 SoC family with system PMU

go to sleep, thus activating the HW FSM, then the DCDC controller will be automatically turned OFF, since its digital state machine lives in the PD_MEM which will power down.

DCDC will not be enabled by the HW FSM. However, it will be enabled by SW, running on either the M33, the M0+ or the SNC. It will be done by first enabling the DCDC engine (DCDC_CTRL1_REG[DCDC_ENABLE]) and then activating the respective output of the DCDC which is of interest (DCDC_Vxx_ENABLE_xV bits).

When the latter occurs, the other drivers of the same rail (namely the LDO or LDO_RET) will be automatically disabled by HW hence, there will never be more than one driver on each power rail.

The LDOs will be activated again after WAKEUP, by the HW FSM according to the configuration in the POWER_CTRL_REG. During this time, the DCDC registers will be reset.

7.2.4 Low Power Clamps

This block consists of clamps that are delivering power to the V30 rail and the V12 rail. These are simple low-power clamps, not regulators hence they will not keep the voltage stable under any load.

The V30 supply follows the VBAT or VBUS voltage with a threshold difference. Its main purpose is to be able to provide some current to the V30 and start the Bandgap while the system is powering up.

The V12 supply will keep the digital Always-ON block alive when in Hibernation mode. There is no need for the bandgap or any Sample & Hold retention LDO to be alive. This clamp is programmable and can be lowered during hibernation to reduce dissipation as much as possible.

7.2.5 Battery Check

The Battery Check (BATCHCHECK) features a programmable VBAT load current, which can be used to modulate the battery ([Figure 17](#)). The constant load is a programmable current source from 0 to 8 mA in steps of 1 mA. The accuracy is +/-2.7%. The internal resistance of a battery can be monitored in this way.

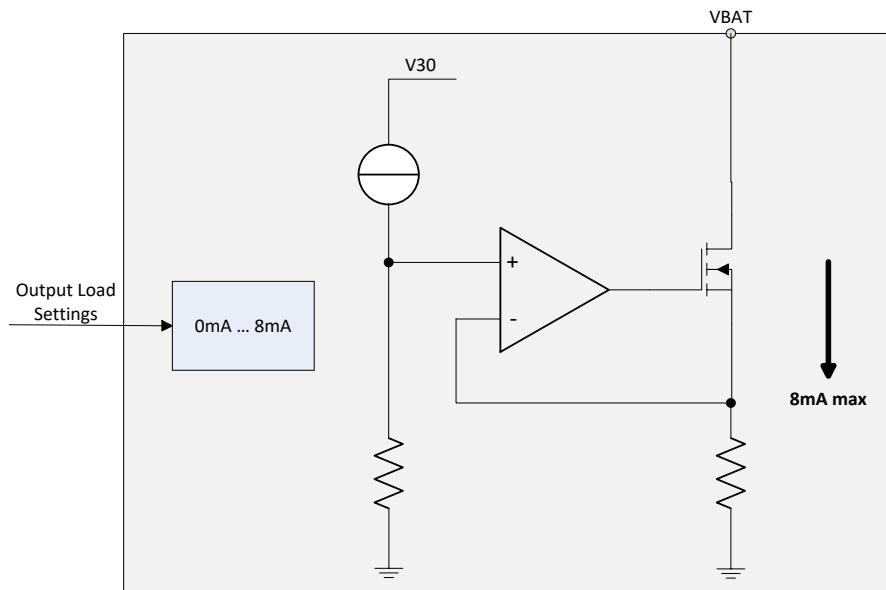


Figure 17: Battery Check Block Diagram

Multi-core Bluetooth 5.2 SoC family with system PMU

The Battery Check contains the following functional blocks:

- Output load setting. Input is an accurate reference current from a bandgap; outputs are a number of (mirrored) currents.
- High gain op-amp
- Matched resistor network

It provides a reference current, coming from the bandgap, which is mirrored into selectable output current(s). This current is routed through a resistor, resulting in a reference voltage on the positive input of the op-amp. Due to the negative feedback and the high gain, the op-amp in combination with the output transistor, will try to force the same voltage on the negative input as well as over the output resistor, resulting in the selected output load current.

Register BATCHECK_REG controls the setting, fine tuning and feature enablement.

7.2.6 Charger

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | x | ✓ | ✓ | ✓ |

The integrated battery charger is suitable for charging different types of batteries (NiMH, Li-phosphate, Li-Co, Li-Mn, NMC). The charger uses an internal pass-device, which limits the external components to a minimum: Only an external buffer capacitor and a temperature-sensor (NTC) are needed.

The charger has a CCCV (Constant Current Constant Voltage) architecture and has battery temperature and chip-temperature protection. The charge levels are between 2.8 V and 4.9 V while the charge currents can be set between 5 mA and 560 mA (Normal charging) or 0.5 mA and 56 mA (Pre-Charge).

Enabling, disabling, and functional states of the charger are controlled by a HW state machine, exceptions and error handling is done via software. All protections and loops for current, voltage, and temperature are autonomous and implemented in hardware. There are indication signals towards the digital control for CC-mode, CV-mode, Die-temperature protection, Battery-temperature high, Battery- temperature low, Battery- temperature ok, and “End-of-Charge” (when the regulated charge current drops below 10% of the programmed value). Software can always monitor what is going on in the charger, but the function of the charger does not require a time-based software interaction. The charger control can be taken over by software if required at any point of the charging sequence.

A battery temperature sensing function is incorporated, using an NTC which guarantees JEITA compliance. This function determines the temperature zone of the battery. If the battery is too cold or hot, it will automatically disable charging. Thresholds are programmable.

All essential measurements needed for the charger control (Vbat, battery temperature) are implemented in hardware, without the need to use any of the ADC channels and/or SW.

The actual charger HW FSM is illustrated in [Figure 18](#):

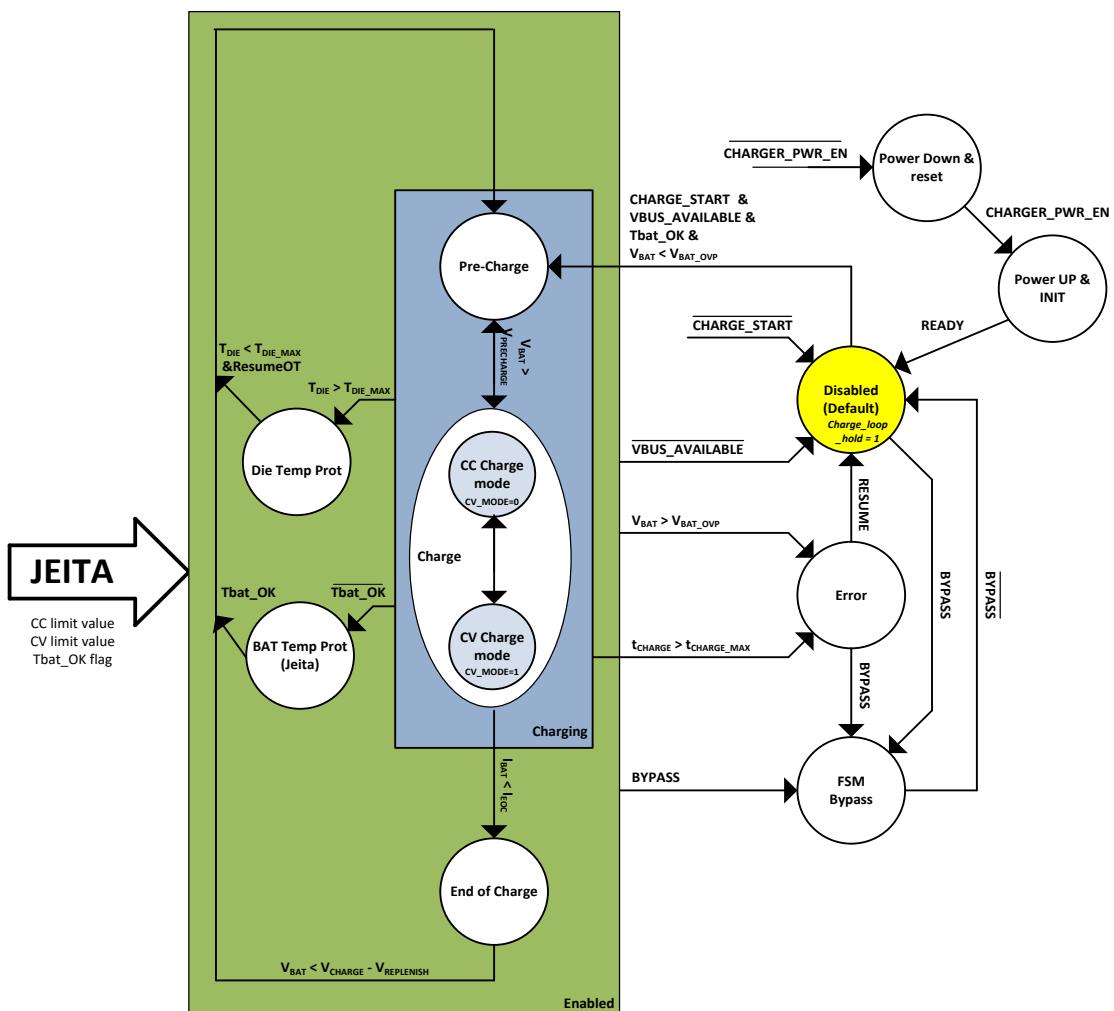


Figure 18: Charger HW Finite State Machine Diagram

7.2.6.1 JEITA

The JEITA standard defines a battery temperature dependent charge profile. In the normal temperature range, the charge current and voltage are at the maximum level for the selected battery. Above and below this normal temperature range, there are two ranges in which the current (and optionally voltage) is reduced. At very low and very high battery temperatures, charging is stopped altogether. The battery temperature is determined using the NTC in the battery pack. This NTC is placed in series with an external resistor and a voltage is applied across them, creating a voltage divider. The output of this divider is compared to an internal reference that is created using a programmable resistor ladder that mimics the behavior of the NTC / resistor combination and has programmable taps at $\sim 1^\circ\text{C}$ intervals, between -10°C and $+55^\circ\text{C}$. The NTC battery monitoring connections are shown in the following figure:

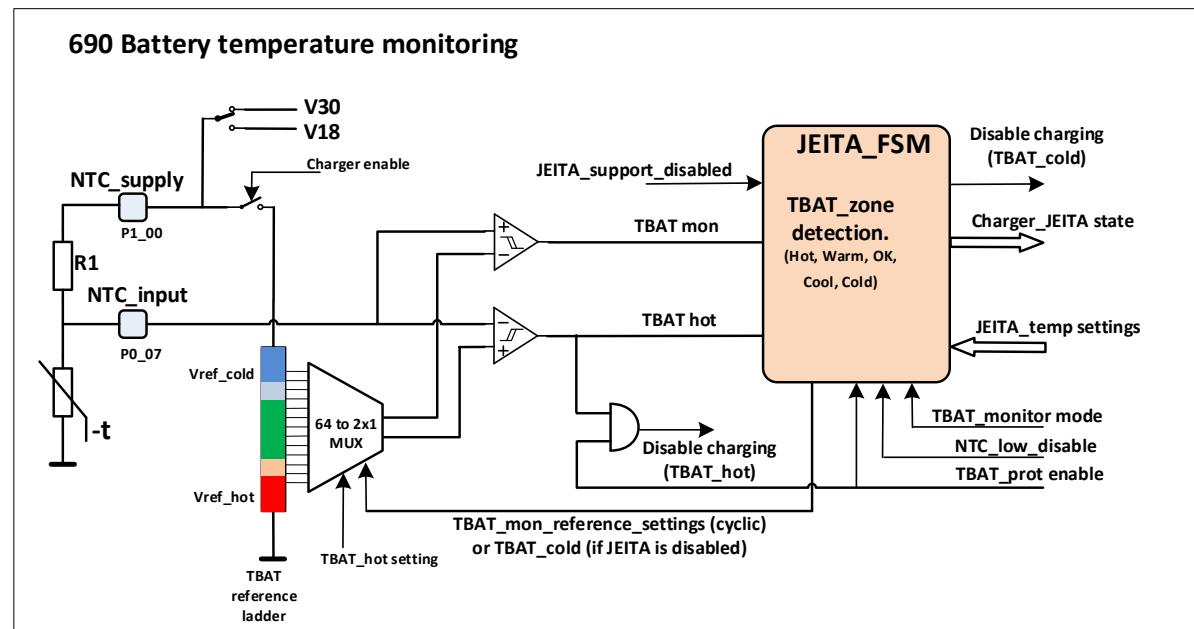


Figure 19: NTC Battery Monitoring Connections

The Battery temperature monitoring FSM (JEITA_FSM) is illustrated in [Figure 20](#):

NOTE

When using P1_00 as the supply for an NTC and P0_07 as the input for an NTC, both pins must be configured as GPIO's (Px_yy_MODE_REG[PID] = 0)

P1_0 must be set as an output driving high and P0_07 as an input without pull-up or pull-down.

Multi-core Bluetooth 5.2 SoC family with system PMU

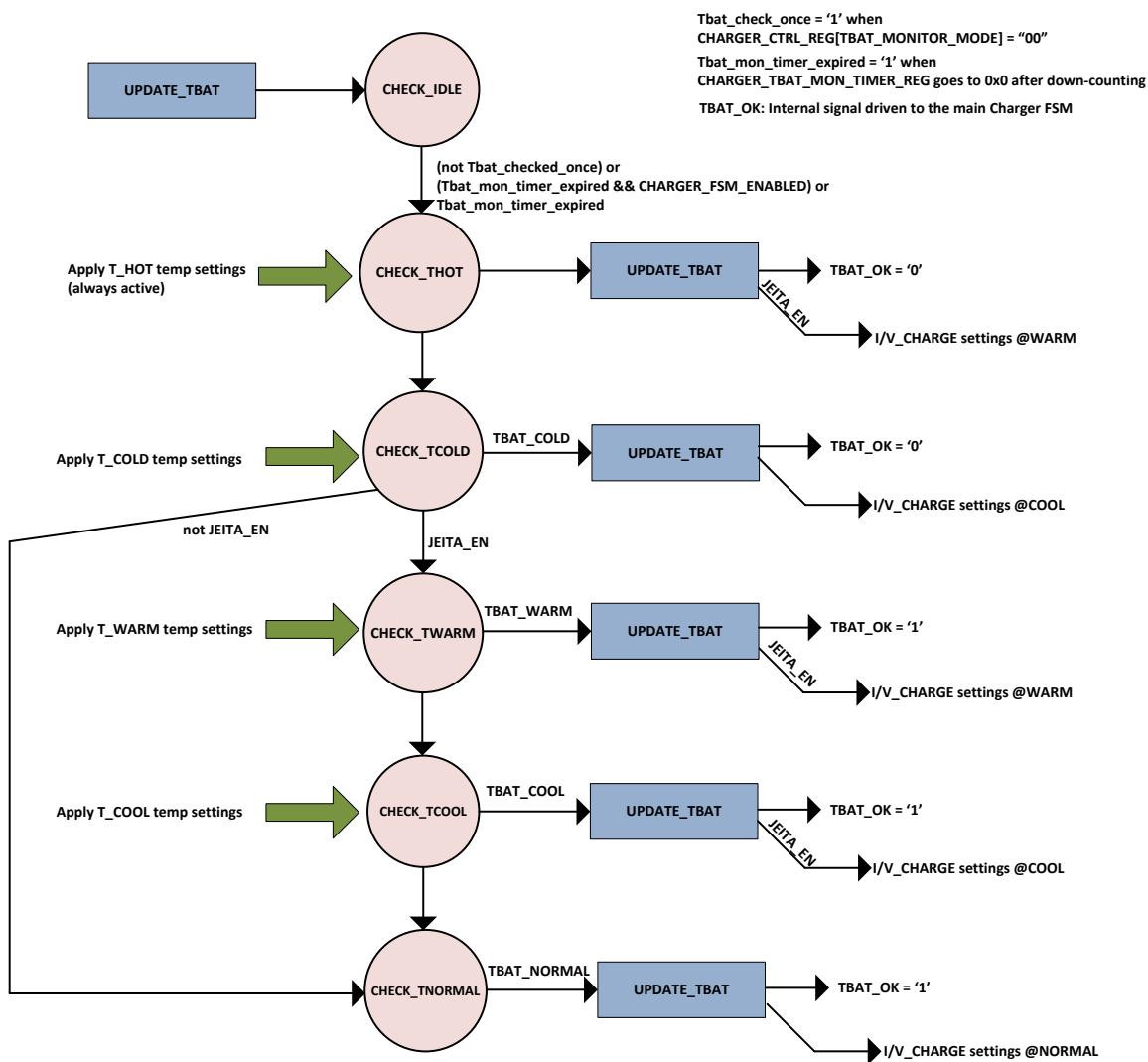


Figure 20: Battery Temperature Monitoring FSM

7.2.6.2 Errors and Flags

The following error conditions can be detected:

- Pre-charge timeout
- CC charge timeout
- CV charge timeout
- Total charge timeout
- Battery over-voltage
- Die temperature protection (error flag but not error state)
- Tbat_HOT & Tbat_COLD: Meaning that the Battery temperature is definitely not OK
- Vbus available: Meaning that VBUS has been removed

The voltage and temperature errors get a direct path to shut down the charger loop.

The error flags are debounced before triggering a state change. Moreover, dedicated error handling routines can start on the assertion of these debounced signals.

Table 79 describes the error flags and when they are asserted:

Table 79: Charger Error Flags

| Error Flag | When Asserted | Description |
|----------------|-------------------------|---|
| VBUS_AVAILABLE | Set if Vbus not present | Direct path to disable loop (part of wired OR with CHARGE_LOOP_HOLD), need for fast response time Needs debounce filter for state change If False, needs to disable charger loop If for some reason the bus is shorted, a high current path from battery to GND will be present, that needs to be blocked as soon as possible Will go to DISABLE state from any charge state (green block in state diagram) |
| VBAT_OVP | Vbat too high | Direct path to disable loop (part of wired OR with CHARGE_LOOP_HOLD), need for fast response time Needs debounce /filter for state change Will stop charging |
| DIE_OVER_TEMP | Set if die too hot | Direct path to disable loop (part of wired OR with CHARGE_LOOP_HOLD), need for fast response time. Debounce for digital. Stop charger, stop Counter set an ERROR_IRQ. Will set CHARGE_LOOP_HOLD Can start additional error routine. Default Resume_OT flag should be set. Resume if temperature is lower again, set by internal threshold in Temp_protect circuit |
| Tbat_HOT | Tbat too high | Default output of battery temp block if not in monitor mode, direct path to disable charger loop |
| Tbat_COLD | Tbat too low | Default output of battery temp block if not in monitor mode, direct path to disable charger loop Charging may continue in Cold zone if CHARGER_CTRL_REG[NTC_LOW_DISABLE] field is set. Otherwise, the Charger's FSM switches to the respective error state (TBAT_PROT), updating also the respective Error IRQ status register |

The flags are debounced with 1 MHz clock.

Upon a timeout or overvoltage or a Vbus problem, the FSM goes to the “Error” state. At this point the charger is disabled, and an IRQ is generated. An interrupt status register contains information on which error occurred to trigger the IRQ.

7.2.6.3 Timers

There are four timers running on a base clock of 1 Hz:

4. Total charge time is up to 10.5 hours; for this reason, it is 16 bits.
5. Pre-charge time is up to 6 hours; for this reason, it is 15 bits.
6. CC mode charge time is up to 6 hours; for this reason, it is 15 bits.
7. CV mode charge time is up to 6 hours; for this reason, it is 15 bits.

Timers will be cleared in EoC and when leaving DISABLED state.

7.2.7 Charger Detection

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | x | ✓ | ✓ | ✓ |

The USB controller has built-in hardware to determine the charger type to which it is connected. Depending on the charger type, battery state and USB connection state, a defined current can be drawn from the charger. The main features of the charger detection circuit are listed below:

- Complies to “Battery Charging Specification” Revision V1.2 December 7, 2010 (BC1.2)
- Charger type detection: Dedicated Charging Port (DCP), Charging Downstream Port (CDP), Standard Downstream Port, PS2 port and Proprietary charger
- Dead battery provision
- Compatible with various smartphone chargers

The details of the charger detection circuit are shown in [Figure 21](#):

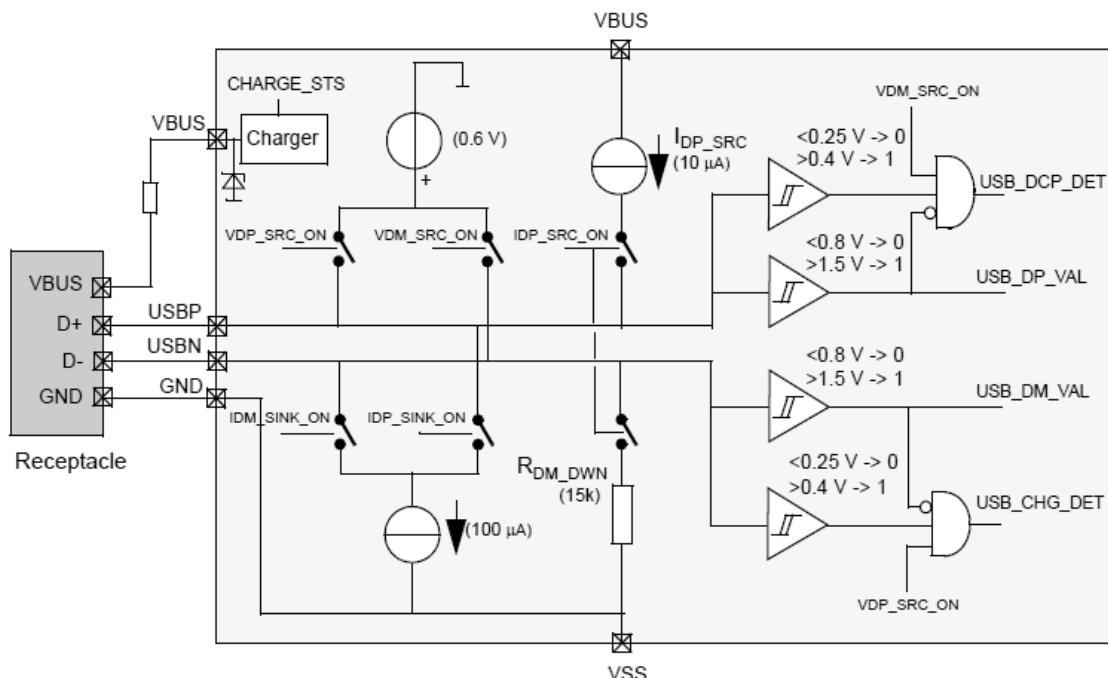


Figure 21: Charger Detection Circuit

The USB interface supports the battery charging with the following hardware blocks as shown in [Figure 21](#).

- A voltage source of 0.6 V can be switched to USBP or USBN with `USB_CHARGER_CTRL_REG` bits `VDP_SRC_ON` resp. `VDM_SRC_ON`
- A current sink of 100 μ A can be switched to USBP or USBN with `USB_CHARGER_CTRL_REG` bits `IDP_SINK_ON` resp. `IDM_SINK_ON`. (Note that internal logic prevents both switches from being enabled at the same time)
- A current source I_{DP_SRC} and R_{DM_DWN} can be enabled with `USB_CHARGER_CTRL_REG[IDP_SRC_ON]`
- A logic level Schmitt trigger is used for USBN, USBP read in `USB_CHARGER_STAT_REG` bits `USB_DM_VAL`, resp. `USB_DP_VAL` logic level (0: <0.8 V 1: >2 V)
- A comparator output `CHG_DET` read in `USB_CHARGER_STAT_REG[USB_CHG_DET]` to detect a level of 0.4 V < USBN < 1.5 V to indicate that a DCP or CDP is connected
- A comparator output `DCP_DET` read in `USB_CHARGER_STAT_REG[USB_DCP_DET]` to detect a level of 0.4 V < USBP < 1.5 V to indicate that a DCP is connected

Multi-core Bluetooth 5.2 SoC family with system PMU

Initially all bits of USB_CHARGER_CTRL_REG must be set to reset value ‘0’.

The presence of VBUS can be checked by reading bit BAT_STATUS_REG[VBUS_AVAILABLE].

The charger detection hardware can operate in polling mode or can generate a USB interrupt to the Cortex®M33. A change in one of the bits [3:0] of register USB_CHARGER_STAT_REG, sets bits USB_MAEV_REG[USB_CH_EV] if the corresponding bits [7:4] are set to ‘1’. If USB_CHARGER_STAT_REG is read, bit USB_CH_EV interrupt is cleared. The interrupt “set” conditions have priority over the “clear” condition of the read access.

7.2.7.1 Contact Detection

If USB_CHARGER_CTRL_REG[IDP_SRC_ON] is set, bit USB_CHARGER_STAT_REG[USB_DP_VAL] indicates that the data pins make contact (see [Table 80](#)). It is the responsibility of the SW to wait until the USBN and USBP contact bouncing has finished before the register is read.

Table 80: USBP, USBN contact detection

| Port | USBP | USBN | USB_DP_VAL |
|---------------------|--------|--------|------------|
| Nothing Connected | >1.5 V | 0 | 1 |
| Standard Downstream | <0.8 V | 0 | 0 |
| Dedicated Charger | <0.8 V | <0.8 V | 0 |
| Charging Downstream | <0.8 V | <0.8 V | 0 |

7.2.7.2 Primary Charger Detection

Primary charger detection is used to detect whether the downstream port has charging capabilities or not. The detection is initiated by setting bits USB_CHARGER_CTRL_REG[VDP_SRC_ON] and USB_CHARGER_CTRL_REG[IDM_SINK_ON]. This enables the voltage source VDP_SRC on USBP and the current source IDM_SINK on USBN. The measured levels on USBP and USBN shown in [Table 81](#) determine the value of bit USB_CHARGER_STAT_REG[USB_CHG_DET].

Table 81: Charger Type Detection

| Port | USBP | USBN | USB_CHG_DET |
|---------------------|-------|--------------------------|-----------------------------|
| Dedicated Charger | 0.6 V | >0.4 V && <1.5 V | 1 |
| Charging Downstream | 0.6 V | First <0.25 V then 0.6 V | 0 then 1 after 1 ms – 20 ms |
| Standard Downstream | 0.6 V | <0.25 V | 0 |
| PS2 | 2 V | 2 V | 0 |

Note that when the charger detection is done before the charging downstream port is enabled its V_{DM_SRC} (so before 20 ms) a standard downstream port is detected, which is safe but incorrect. After the V_{DM_SRC} has been enabled in the charging downstream port, a charger port is detected.

7.2.7.3 Secondary Charger Detection

Secondary charger detection can be used to distinguish a dedicated charger or a charging downstream port. The detection is initiated by setting bits USB_CHARGER_CTRL_REG[VDM_SRC_ON] and USB_CHARGER_CTRL_REG[IDP_SINK_ON]. This enables V_{DM_SRC} and I_{DP_SINK}. The difference between a DCP and a CDP is shown [Table 82](#).

Table 82: Secondary Charger Detection

| Port | USBP | USBN | USB_DCP_DET |
|---------------------|---------------|-------|-------------|
| Dedicated Charger | >0.4 V <1.5 V | 0.6 V | 1 |
| Charging Downstream | <0.25 V | 0.6 V | 0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

7.2.7.4 Smartphone Charger Detection

The battery charger detection circuit is able to detect smartphone chargers with characteristics shown in Table 83.

Table 83: Smartphone Charger Characteristics

| USBP | USBN | Load Current |
|------|-------|--------------|
| 2.0V | 2.0 V | Up to 500 mA |
| 2.0V | 2.8 V | Up to 1 A |
| 2.8V | 2.0 V | Up to 2 A |
| 2.8V | 2.8 V | Up to 2.4 A |

The smartphone charger circuit can be enabled by the USB_CHARGE_ON bit of the USB_CHARGER_CTRL_REG register. The results of the detection will be available on the USB_CHARGER_STAT_REG register.

7.2.8 Rails Discharge

The power rails have a software-controlled discharge capability which practically uses multiple NMOS transistors to rapidly discharge the external decoupling capacitors. This feature actually enables power cycling of the external components when system is resetting.

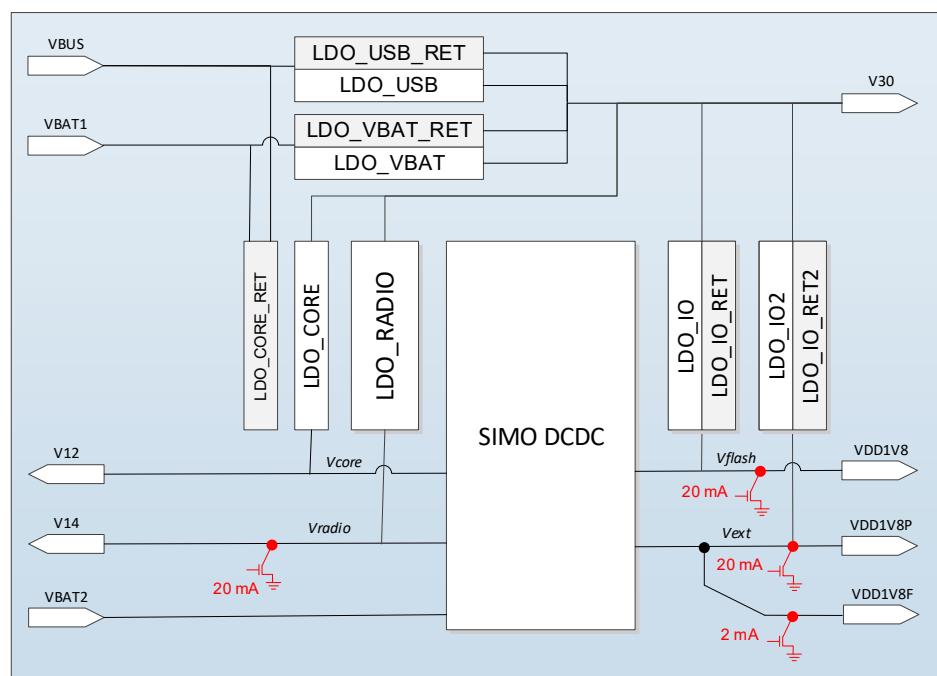


Figure 22: Power rail discharging feature

There is a hardwired configuration for the allowed discharging current of each rail which is either 2, or 20 mA. The configuration for the discharging elements is illustrated in Figure 22.

Note that, triggering of the discharging process can only happen with use of software writing to DISCHARGE_RAIL_REG.

8 Power Domain Controller (PDC)

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

8.1 Introduction

The Power Domain Controller (PDC) is responsible for taking action after waking up or before going to sleep regarding the activation/de-activation of the seven digital power domains of the system. It follows the HW FSM for start-up/wakeup and it precedes the go to sleep HW FSM described in Section [HW FSM](#). It is a digital hardware state machine that defines the following parameters after consulting a programmable look-up table (LUT):

- Which power domain to activate after a wake-up trigger
- If XTAL32M needs to be started after a wake-up trigger
- If the system is allowed to go to deep sleep (trigger the HW FSM to disable DCDC, Bandgap and LDOs)

The PDC can be triggered by any GPIO, general purpose timers, RTC, the MAC timer, the Motor Controller, VBUS or SWD presence, or even by Software a trigger issued by one of the three masters of the system (Cortex-M33, CMAC or SNC).

Features

- 16 places of 13-bit words Look-Up Table (LUT) for defining what the system should do upon any wake-up trigger
- Triggers from IOs, internal timers/controllers, or SW
- Triggers from internal diagnostic signals
- Monitors if a power domain is actively used by any master before shutting it down
- Allows system to go to deep sleep if all masters' domains are off
- System wakeup can be overridden by accessing the PMU_CTRL_REG

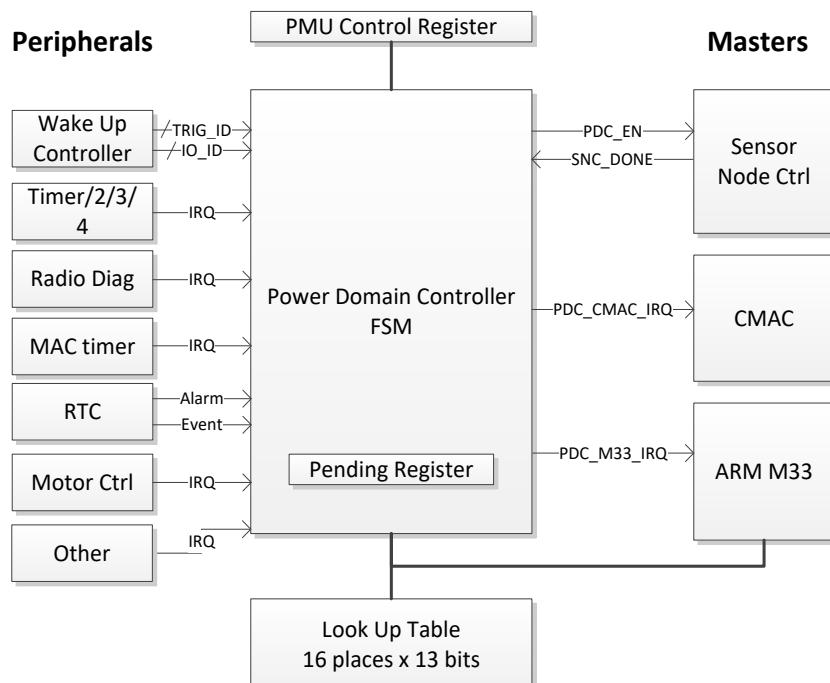


Figure 23: Power Domain Controller Block Diagram

8.2 Architecture

8.2.1 Look-Up Table

The Look-Up Table (LUT) is initialized at cold boot by the Cortex-M33. It is instructing the PDC which digital power domains to activate based on the triggering source.

The format of the LUT is presented in [Table 84](#):

Table 84: PDC LUT format

| | Bit | Function | | |
|----------|-----|--|-------------------------|--------------------------------|
| Triggers | 0 | if 0x00 then P0 GPIO | if 0x01 then P1 GPIO | if 0x02 or 0x3 then Peripheral |
| | 1 | | | |
| | 2 | Pin_ID | Pin_ID | Periph_ID |
| | 3 | Pin_ID | Pin_ID | Periph_ID |
| | 4 | Pin_ID | Pin_ID | Periph_ID |
| | 5 | Pin_ID | Pin_ID | Periph_ID |
| | 6 | Pin_ID | Pin_ID | |
| Action | 7 | Enable XTAL32M | | |
| | 8 | Enable PD_TMR | | |
| | 9 | Enable PD_PER | | |
| | 10 | Enable PD_COM (Needed for using the GPIOs) | | |
| | 11 | Wake up Master_ID | | |
| | 12 | Wake up Master_ID | | |

A look up table of 13 bits width describes what happens on each trigger. The depth of the LUT is 16 places, i.e. the system supports up to 16 different configurations given a specific trigger for waking up, but this could be changed dynamically by application software if needed.

There are 3 different trigger types that will be evaluated when a trigger comes according to the value bits 0 and 1 of each LUT entry:

1. If Bits[1:0]=0x0 then it is a GPIO toggle from P0. Bits [6:2] contain the pin number that has triggered.
2. If Bits[1:0]=0x1 then it is a GPIO toggle from P1. Bits [6:2] contain the pin number that has triggered.
3. If Bits[1:0]=0x2 or 0x3 then it is a trigger from some peripheral. Bits [5:2] define the peripheral according to [Table 85](#):

Table 85: Peripheral Trigger Encoding

| ID | Peripheral |
|----|--------------------|
| 0 | Timer |
| 1 | Timer2 |
| 2 | Timer3 |
| 3 | Timer4 |
| 4 | RTC Alarm/Rollover |
| 5 | RTC Timer |
| 6 | MAC Timer |

| ID | Peripheral |
|-------|---|
| 7 | Motor Controller |
| 8 | XTAL32MDRY_IRQ |
| 9 | RFDIAG_IRQ |
| 10 | VBUS_IRQ or Debounced IO IRQ or JTAG_IRQ or CMAC2SYS_IRQ |
| 11 | Sensor Node Controller IRQ |
| 12-14 | Reserved |
| 15 | Software Trigger |

Bits[12:7] explain what needs to be done upon a trigger from the triggering sources as explained so far. More specifically, a triggering source might request to:

- Enable the XTAL32M. This bit will be set if clock precision is required by the application. Note that, the PDC state machine will only enable the XTAL32MHz block but switching the system clock to the XTAL32M will not be done. This has to be done by software. Since multiple masters might want to switch to XTAL32, switching to this clock can be done by any master but switching back should not be allowed. Turning off the XTAL32M will be done automatically when the system enters deep sleep (e.g. the PDC allows the HW FSM to turn off all LDOs and Bandgap).
- Enable PD_TMR or PD_PER. These digital power domains are the only ones that do not contain a master. PD_MEM will always be enabled since all masters will be using this power domain.
- Enable PD_COM. Although this power domain is controlled by its own master (SNC), it might be requested by others because this power domain contains the GPIO mode configuration. To use GPIOs, the PD_COM must be activated.
- Issue a wake up IRQ/Signal to any other master. Hence a master can wake up another master with use of a LUT entry. The Master ID is encoded as follows:

Table 86: Master Trigger Encoding

| ID | Master |
|-----|------------------------------|
| 0x0 | Reserved |
| 0x1 | Cortex-M33 |
| 0x2 | CMAC |
| 0x3 | Sensor Node Controller (SNC) |

8.2.2 Operation

The PDC will re-evaluate all available information every time there is a trigger input. It will do the same every time there is feedback from one of the three masters indicating that they have finished what they were triggered to do. The latter is implemented using signaling between the masters and the PDC, namely:

- A deep sleep signal coming from Cortex-M33 or CMAC which is asserted when the Cortex-M33 SCR bit is set and the WFI command executed
- A done signal coming from SNC which is asserted when the SLP command is executed

Every time a trigger occurs, the PDC will follow the action plan as programmed in the LUT. It will also assert the respective bit number in the PENDING register that keeps one bit per LUT entry (e.g. if LUT entry #2 was triggered, PENDING[2] will also be asserted). The master that has been woken up, will acknowledge the PENDING bit and after finishing its tasks, it will notify the PDC that any request for power domains activation is not valid anymore. The PDC will cross-check the complete LUT to see if there is any other active entry that still uses any of the power domains requested. If not, these

Multi-core Bluetooth 5.2 SoC family with system PMU

power domains will be shut down. If other masters still use one of the domains, then it will not allow this domain to be powered down.

Note that, the PENDING register should be acknowledged as soon as possible and well before issuing a SLP/WFI command.

If there is no active LUT entry where one of the three masters is still up and running, then the PDC will allow the system to go to deep sleep (i.e. invoke the HW FSM and turn off bandgap and LDOs). Any power domain will be allowed to turn off, depending on the value of the respective field in the PMU_CTRL_REG. Hence, if for example, PMU_CTRL_REG[RADIO_SLEEP] = 1 then given the fact that no LUT entry is active, the power domain will be turned off.

8.3 Programming

There is a simple sequence of steps that needs to be followed to program and use the Power Domain controller:

8. Add a PDC entry by writing the PDC_CTRLx_REG:
 - a. Select Trigger (TRIG_SELECT):
 - i. 0: Trigger is a GPIO on Port 0 (selectable through Wake-Up Controller).
 - ii. 1: Trigger is a GPIO on Port 1 (selectable through Wake-Up Controller).
 - iii. 2: Trigger is a Peripheral IRQ (see register description).
 - iv. 3: Trigger is another Master (see register description).
 - b. Select Trigger ID (TRIG_ID):
Valid when TRIG_SELECT = 0x0, 0x1 or 0x2 (see register description for options).
 - c. Enable extra options if needed (EN_COM, EN_PER, EN_TMR, EN_XTAL).
 - d. Select the master to wakeup (PDC_MASTER):
 - i. 0: PDC entry is disabled.
 - ii. 1: Wake up Arm Cortex-M33.
 - iii. 2: Wake up CMAC.
 - iv. 3: Wake up Sensor Node Controller.
9. Check if a PDC entry has been triggered (PDC_PENDING_REG).
10. If needed, trigger a PDC by SW (PDC_SET_PENDING_REG).
11. Clear any pending requests and/or IRQs (PDC_ACKNOWLEDGE_REG).

9 Brown-Out Detector

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

9.1 Introduction

The brown-out detector (BOD) is a voltage monitoring circuit that triggers a HW reset if LDOs or supply voltages go below a certain threshold.

The BOD is active in either ACTIVE, SLEEP, or DEEP_SLEEP mode with periodically a programmable interval (PMU_SLEEP_REG[BOD_SLEEP_INTERVAL]).

While in DEEP_SLEEP mode, the detector is regularly activated for a voltage comparison of the supply sources after which it goes in DEEP_SLEEP mode again. The amount of time BOD is active is one low power clock period.

Features

- Independent voltage monitoring of seven power rails
- Programmable BOD levels
- Periodic detection mechanism, available during active and sleep periods

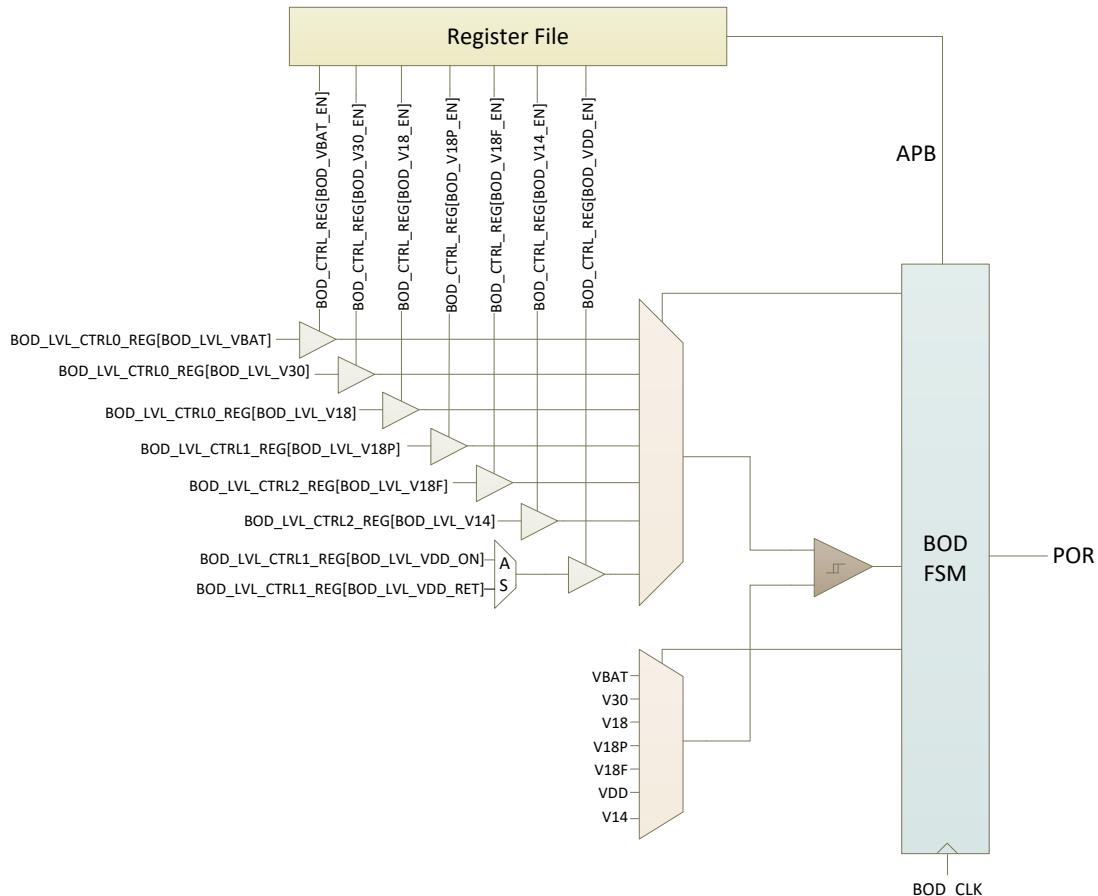


Figure 24: Brown-Out Detector Block Diagram

9.2 Architecture

9.2.1 Brown-Out Detector Monitor Levels

Although the voltage levels are programmable, their reset levels have a value matching their default operating value. The comparator levels are programmable according to formula:

$$\text{BOD_LVL_CTRLx_REG} [\text{BOD_LVL_Vxx}] = 1.2 * (\text{BOD_LVL} + 1)/192$$

For VBAT, a 1.5x scaler is enabled:

$$\text{BOD_LVL_CTRL0_REG} [\text{BOD_LVL_VBAT}] = 1.5 * (1.2 * (\text{BOD_LVL} + 1)/192)$$

Table 87 shows the comparator registers and their default levels (V_{th_bod}) below which, a reset is triggered.

Table 87: Brown-out detectors and default levels

| BOD_LVL_CTRLx_REG | Default Level (V) | Default Level (Hex) |
|-------------------|-------------------|---------------------|
| VBAT | 2.475 | 0xAF |
| V30 | 1.65 | 0x107 |
| V18 | 1.65 | 0x107 |
| V18P | 1.65 | 0x107 |
| V18F | 1.65 | 0x107 |
| V14 | 1.25 | 0xC7 |
| VDD (Active) | 1.05 | 0xA7 |
| VDD (Sleep) | 0.7 | 0x6F |

All individual brown-out detectors are disabled at startup. To enable them, set the corresponding bit in BOD_CTRL_REG[BOD_Vxx_EN].

Note that the BOD levels are checked in a sequential way. The more detectors are enabled, the slower the detection rate is.

The BOD FSM timing, checking up to 7 channels is presented in the following figure:

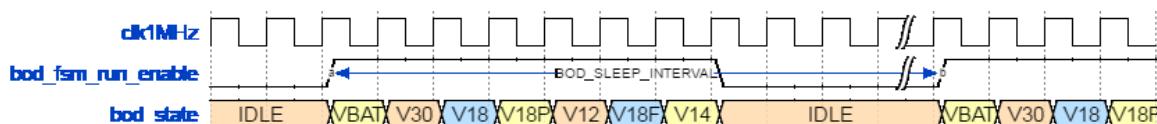


Figure 25: BOD FSM Timing Diagram in Sleep Mode

9.2.2 Brown-Out Detector Clock

The clock of the BOD FSM is 8 MHz in active and 1 MHz in sleep cases. The BOD_SLEEP_INTERVAL is counting down LP_CLK clocks while in sleep mode.

The BOD clock can further be reduced using BOD_CTRL_REG[BOD_CLK_DIV]. A slower clock period, however, reduces the detection of small voltage drops and spikes.

9.3 Programming

There is a simple sequence of steps that needs to be followed to configure the Brown-Out Detector:

1. Setup the interval when Brown Out Detection is activated to check on the power rails voltage (PMU_SLEEP_REG[BOD_SLEEP_INTERVAL]).
2. Configure the BOD voltage levels for the chosen power rails (BOD_LVL_CTRLx_REG).
3. Enable the BOD monitoring for the chosen power rails (BOD_CTRL_REG[BOD_xxxx_EN]).

10 Reset

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

10.1 Introduction

The DA1469x has a RSTN pad which is active low. It contains a RC filter for spikes suppression with 400 kΩ resistor and a 2.8 pF capacitor. It also includes a 25 kΩ pull-up resistor. This pad should be driven externally using a FET or a single button connected to Ground. The typical latency of the RSTN pad is about 2 µs.

Additionally, a configurable Power-on Reset circuitry is included. to allow for a programmable time delayed POR functionality from a configurable reset source. By default, this circuitry is connected to the RSTN pin, but SW can remap it to any GPIO.

A software reset is available. This is done by either programming a specific register or triggering it via the debugger interface.

Features

- RC spike filter on RSTN to suppress external spikes (400 kΩ, 2.8 pF)
- Three different reset lines (SW, HW and POR)
- Reset cause is latched in a specific register
- Configurable POR circuitry

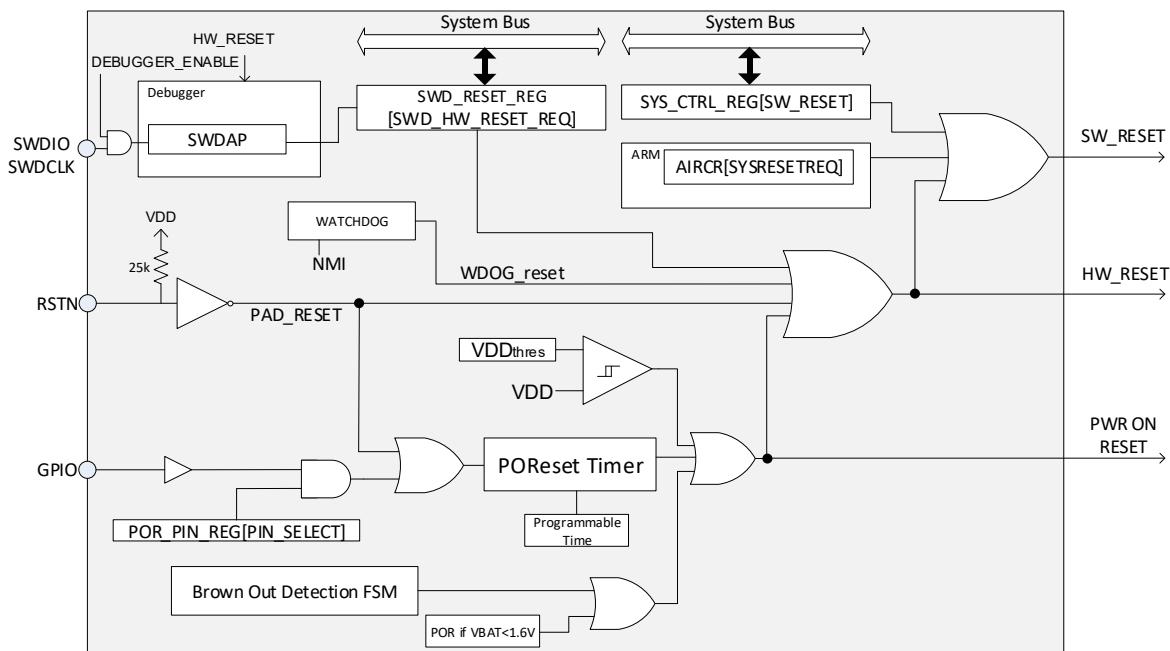


Figure 26: Reset Block Diagram

There are three main reset signals in the DA1469x:

1. The PWR ON reset, triggered by a GPIO set as POR source with selectable polarity and/or the RST pad, after a programmable time delay.
2. The HW reset, triggered by the RST pad when it becomes active for a short period of time (less than the programmable delay for POR).

Multi-core Bluetooth 5.2 SoC family with system PMU

3. The SW reset, triggered by writing the SYS_CTRL_REG[SW_RESET] bit or Arm's AIRCR[SYSRESETREQ] register.

10.2 Architecture

The Power-on Reset (POR) signal is generated as follows:

- Internally; it will release the system's flip flops as soon as the VDD voltage crosses the minimum threshold value
- Brown-Out Detection FSM senses the various internal voltage levels to be higher than the programmed thresholds
- Externally by a Power-on Reset source (RSTN pad or GPIO)

The HW reset can be automatically triggered at system wakeup from the Extended or Deep Sleep mode, by programming bit PMU_CTRL_REG[RESET_ON_WAKEUP]. The PWR ON reset and the HW reset runs the cold start-up sequence and the BootROM code is executed.

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

Certain registers are reset by Power-on Reset only. These registers are listed in [Table 88](#).

Table 88: Reset Signals and Registers

| Reset Source | Registers |
|--------------|---|
| POR Reset | SYS_STAT_REG CLK_XTAL32K_REG CLK_RTC DIV_REG BANDGAP_REG POR_VBAT_CTRL_REG POR_PIN_REG POR_TIMER_REG RESET_STAT_REG[PORESET_STAT] RTC_CONTROL_REG RTC_KEEP_RTC_REG |
| HW Reset | BOD_CTRL_REG BOD_LVL_CTRL0_REG BOD_LVL_CTRL1_REG BOD_LVL_CTRL2_REG CACHE_ASSOCCFG_REG CACHE_CTRL1_REG CACHE_CTRL2_REG CACHE_FLASH_REG CACHE_LNSIZECFG_REG CLK_AMBA_REG CLK_CTRL_REG CLK_FREQ_TRIM_REG CLK_RADIO_REG CLK_RC32K_REG CLK_RC32M_REG CLK_RX_REG DCDC_CTRL1_REG DCDC_CTRL2_REG DCDC_IRQ_CLEAR_REG DCDC_IRQ_MASK_REG DCDC_IRQ_STATUS_REG DCDC_STATUS1_REG DCDC_STATUS2_REG DCDC_STATUS3_REG |

| Reset Source | Registers |
|--------------|---|
| | DCDC_STATUS4_REG DCDC_TEST_REG DCDC_V14_REG DCDC_V18_REG DCDC_V18P_REG DCDC_VDD_REG DEBUG_REG LDO_VDDD_HIGH_CTRL_REG OTPC_MODE_REG OTPC_PADDR_REG OTPC_PWORD_REG OTPC_STAT_REG OTPC_TIM1_REG OTPC_TIM2_REG P0_PAD_LATCH_REG P0_RESET_PAD_LATCH_REG P0_SET_PAD_LATCH_REG P1_PAD_LATCH_REG P1_RESET_PAD_LATCH_REG P1_SET_PAD_LATCH_REG PDC_ACKNOWLEDGE_REG PDC_CTRL0_REG PDC_CTRL1_REG PDC_CTRL10_REG PDC_CTRL11_REG PDC_CTRL12_REG PDC_CTRL13_REG PDC_CTRL14_REG PDC_CTRL15_REG PDC_CTRL2_REG PDC_CTRL3_REG PDC_CTRL4_REG PDC_CTRL5_REG PDC_CTRL6_REG PDC_CTRL7_REG PDC_CTRL8_REG PDC_CTRL9_REG PDC_PENDING_CM33_REG PDC_PENDING_CMAC_REG PDC_PENDING_REG PDC_PENDING_SNC_REG PDC_SET_PENDING_REG PLL_SYS_CTRL1_REG PLL_SYS_CTRL2_REG PLL_SYS_STATUS_REG PMU_CTRL_REG PMU_SLEEP_REG PMU_TRIM_REG POWER_CTRL_REG QSPIC_BURSTBRK_REG QSPIC_BURSTCMDA_REG QSPIC_BURSTCMDB_REG QSPIC_CHCKERASE_REG QSPIC_CTR_CTRL_REG QSPIC_CTR_EADDR_REG QSPIC_CTR_KEY_0_3_REG QSPIC_CTR_KEY_12_15_REG QSPIC_CTR_KEY_16_19_REG QSPIC_CTR_KEY_20_23_REG QSPIC_CTR_KEY_24_27_REG QSPIC_CTR_KEY_28_31_REG QSPIC_CTR_KEY_4_7_REG QSPIC_CTR_KEY_8_11_REG |

| Reset Source | Registers |
|----------------|---|
| | QSPIC_CTR_NONCE_0_3_REG QSPIC_CTR_NONCE_4_7_REG QSPIC_CTR_SADDR_REG QSPIC_CTRLBUS_REG QSPIC_CTRLMODE_REG QSPIC_DUMMYDATA_REG QSPIC_ERASECMDA_REG QSPIC_ERASECMDB_REG QSPIC_ERASECTRL_REG QSPIC_GP_REG QSPIC_READDATA_REG QSPIC_RECVDATA_REG QSPIC_STATUS_REG QSPIC_STATUSCMD_REG QSPIC_WRITEDATA_REG QSPIC2_AWRITECMD_REG QSPIC2_BURSTBRK_REG QSPIC2_BURSTCMDA_REG QSPIC2_BURSTCMDB_REG QSPIC2_CHCKERASE_REG QSPIC2_CTRLBUS_REG QSPIC2_CTRLMODE_REG QSPIC2_DUMMYDATA_REG QSPIC2_ERASECMDA_REG QSPIC2_ERASECMDB_REG QSPIC2_ERASECTRL_REG QSPIC2_MEMBLEN_REG QSPIC2_READDATA_REG QSPIC2_RECVDATA_REG QSPIC2_STATUS_REG QSPIC2_STATUSCMD_REG QSPIC2_WRITEDATA_REG RAM_PWR_CTRL_REG RESET_STAT_REG SECURE_BOOT_REG SWD_RESET_REG SYS_CTRL_REG TRIM_CTRL_REG USB_RXC2_REG WATCHDOG_CTRL_REG WATCHDOG_REG |
| Watchdog Reset | RESET_STAT_REG[CMAC_WDOGRESET_STAT] RESET_STAT_REG[WDOGRESET_STAT] |
| SW Reset | The rest of the Register File |

10.2.1 Power-On Reset From Pin

The Power-on Reset function can be triggered at timer expiration. It is available at two sources:

1. Reset Pad (RSTN): Reset pad is always capable of producing a Power-on Reset.
2. GPIO Pin: A GPIO can be selected by the user application to act as POR source.

The POR_TIMER_REG configures the time needed for the PORReset signal to be active. by. The register field POR_TIME is a 7-bits field (maximum value is 0x7F), which holds a multiplication factor for counting RC32K clock periods. This is explained in the following formula:

Total time for POR = POR_TIME x 4096 x RC32k clock period,

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

The maximum time for issuing a POR is ~16.2 seconds at 25 °C, while the default value is ~3 seconds (POR_TIME=0x18).

Multi-core Bluetooth 5.2 SoC family with system PMU

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

$$T_{POR\text{cold}} = 13 \text{ s}$$

$$T_{POR\text{hot}} = 20.8 \text{ s}$$

The Power-on Reset timer is clocked by the RC32k clock. If the application disables the RC32k, then hardware takes care of re-enabling the RC32k clock. Note that, if POR is generated from the Reset pad, RC32k will operate with the default (reset) trim value. If a GPIO is used as a POR source, the RC32k clock will be trimmed. The timing deviation between both cases is expected to be minor.

The same clock source (RC32k) is used as an input to the watchdog timer. Hence, even if the application has accidentally disabled the clock, hardware will start it again and the watchdog will still run.

When a GPIO is used as a Power-on Reset 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 over 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 Active High.

The operation of the Power-on Reset for both Reset pad and GPIO is depicted in [Figure 27](#).

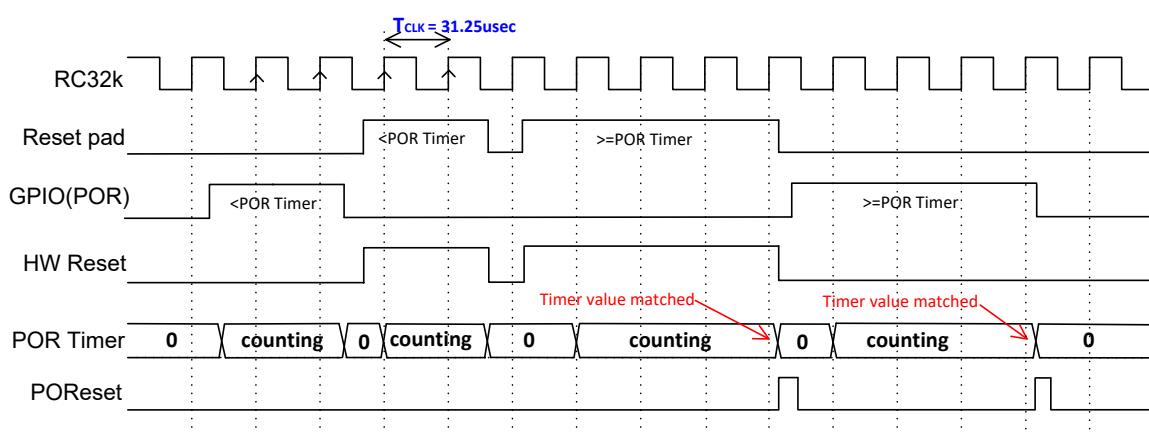


Figure 27: Power-on Reset timing diagram

If any of the POR sources is asserted, then the POR timer starts to count. When a POR source is released before the timer has expired, POR timer will reset to 0. If a second source is asserted while the first is already asserted and the first is released after that point, POR will occur; assuming that the total time of both sources kept asserted is larger or equal than the POR_TIME.

The POR_PIN_REG[PIN_SELECT] field cannot survive any Reset (POR, HW, SW) hence the user must take special care on setting up the GPIO POR source right after a reset. This also applies for the POR_TIMER_REG[POR_TIME] field after a Power-on Reset.

Users should consider that if a GPIO is used as 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 ~100 nA and it is also present during sleep time period. POR from Reset pin does not add this dynamic current consumption.

10.3 Programming

There is a simple sequence of steps that needs to be followed to configure the POR from GPIO functionality:

4. Select the GPIO to be set as POR source (POR_PIN_REG[POR_PIN_SELECT]).
5. Setup the input polarity of the GPIO that causes POR (POR_PIN_REG[POR_PIN_POLARITY]).
6. Configure the time for the POR to happen (POR_TIMER_REG[POR_TIME]). Default time is ~3 seconds.

Multi-core Bluetooth 5.2 SoC family with system PMU

Remember that to set up the time that the Reset pin produces a POR, only POR_TIMER_REG has to be set.

11 Arm Cortex-M33

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

11.1 Introduction

The Cortex-M33 processor is a low gate count, highly energy efficient processor that is intended for microcontroller and deeply embedded applications. The processor is based on the Armv8-M architecture and is primarily for use in environments where security is an important consideration. The interfaces that the processor supports include:

- Code AHB (C-AHB) interface
- System AHB (S-AHB) interface
- External PPB (EPPB) APB interface
- Debug AHB (D-AHB) interface

Arm Cortex-M33 will not retain its status when going to any sleep modes that switch off its power domain. So the CPU will always wake up in reset. Restoration of the CPU state is done by SW.

Features

- Supports the Armv8-M Main Extension. The processor has optional support for one or more of the following extensions:
 - The Floating-point Extension
 - The Digital Signal Processing (DSP) Extension
 - The Debug Extension
- An in-order issue pipeline
- Thumb-2 technology
- Configurable to perform data accesses as either big or little endian
- Nested Vectored Interrupt Controller (NVIC)
- Floating Point Unit (FPU) supporting single-precision arithmetic
 - Combined multiply-add instructions for increased precision (Fused MAC)
 - Hardware support for conversion, addition, subtraction, multiplication with optional accumulate, division, and square-root
 - Hardware support for denormals and all IEEE Standard 754-2008 rounding modes
 - 32 32-bit single-precision registers or 16 64-bit double-precision registers
 - Lazy floating-point context save
- Support for exception-continuable instructions
- Micro Trace Buffer (MTB)

11.2 Architecture

11.2.1 Interrupts

This section lists all 40 interrupt lines, except the NMI interrupt, and describes their source and functionality. The overview of the interrupts is illustrated in the following table:

Table 89: Interrupt List

| # | Name | Type | Polarity | Description |
|---|---------|-------|-------------|---------------------------------------|
| 0 | SNC_IRQ | Level | Active High | Sensor Node Controller interrupt line |
| 1 | DMA_IRQ | Pulse | Active High | General Purpose DMA interrupt line |

Multi-core Bluetooth 5.2 SoC family with system PMU

| # | Name | Type | Polarity | Description |
|----|----------------------|-------------|-------------|---|
| 2 | CHARGER_STATE_IRQ | Pulse | Active High | Serves both the Charger FSM as well as the JEITA FSM |
| 3 | CHARGER_ERROR_IRQ | Pulse | Active High | Charger error interrupt line |
| 4 | CMAC2SYS_IRQ | Level | Active High | CMAC and mailbox interrupt line |
| 5 | UART_IRQ | Level | Active High | UART interrupt line |
| 6 | UART2_IRQ | Level | Active High | UART2 interrupt line |
| 7 | UART3_IRQ | Level | Active High | UART3 interrupt line |
| 8 | I2C_IRQ | Level | Active High | I2C interrupt line |
| 9 | I2C2_IRQ | Level | Active High | I2C2 interrupt line |
| 10 | SPI_IRQ | Level | Active High | SPI interrupt line |
| 11 | SPI2_IRQ | Level | Active High | SPI2 interrupt line |
| 12 | PCM_IRQ | Pulse | Active High | PCM interrupt line |
| 13 | SRC_IN_IRQ | Level/Pulse | Active High | SRC input interrupt line. Level if FIFO is enabled |
| 14 | SRC_OUT_IRQ | Level/Pulse | Active High | SRC output interrupt line. Level if FIFO is enabled |
| 15 | USB_IRQ | Level | Active High | USB interrupt line |
| 16 | TIMER_IRQ | Level | Active High | TIMER interrupt line |
| 17 | TIMER2_IRQ | Level | Active High | TIMER2 interrupt line |
| 18 | RTC_IRQ | Level | Active High | RTC interrupt line |
| 19 | KEY_WKUP_GPIO_IRQ | Level | Active High | Debounced button press interrupt. This interrupt is first driven to the PDC and then directed to the required masters to be waken up/notified |
| 20 | PDC_M33_IRQ | Level | Active High | This is an interrupt coming from the PDC indicating that the Cortex-M33 needs to be waken up due to a GPIO/Peripheral/other master request. |
| 21 | VBUS_IRQ | Level | Active High | VBUS presence interrupt |
| 22 | MRM_IRQ | Pulse | Active High | Cache miss rate monitor interrupt |
| 23 | MOTOR_CONTROLLER_IRQ | Level | Active High | MOTOR controller interrupt line |
| 24 | TRNG_IRQ | Pulse | Active High | True Random Number Generation interrupt. |
| 25 | DCDC_IRQ | Pulse | Active High | DCDC interrupt. Generated upon time out threshold reach. |
| 26 | XTAL32MDRY_IRQ | Pulse | Active High | Indicates that XTAL32M oscillator is trimmed and settled and can provide a reliable 32 MHz clock |
| 27 | GPADC_IRQ | Level | Active High | General Purpose analog-digital converter interrupt |
| 28 | SDADC_IRQ | Level | Active High | Sigma-Delta analog-digital converter interrupt |
| 29 | CRYPTO_IRQ | Level | Active High | Crypto interrupt. Sources: AES or HASH function interrupt |
| 30 | CAPTIMER1_IRQ | Pulse | Active High | GPIO triggered Timer1 Capture interrupt |
| 31 | RFDIAG_IRQ | Pulse | Active High | Baseband or Radio Diagnostics Interrupt. |
| 32 | LCD_CONTROLLER_IRQ | Pulse | Active High | Parallel LCD Controller interrupt line |

Multi-core Bluetooth 5.2 SoC family with system PMU

| # | Name | Type | Polarity | Description |
|----|--------------|-------|-------------|---|
| 33 | PLL_LOCK_IRQ | Level | Active High | Indicates that DLL/PLL is locked at 96MHz |
| 34 | TIMER3_IRQ | Level | Active High | TIMER3 interrupt line |
| 35 | TIMER4_IRQ | Level | Active High | TIMER4 interrupt line |
| 36 | LRA_IRQ | Pulse | Active High | LRA/ERM interrupt line |
| 37 | RTC_EVENT | Level | Active High | RTC event interrupt line |
| 38 | GPIO_P0_IRQ | Level | Active High | GPIO port 0 toggle interrupt line |
| 39 | GPIO_P1_IRQ | Level | Active High | GPIO port 1 toggle interrupt line |

11.2.2 Reference

The register descriptions for the Nested Vectored Interrupt Controller (NVIC), the System Control Block (SCB) and the System Timer (SysTick) of the Arm Cortex-M33 can be found in the following documents, available on the Arm website:

Devices Generic User Guide:

<https://developer.arm.com/docs/100235/latest/preface>

Technical Reference Manual:

<https://developer.arm.com/docs/100230/0004>

12 Cache Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

12.1 Introduction

The cache controller is used to accelerate the system performance of the Arm Cortex-M33 executing from QSPI FLASH. It also conserves power by reducing the access of the external QSPI FLASH. The cache dynamically loads program and data code into the cache Data RAM and executes from there.

The cache controller is controlled via the CACHE_*_REGs. The cache administration is kept in TAG memory. This memory can be invalidated by asserting the FLUSH bit in the CACHE_CTRL1_REG. The Arm Cortex-M0+ is halted during such an invalidation; it resumes automatically. N-way associative replacement strategy is based on the value of a pseudo random LFSR.

The cache controller does not support a runtime configuration of cache line size and associativity. The selection of the configurations depends on the code type and application; it is determined empirically.

For debugging, the Data and TAG memory can be monitored on the AHB-SYS bus (See memory map). The cache is used for dynamic code and data caching. As an alternative for fast code executions, the data-RAM can be used for static code storage. This code must be copied from QSPI FLASH.

Features

- Cacheable range of up-to 32 Mbyte starting from QSPI start address, adjustable length up to $N \times 64\text{kByte}$
- Cache size configurable, default is 16 kB, TAG RAM size is 4 kB
- Runtime configurable cache line 8, 16 or 32 bytes
- Runtime configurable 1, 2 or 4-way associativity
- Built-in TAG memory invalidation (FLUSH)
- Random number (LFSR) for 2, 4-way replacement strategy
- Cache Data and TAG monitoring
- Instruction and Data caching upon read access, no write path to cache available
- Bypass mode
- Cache internal latency:
 - Zero wait cycle for cache hits for same cache line
 - One wait cycle for cache hits when changing cache line
 - $4 + (\text{cache line size}/4 \text{ cycles})$ for cache misses
 - Zero cycle in transparent bypass mode

Multi-core Bluetooth 5.2 SoC family with system PMU

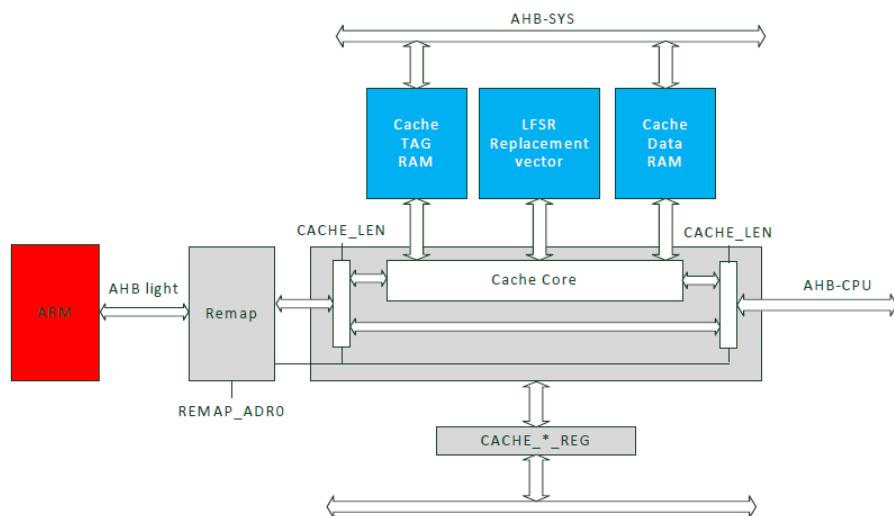


Figure 28: Cache Controller Block Diagram

12.2 Architecture

12.2.1 Cacheable Range

The cache controller caches address range 0-0xFFFFFFFF (32 MB). If REMAP_ADR0=0x1 or 0x2, all addresses from 0 to CACHE_LEN will be cached, else the cache controller automatically asserts the bypass mode. The bypass mode can be forced for all addresses by setting CACHERAM_MUX =0.

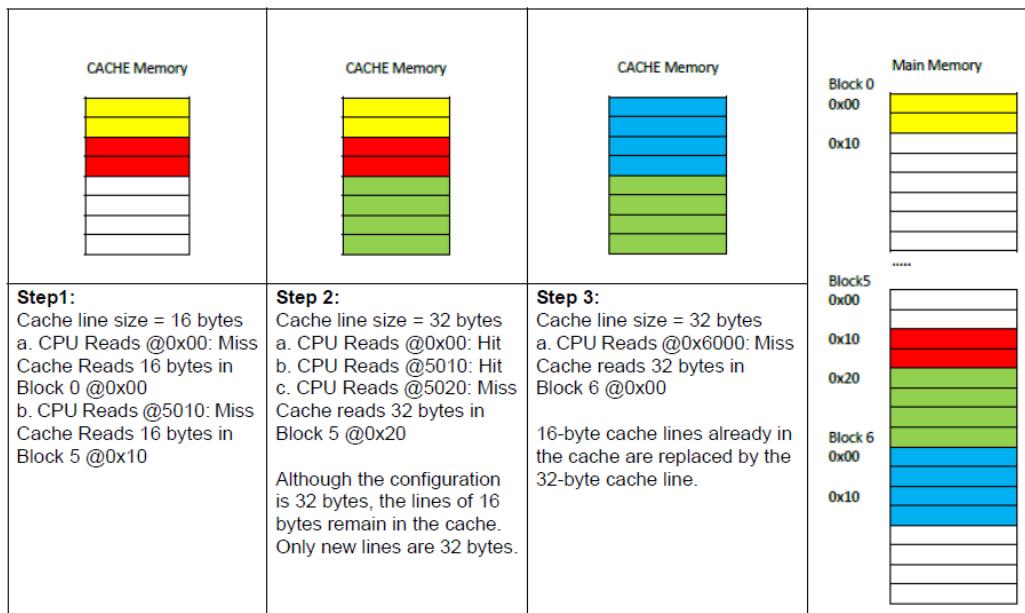
Note that the CACHE_LEN setting is only applicable for the QSPI FLASH remap case as defined in the CACHE_FLASH_REG.

12.2.2 Runtime Reconfiguration

Associativity and cache line size of cache can always be reconfigured by writing the CACHE_ASSOCCFG_REG or CACHE_LNSIZECFG_REG registers. Reconfiguration is done without wait state and without flushing the cache. All the data available in the cache memory are kept except when the associativity is reduced (4-way => 2-way and 2-way => 1-way). In that case, typically half of the data is inaccessible.

12.2.2.1 Cache Line Reconfiguration

The dynamic configuration of the cache line size uses a physical line size of 8 bytes. When the cache line size is defined as 16 bytes or 32 bytes, 2 or 4 physical lines are involved. Reconfiguration of the cache line occurs only when lines are replaced. Even if the cache line configuration is set to 32 bytes, cache lines of 16 bytes may remain in the cache memory. This is explained by means of example in the [Table 90](#):

Table 90: Cache Line Size Reconfiguration Example

12.2.2.2 TAG Memory Word

The administration memory word decoding is presented in [Table 91](#):

Table 91: TAG Memory Layout

| Bit 23 | Bits 22:2 | Bits 1:0 |
|--------|-------------|---|
| 0 | AHB address | 00: invalid cache line 01: 8 valid bytes 10: 16 valid bytes 11: 32 valid bytes |

12.2.2.3 Associativity Reconfiguration

To enable associativity reconfiguration, the cache memory is organized into four banks. Depending on the associativity, the four banks can operate as 4-way, or be concatenated (linked together), resulting in either a 2-way or 1-way cache.

12.2.3 Replacement Strategy

The cache controller fills each line of the cache first, starting from way-0 to way-3. When a line is completely full, a new way-x victim is chosen in a pseudo random way. When a replacement is required, the cache controller reads the value generated by a pseudo-random number generator to select which way to replace. The pseudo-random number generator is realized using a Linear Feedback Shift Register (LFSR).

12.2.4 Cache Reset

The cache controller has two reset signals connected:

- The HW_RESET: When this reset is activated, all cache logic and registers are reset to their default values, while all data in the TAG memories are cleared and all CACHE_*_REG are set to their reset values. It takes around 450 AHB cycles to clear the cache TAG memory. If a fetch

Multi-core Bluetooth 5.2 SoC family with system PMU

request occurs during this reset period, the request will be considered at the end of the reset and wait-states will be inserted.

- The SW_RESET: Upon a SW_RESET the cache state machine and TAG memories are reset, but the CACHE_*_REG are not affected and will remain as programmed.

12.2.5 Miss Rate Monitor

The system includes a cache miss, which monitors circuitry that gives real time information on the number of cache misses within a certain amount of time. When a programmable threshold is reached, an interrupt is sent to the CPU to act. The CPU can dynamically change the cache line size, the associativity, or start the PLL to decrease cache line fetch time, and consequently power. It can even apply a combination of these techniques to adjust the system's parameters accordingly. This block only operates while the system is in active mode. The main features are:

- Up to 10 ms active time interval counter
- Registered amount of cache misses
- Registered amount of cache hits
- Programmable threshold of cache misses that generates an interrupt.

The CACHE_MRM_HITS_REG contains the amount of cache hits. The CACHE_MRM_MISSES_REG contains the number of misses counted within the time interval programmed at the CACHE_MRM_TINT_REG in CPU clock cycles.

12.2.6 Miss Latency and Power

This section describes the amount of time (in clock cycles) required from a cache miss up to the point the required code/data are fetched back to the CPU and execution continues. The cache miss latency (T_{CM2L}) can be split into the following intervals:

T_{CM2R} : Time from the cache miss up to request from the QSPI Controller.

T_{R2QA} : Time from request up to actual access start.

T_{RDRL} : Time for reading data from the FLASH.

T_{CLAT} : Time required to get data to the CPU (cache latency).

The final amount of clock cycles is calculated by the following equation:

$$T_{CM2L} = T_{CM2R} + T_{R2QA} + T_{RDRL} + T_{CLAT}$$

where T_{RDRL} depends on the amount of data requested and is provided by the following formula:

$$T_{RDRL} = T_{CMD} + T_{ADDR} + T_{DUM} + (N_{CACHELINE} * 2) + T_{PIPE}$$

For example, give a cache line configuration of 16 bytes, the amount of clock cycles required to read the data from the FLASH is:

$$T_{RDRL} = 2 + 8 + 4 + (16 * 2) + 1 = 47 \text{ clock cycles.}$$

An overview of the cache miss latency calculation is shown in [Table 92](#):

Table 92: QSPI FLASH Cache Miss Latency

| Time Interval | Clock Cycles | Example |
|---------------|---|---------|
| T_{CM2R} | 3 | 3 |
| T_{R2QA} | 2 | 2 |
| T_{RDRL} | $T_{CMD} + T_{ADDR} + T_{DUM} + (N_{CACHELINE} * 2) + T_{PIPE}$ | 47 |
| T_{CLAT} | 4 | 4 |
| T_{CM2L} | for 16 bytes cache line (QPI mode) | 56 |

12.3 Programming

12.3.1 Cache Controller Programming

There is a simple sequence of steps that needs to be followed to configure the Cache Controller:

1. Disable the Cache by clearing the CACHE_CTRL2_REG[CACHE_LEN] bit field.
 2. Setup the Cache RAM size by programming
CACHE_CTRL_3_REG[CACHE_RAM_SIZE_RESET_VALUE]
 3. Setup the Cache line size (CACHE_LNSIZECFG_REG). Default is 8 bytes.
 4. Setup the Cache associativity (CACHE_ASSOCCFG_REG). Default is 4-way associativity.
 5. Flush Cache contents (CACHE_CTRL1_REG[CACHE_FLUSH]).
 6. Enable the cache (CACHE_CTRL2_REG[CACHE_LEN] != 0).
- The size of the cacheable QSPI Flash memory can be specified in
CACHE_CTRL2_REG[CACHE_LEN] when CACHE_CTRL2_REG[CACHE_LEN] != 1. If
CACHE_CTRL2_REG [CACHE_LEN] = 1, then the memory space is specified by
CACHE_FLASH_REG [FLASH_REGION_OFFSET].

12.3.2 Miss Rate Monitor Programming

There is a simple sequence of steps that needs to be followed to configure the Miss Rate Monitor:

1. Freeze all counters by clearing the CACHE_MRM_CTRL_REG[MRM_START] bit.
2. Setup the thresholds that produce interrupts:
 - a. Cache Misses threshold: CACHE_MRM_MISSES_THRES_REG[MRM_MISSES_THRES]
 - b. Cache Hits threshold: CACHE_MRM_HITS_THRES_REG[MRM_HITS_THRES]
 - c. Time passed threshold: CACHE_MRM_TINT_REG[MRM_TINT]
3. Unmask all interrupts by setting the CACHE_MRM_CTRL_REG[MRM_IRQ_MASK] bit.
4. Enable the chosen interrupts:
 - a. CACHE_MRM_CTRL_REG[MRM_IRQ_HITS_THRES_STATUS] : The number of cache hits reached the programmed threshold.
 - b. CACHE_MRM_CTRL_REG[MRM_IRQ_MISSES_THRES_STATUS] : The number of cache misses reached the programmed threshold.
 - c. CACHE_MRM_CTRL_REG[MRM_IRQ_TINT_STATUS] : The time interval counter reached the end.
5. Enable all counters by setting the CACHE_MRM_CTRL_REG[MRM_START] bit.
6. Read the results (misses and/or hits) in the CACHE_MRM_MISSES_REG and CACHE_MRM_HITS_REG registers.

13 Bus

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

13.1 Introduction

The DA1469x is equipped with a multi-layer AMBA bus, which enables parallel data paths between different masters and slaves.

The bus matrix comprises three main busses:

1. AHB-CPUC bus where the Cache is master. This is the primary bus from which code is executed. Memory map range is 0x00000000- 0x1FFFFFFF.
2. AHB-CPUS bus, where the Cortex-M33 is master. This is the bus for the system. The memory map range is 0x20000000- 0xDFFFFFFF and beyond 0xE0100000.
3. AHB-DMA bus where the RFMON, the Generic DMA, the LCD Controller, and the AES/HASH can be masters.

There are several slaves, sitting behind interconnection multiplexers (ICMs), that allow access from both AHB busses; namely:

- QSPI FLASH memory controller
- QSPI RAM memory controller
- 32-bit APB peripheral registers
- The RAM controller
- The OTP controller
- The AHB register file containing registers for the QSPI FLASH/RAM controller, the CMAC, the Crypto the OTPC, etc.

Features

- Enables parallelization of data transfers from:
 - Peripherals to memory (GP DMA),
 - Cortex-M33 data read/writes from RAM and
 - Cortex-M33 executing code from QSPI FLASH
- Supports parallel accessing of Sensor Node, GP DMA, MACCPU, and SYSCPU on (different) memory segments without collisions.

Multi-core Bluetooth 5.2 SoC family with system PMU

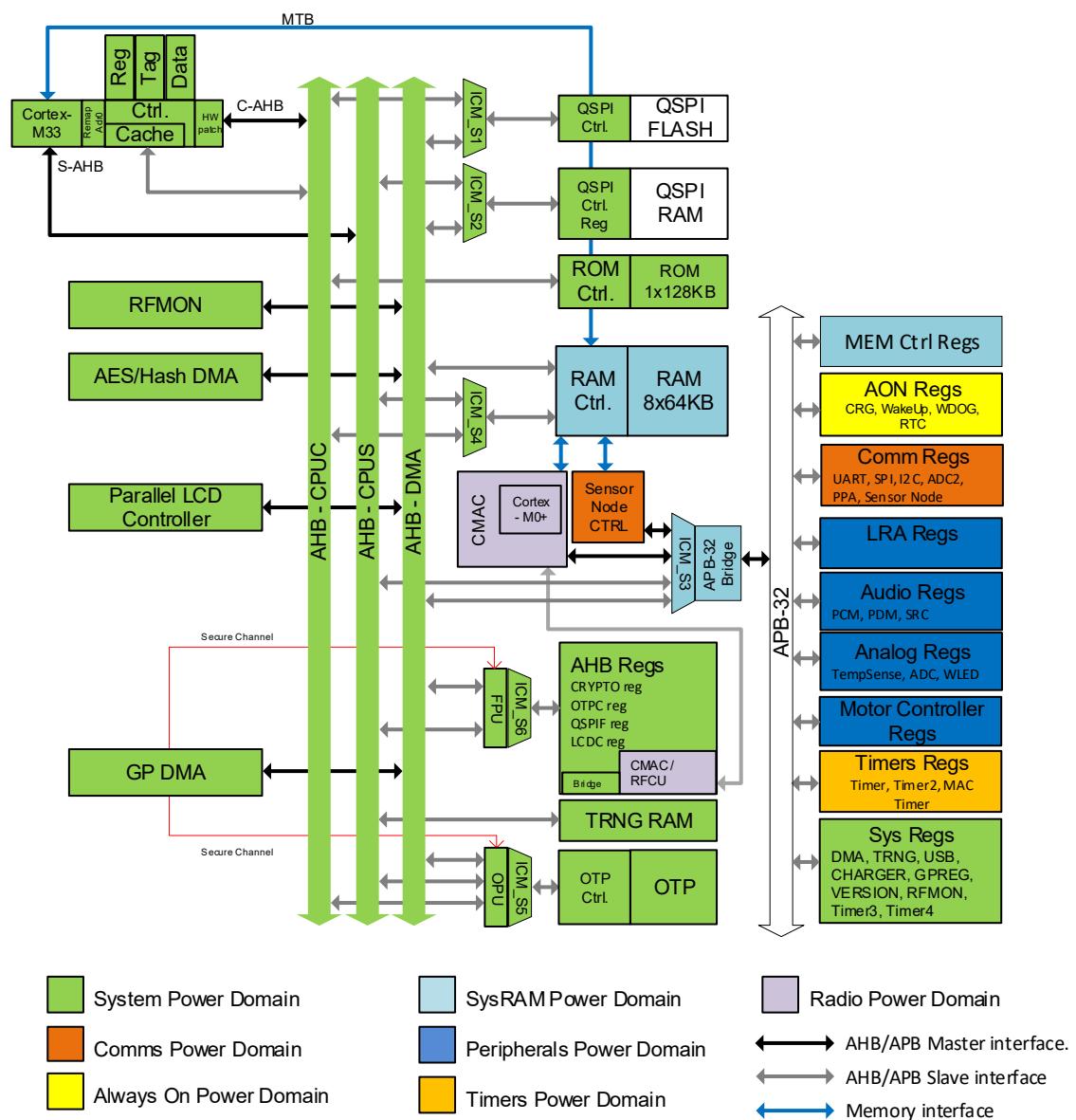


Figure 29: Bus Architecture

13.2 Architecture

The architecture of the AMBA bus matrix is shown in the [Figure 29](#). There are four potential masters for the APB32, namely: Sensor Node Controller, CMAC, Cortex-M33, and the DMA engines. A multiplexer is required there to allow access to a single master on the APB32 multiplexing, between the Sensor Node and the bridge.

There are two protection units - (1) the OTP Protection Unit (OPU), and (2) the FLASH Controller Register Unit (FPU)] – that form part of the system's security perimeter:

- The OPU protects the OTP section where keys are stored; keys that the CPU cannot read. Access to this space is only allowed when a special signal (secure channel). Read/Write protection strategy should follow the sticky bits definition.
- The FPU protects the QSPI FLASH Controller registers that keep the AES key (write only) from being written by the CPU and the AES/HASH registers from being read by the CPU. Access is only allowed by the GP DMA and only if the respective secure channel signal is activated. Rest of accesses to other register files are completely transparent and will not be gated by the FPU. Read/Write protection strategy should follow the sticky bits definition.

Multi-core Bluetooth 5.2 SoC family with system PMU

The priorities of the arbitration on the AHB-DMA bus should be programmable since the LCD DMA can actually occupy the bus continuously. The default priorities are listed in the following table:

Table 93: AHB-DMA Master Priorities

| Priority | Master |
|------------|----------------|
| 1 | RFMON |
| 2 | LCD Controller |
| 3 | GPDMA |
| 4 (lowest) | AES/HASH |

14 Configurable MAC (CMAC)

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

14.1 Introduction

The Configurable Medium Access Controller (CMAC) is a Flexible MAC HW block, based on Arm Cortex-M0+ that can be programmed to support multiple protocols.

The CMAC executes code from the system RAM with zero wait states. It also has access to all APB peripherals of the system.

Features

- Implements BLE 5.x controller stack, including HCI
- Optional BLE 5.x Features supported:
 - 2 Mbps
 - Advertising Extensions
 - Channel selection algorithm #2
 - Periodic Advertising
 - High Duty Cycle Non-Connectable Advertising
- Autonomous operation for Advertising or keep-alive connections
- Autonomous execution and sleep cycles
- Rapid wake-up and go-to-sleep operation
- Size and base address of RAM for code execution is configurable
- Accelerators in hardware:
 - Link Layer and framing Timers
 - AES-128-bit crypto engine
 - Whitening 7-bit engine, compliant with BLE standard.
 - CRC 24-bit engine, compliant with BLE standard
 - 32 bits wide Correlator
 - AoA/AoD support in HW
 - Coexistence interface

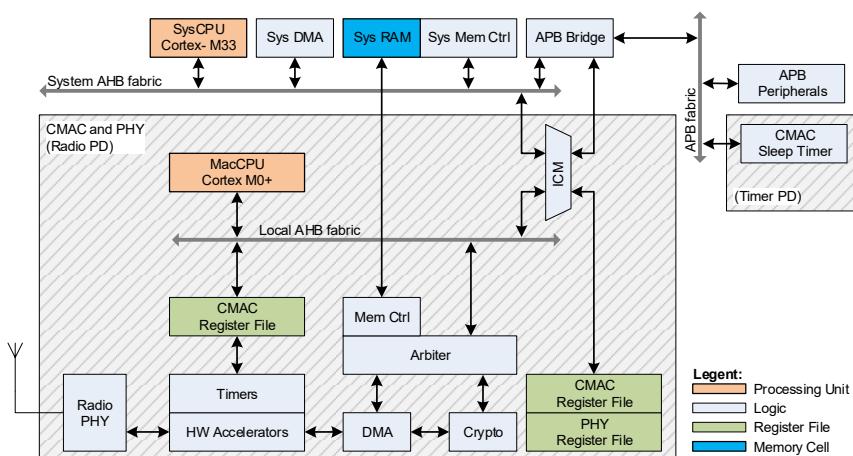


Figure 30: Configurable MAC Block Diagram

14.2 Architecture

The CMAC block is an autonomous system that can execute BLE and other protocols, if there is enough HW accelerators and Firmware throughput.

A dedicated Cortex-M0+ executes code through the local memory controller, which reuses the system memory RAM for storing code and data. A hardware bit stream controller performs all real time functions by using hardware accelerators like whitening, CRC, and crypto engines. A Link Layer Timer schedules protocol operations and radio transactions. The radio transactions are programmed to the CMAC registers. The HW will execute them automatically in the scheduled moment, activating the proper accelerators.

Radio Register File and CMAC Sleep Timer Register Files are accessible by both CPUs, but typically only CMAC CPU access them.

14.2.1 Dataflow

CMAC can implement the BLE Data Link Layer, providing an HCI interface towards the system processor Cortex-M33.

The communication of CMAC processor Cortex-M0+ with the system processor Cortex-M33 is performed via IRQ signals and the common system memory RAM. A mailbox mechanism can be used to exchange command and data structures between the two CPUs.

14.2.2 Diagnostics

[Table 94](#) shows available CMAC diagnostics signals.

Table 94: CMAC Diagnostic Signals

| Diagnostics | Signal | Description |
|-------------------|-----------|---|
| CMAC_DIAG_0 | TX EN | Data transmit enable signal |
| CMAC_DIAG_1 | RX EN | Data receive enable signal |
| CMAC_DIAG_2 | DATA EN | Tx/Rx Data Enable pulse |
| CMAC_DIAG_3 | DATA COMB | TX and RX data bits (combined on the same line) |
| CMAC_DIAG_4 | Reserved | - |
| CMAC_DIAG_5 | Reserved | - |
| CMAC_DIAG_6 | Reserved | - |
| CMAC_DIAG_7 | CORR COMB | Indicates that the correlator is active |
| CMAC_DIAG_8 | CRC | RX CRC LFSR is zero to indicate correctly received packet |
| CMAC_DIAG_9 to 15 | Reserved | - |

The functionality of the diagnostic signals is depicted in [Figure 31](#).

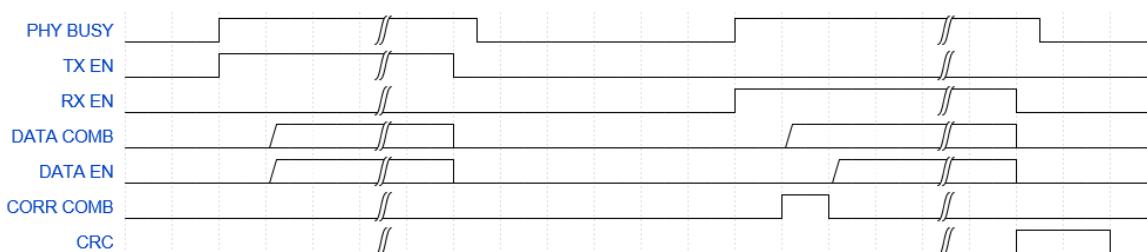


Figure 31: CMAC Diagnostics Timing Diagram

15 Sensor Node Controller (SNC)

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

15.1 Introduction

The Sensor Node Controller is a sophisticated hardware state machine, implementing a very short list of instructions that are enough to create small programs to manipulate communication controllers and the sensors connected to them. It is very small and can operate autonomously, without waking up the rest of the system. The minimum instruction set allows for polling sensor status bits, compare register to memory values, transfer data from communication interfaces to system RAM, branch on comparison and the like, as well as dissipating minimal current.

Features

- Minimal Instruction Set for Sensor manipulation uCode generation
- System RAM used for uCode storage as well as data
- DMA capabilities for storing data directly from serial interfaces to system RAM
- Immediate execution after domain power up and interrupt pending
- Running at system clock speed or less (programmable)

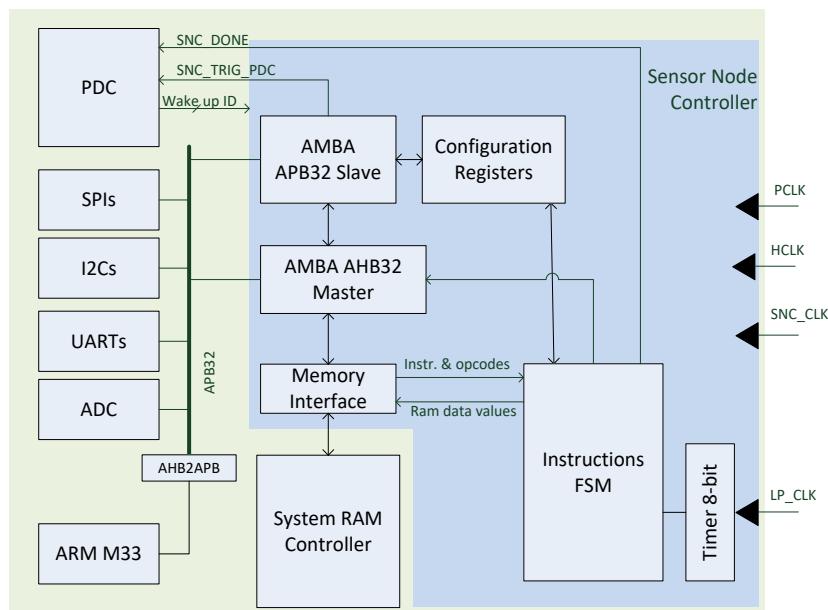


Figure 32: Sensor Node Controller Block Diagram

15.2 Architecture

The Sensor Node Controller is a master on the AHB bus. It resides in the same power domain as all the communication interfaces and $\Sigma\Delta$ ADC. It can control other power domains connected to the APB32 bus.

The actual micro-code resides in the System RAM. The Sensor Node Controller has a direct memory connection to the System RAM. It operates at system clock speed. It can generate an interrupt that tells the Power Domains Controller the sensor reading operations are done, and the whole system can now be powered OFF.

Since the Sensor Node Controller can be activated at the same time as the system CPU, the BUSY Status Register (BSR) can indicate the availability of the communication interfaces. The System CPU

Multi-core Bluetooth 5.2 SoC family with system PMU

should always check the BSR before accessing any of the communication interfaces, to make sure there is no conflict.

The Sensor Node Controller comprises the following blocks as shown in [Figure 23](#):

1. The Instructions FSM (I_FSM) with the 8-bit Timer. This HW FSM, implements a minimized proprietary instruction set, which is tailored to the Sensors interfacing and data acquisition. This instruction set is described in detail in the following section. The 8-bit Timer is used to implement a precise delay. It is controlled by one of the instructions.
2. A memory interface. This interface reads instructions the I_FSM executes from the system RAM. It also stores data fetched from sensors via the communication interfaces, using the master APB32 interface.
3. A master AHB. Implements writing and reading to communication and peripheral controllers.
4. A slave APB32 used for configuring the internal registers. This is supposed to happen by the Arm Cortex-M33 CPU during initialization of the system. Base addresses in System RAM of the uCode and data buffers for the various sensors support, should be defined in the register file.
5. A register file. This file contains registers needed to configure basic functions of the Sensor Node (r/w), and the registers holding the currently executed opcode and the program counter (read-only, for debugging purposes).

15.2.1 Sensor Node Instruction Set

The Sensor Node Instruction Set (SeNIS) is summarized in [Table 95](#):

Table 95: Sensor Node Instruction Set Overview

| Command | Operand1 | Operand2 | Description |
|---------|--|---|---|
| WADAD | Value indicating an Address 1+19=20 bits MSbit=1, Register MSbit=0, RAM | Pointer to RAM/register indicating an Address 1+19 bits MSbit=0, RAM MSbit=1, Register | Store contents of Reg/RAM defined by the pointer in Operand2 into the address defined by the value of Operand1. $\text{Bit 27} = 0: [\text{Operand1}]$ $\text{Bit 27} = 1: (\text{Operand1})$ $\text{Bit 26} = 0: (\text{Operand2})$ $\text{Bit 26} = 1: [\text{Operand2}]$ |
| WADVA | Value indicating an Address 1+19=20 bits MSbit=1, Register MSbit=0, RAM | Value 32 bits | Store value in Operand2 into the address defined by the value of Operand 1. $\text{Bit 27} = 0: [\text{Operand1}] \leftarrow \text{Operand2}$ or $\text{Bit 27} = 1: (\text{Operand1}) \leftarrow \text{Operand2}$ |
| TOBRE | Value indicating an Address 1+19=20 bits MSbit=1, Register MSbit=0, RAM | Mask 32 bits | XOR the value with the mask. If mask contains 1 at a specific bit place, then this bit's value is toggled. $(\text{Operand1}) \leftarrow (\text{Operand1}) \text{ XOR Operand2}$ |
| RDCBI | Value indicating an Address 1+19=20 bits MSbit=1, Register MSbit=0, RAM | Bit place (0 to 31) | Read and compare the contents of Reg/RAM defined by the value of Operand1 with “1” at the bit place $\text{If } (\text{Operand1}):(\text{Operand2})=1 \text{ then EQUALHIGH_FLAG=true}$ |
| RDCGR | Value indicating an Address 1+19=20 bits MSbit=1, Register MSbit=0, RAM | Value indicating an Address 1+19=20 bits MSbit=1, Register MSbit=0, RAM | Compare contents residing in the Address defined by Operand1 with contents residing in the Address defined by Operand2. $\text{If } (\text{Operand1})>(\text{Operand2}) \text{ then GREATERVAL_FLAG=true}$ |
| COBR | Value indicating a RAM Address 20 bits (20 th bit is don't care) | Branch according to EQUALHIGH_FLAG or GREATERVAL_FLAG | <ul style="list-style-type: none"> • If Operand2= 0x0A, then branch to (Operand1) address if EQUALHIGH_FLAG=True |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Command | Operand1 | Operand2 | Description |
|---------|--|--------------------------------------|---|
| | | or For a specific number of times | <ul style="list-style-type: none"> If Operand2= 0x1A, then branch to [Operand1] address if EQUALHIGH_FLAG=True If Operand2= 0x05, then branch to (Operand1) address if GREATER_FLAG=True If Operand2= 0x15, then branch to [Operand1] address if GREATER_FLAG=True Operand2= 0b1yyyyyy, then branch to (Operand1) address for up to 128 times |
| INC | Value indicating a RAM Address 1+19 = 20 bits | None | Increments the content of (Operand1) memory address either by 1 (default) or by 4, depending on bit [19] of Operand1, as follows: <i>Operand1[19] = 0 :</i> $(Operand1) \leftarrow (Operand1) + 1$ <i>Operand1[19] = 1 :</i> $(Operand1) \leftarrow (Operand1) + 4$ |
| DEL | Value indicating a number of clock ticks 8 bits | None | Starts a delay of <Operand1> ticks, where a tick is an internal 8-bit timer running on the sleep clock (32KHz). After timer expires, program execution continues. |
| SLP | None | None | Designates the end of the execution. Will automatically generate a signal pulse to the Power Domains Controller to set the system in sleep and power down the Sensor Controller |
| NOP | None | None | No Operation |

(Operand): When in simple brackets, then the Operand's value is an address to either System RAM or a Register (direct addressing).

[Operand]: When in square brackets, then the Operand's value is a pointer to a memory space in which the requested address resides (indirect addressing).

The actual opcodes and Operand sizes of each of the instructions are shown in [Table 96](#):

Table 96: Opcode and Operand sizes

| Instruction | Opcode (4 bits) | Operand1 size | Operand2 size | Addressing Bits |
|-------------|-----------------|---------------|---------------|-----------------|
| NOP | 0x0 | - | - | - |
| WADAD | 0x1 | 20 bits | 20 bits | 2 bits |
| WADVA | 0x2 | 20 bits | 32 bits | 1 bit |
| TOBRE | 0x3 | 20 bits | 32 bits | - |
| RDCBI | 0x4 | 20 bits | 5 bits | - |
| RDCGR | 0x5 | 20 bits | 20 bits | - |
| COBR | 0x6 | 20 bits | 8 bits | - |
| INC | 0x7 | 20 bits | | |
| DEL | 0x8 | 8 bits | | |
| SLP | 0x9 | | | |
| Reserved | 0xA – 0xF | | | |

15.2.2 SNC Clocking Considerations

If the LP clock is used as system clock, the SNC clock should be equal to the LP clock. If the SNC clock is divided, the DEL command may not function properly. The proper configuration demands the register be:

- CLK_AMBA_REG[HCLK_DIV] = 0
- CLK_AMBA_REG[PCLK_DIV] = 0
- CLK_COM_REG[SNC_DIV] = 0

15.3 Programming

Programming the Sensor Node Controller for use requires a simple sequence of steps. There is a detailed library of macros available for manipulating interfaces. The following steps explains the controller's initialization and configuration.

1. Load the SNC execution code into the RAM.
2. Setup the SNC clock by configuring the clock divider (CLK_COM_REG[SNC_DIV]).
3. Setup the SNC base address (SNC_BASE_REG).
4. Execute the code:
 - a. Using PDC: Setup PDC entry.
 - b. Manual code execution:
 - i. Disable PDC triggering by setting the SNC_CTRL_REG[SNC_SW_CTRL] bit.
 - ii. Enable the SNC by setting the SNC_CTRL_REG[SNC_EN] bit.
 - iii. Check the SNC_STATUS_REG[SNC_DONE_STATUS] bit:
0: Indicates the SNC is running.
1: Indicates the SNC has finished the code execution.

16 Memory Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

16.1 Introduction

The memory controller allows access to the 512 kB RAM (384 kB on DA14691) pool by the system's masters; these include the: SysCPU, MACCPU Sensor Node Controller, General Purpose DMA controller, and LCD Controller. It comprises two AHB ports (GPDMA and SysCPU) and three Memory ports (Sensor Node Controller, MACCPU, and Micro Trace Buffer (MTB)).

The memory controller implements an intelligent addressing scheme so that RAM is not fragmented by various data or code allocations. At the same time, it allows parallel access of multiple data streams on different RAM cells, transparent to the application software. So, it allows the System CPU to store/read application variables, while the MAC CPU executes code, the LCD reads a frame buffer and the Sensor Node stores sensor data at the same time, without conflict and arbitration that would result in wait states.

Definition of the different segments allocated in memory, is fully programmable. In cases where two or more segments are sharing the same RAM cell, hardware arbitration will resolve conflicts by issuing wait states to the masters having the least priority.

Features

- Supports flexible memory allocation per master to avoid fragmentation
- Parallelizes data flows from/to various masters on the system
- Allows for programmable priority scheme per RAM cell
- Generates wait cycles to AHB masters

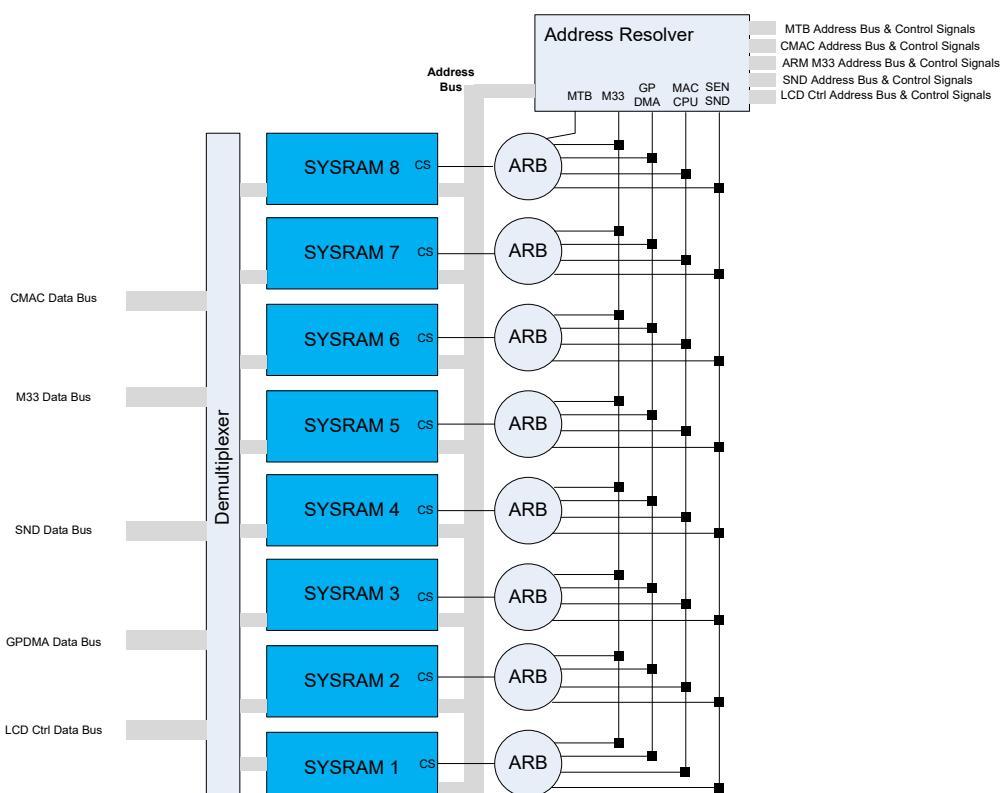


Figure 33: Internal Architecture of the Memory Controller

Multi-core Bluetooth 5.2 SoC family with system PMU

16.2 Architecture

The internal architecture of the memory controller is shown in [Figure 33](#).

Where two or more masters need access to the same RAM cell but different addresses (each on its own segment), then arbitration is used. If wait cycles (for the least prioritized) are not acceptable, a reconfiguration of the segment boundaries, at the cost of some RAM fragmentation, should be considered.

All masters should support a “Ready” like signal functionality, so that they can be stalled for a specific amount of clock cycles. This makes the arbitration signals for the AHB interfaces (SysCPU, Parallel LCD and GP DMA) HREADY, while the memory interfaces (Sensor Node and MTB) should also support a respective input (MREADY) and should only proceed in reading/writing the RAM cell if this signal is high.

The CMAC does not support any MREADY functionality, so it always has the highest priority.

An overview of the basic metrics of each master accessing the memory controller, is shown in [Table 97](#):

Table 97: Memory Controller Masters Access Metrics

| Master | Frequency Range | Stall Mechanism | Access Rate | Wait States Tolerance |
|----------------|-----------------|-----------------|---|--|
| Cortex-M33 | 32 MHz – 96 MHz | AHB HREADY | Non-burst accesses for data and/or code | Should keep that < 4 AHB clocks to avoid performance degradation |
| CMAC | 32 MHz – 96 MHz | None | Non-burst accesses for data and/or code | 0 |
| MTB | 32 MHz – 96 MHz | None | Non-burst accesses for data and/or code | 0 |
| Sensor Node | 2 MHz – 32 MHz | MREADY signal | Up to 8-beat burst access for data. Non-burst for code | It is application dependent. We should keep that <10 clocks |
| GP DMA | 32 MHz – 96 MHz | AHB HREADY | 8-beat | < 6 AHB clocks |
| LCD Controller | 32 MHz – 96 MHz | AHB HREADY | 8-beat burst access for data. | < 6 AHB clocks |

Two different arbitration schemes are provided:

1. Static priority of masters. The MEM_PRIO_REG defines three fields, one for each AHB layer and a third for the Sensor Node Controller. The priority level of each channel can be defined by programming these fields. Arbitration is always done in favor of the highest priority master.
2. Round Robing priority of masters. To avoid stalling a master for very too long, another register (MEM_STALL_REG) can be programmed with the maximum amount of clock cycles that a channel might be stalled. As soon as this number is reached, this channel will automatically retrieve highest priority.

Programming the MEM_STALL_REG with values other than 0, enables the second scheme.

16.3 Programming

Programmability is required for the range (Start and Stop address) of segments that need to be defined in the memory, depending on the type and size of the application. The following list contains the least number of segments needed for proper operation of the system.

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 98: Memory Segments Description

| Segment Name | Description | Access | Start/Stop Address Registers |
|-------------------|--|--|---|
| CMAC stack | Controller stack code and temporary variables | Cortex-M0+ (read, write) Cortex-M33 (write) DMA (write) | CMAC_CODE_BASE_REG (Note 1) CMAC_DATA_BASE_REG CMAC_END_REG |
| Mailbox | Implements the structure which is used for exchanging commands and data between the 2 CPUs | Cortex-M0+ (read, write) Cortex-M33 (read, write) | |
| Cortex-M33 Code | Special code segment for SysCPU. HW Patching code can be placed here as well | Cortex-M33 (read, write) | |
| Cortex-M33 Data | Application data space | Cortex-M33 (read, write) GP DMA (read, write) | |
| Sensor Node uCode | SeNIS uCode space | Sensor Node (read) Cortex-M33 (write) GP DMA (write) | |
| Sensor Node Data | Buffers for storing data read from Sensor Node Operation | Sensor Node (write) Cortex-M33 (read) Cortex-M0+ (read) GP DMA (read) | SNC_CODE_BASE_REG |
| System | IVT and others | 0x000000 | 0x000800 |

Note 1 This value should also be programmed in the Cortex-M33 MPU region 1 to ensure the Application does not corrupt CMAC code/data.

[Figure 34](#) shows an example of an actual mapping of the segments mentioned. This is not mandatory but is recommended.

Multi-core Bluetooth 5.2 SoC family with system PMU

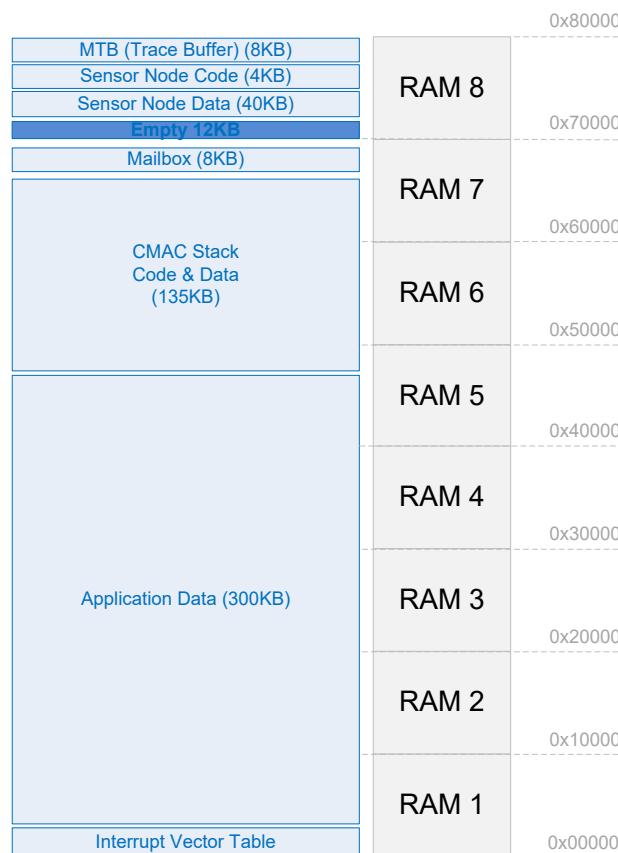


Figure 34: Example of Memory Segments Mapping

In the example in [Figure 34](#), there is no memory fragmentation other than at RAM8. This is the RAM where the Micro Trace Buffer (MTB) is allocated. In this case, MTB which has no stalling mechanism, should have the highest priority. The rest of the allocated MTB RAM can still be used for other masters, but not for CMAC, because this master also has no stalling mechanism. So, CMAC and MTB should not be allocated on the same RAM cell.

The rest of the RAM cells can be accessed by multiple masters, all at the same time. Arbitration on every cell is required to manage this. This means wait states insertion for the masters.

A different layout can be selected to solve such conflicts. This is done by defining frequent users of the memory on different RAM cells. This will improve performance (no wait states) at cost of memory fragmentation (empty holes in the memory space).

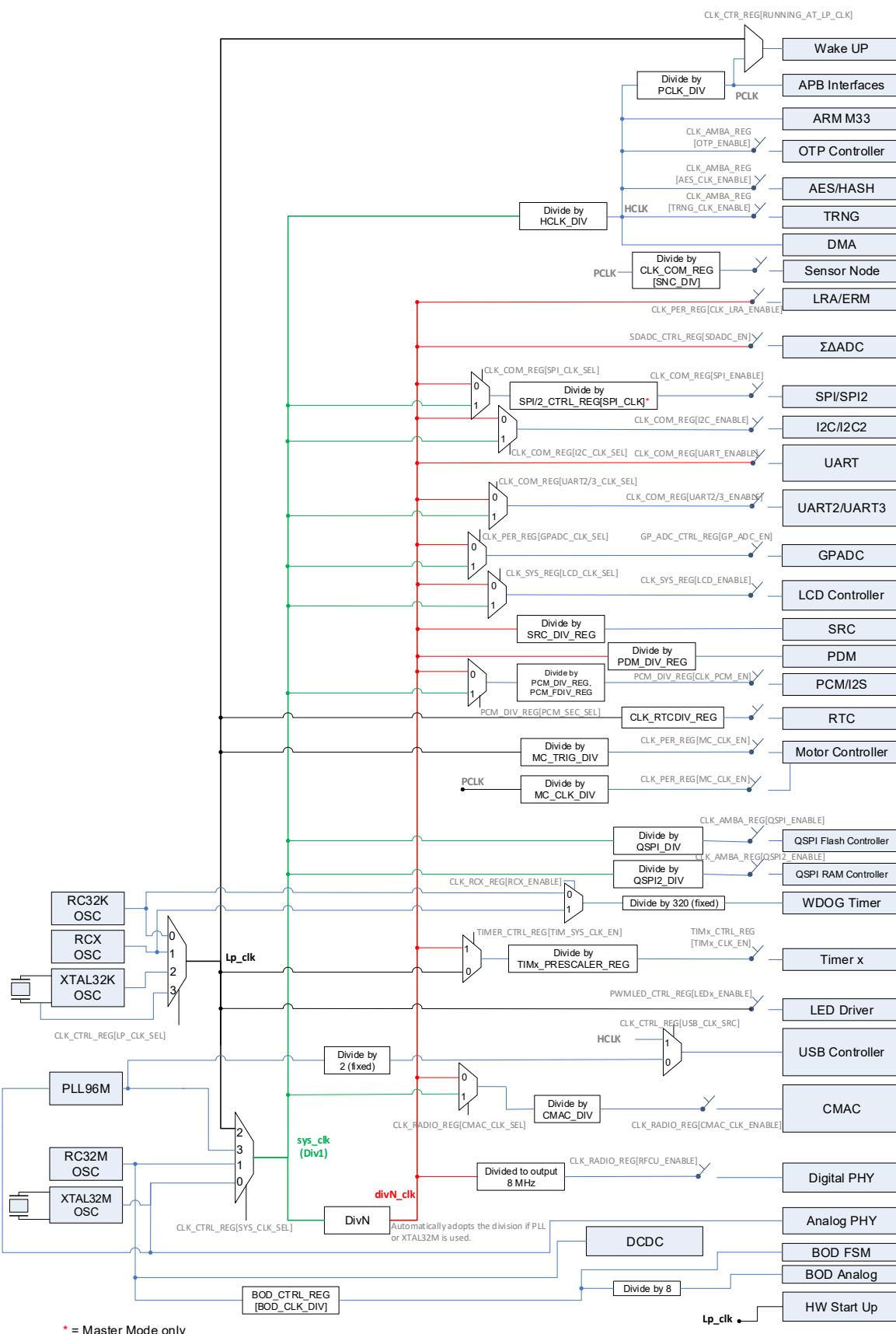
17 Clock Generation

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

17.1 Clock Tree

The generation of the system's clocks is described in detail in [Figure 35](#):

Multi-core Bluetooth 5.2 SoC family with system PMU



Multi-core Bluetooth 5.2 SoC family with system PMU

Figure 35: Clock Tree Diagram

The diagram shows the possible clock sources and all different divisions and multiplexing paths towards the generation of each block's clock. Also, the required registers that must be programmed, are also labelled on the same diagram. There are some main clock lines which are of interest:

- **lp_clk** (black bold line): this is the low power clock used for the sleep modes and can only be the RCX, RC32K or XTAL32K.
- **sys_clk** (green line): this is the system clock, - used for the AMBA clock (hclk) - which runs the CPU, memories, and the bus. The source of this clock can be any oscillator, the PLL, or even an externally supplied digital clock.
- **divn_clk** (red line): this is a clock which automatically adjusts the division factor on the sys_clk to always generate 32 MHz. This enables the dynamic activation of the PLL to provide more processing power at the CPU, without affecting the operation of blocks designed for 32 MHz.

Using the PLL and dividing the sys_clk by 2 while accessing peripherals should not be programmed. The apb_clk will be a multiple of 48MHz while the actual blocks will be operating on 32MHz. Hence, synchronization issues might be observed. If not accessing peripherals, the CPU can operate at 48MHz without a problem.

17.2 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 a second at 32.768 kHz (XTAL32K). The 32.768 kHz oscillator has no trimming capabilities and is used as the clock of the Extended Sleep mode. The 32 MHz oscillator can be trimmed.

The principle schematic of the two oscillators is shown in [Figure 36](#). No external components to the DA1469x are required other than the crystal itself. If the crystal has a case connection, it is advised to connect the case to ground.

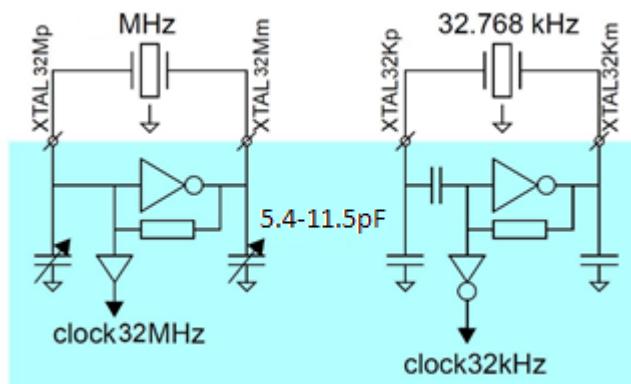


Figure 36: Crystal Oscillator Circuits

17.2.1 Frequency Control (32 MHz Crystal)

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.

With CLK_FREQ_TRIM_REG[XTAL32M_TRIM] = 0x2BF the maximum capacitance, and minimum frequency is selected. With CLK_FREQ_TRIM_REG[XTAL32M_TRIM] = 0x000 the minimum capacitance, and maximum frequency is selected.

The ten least significant bits of CLK_FREQ_TRIM_REG register (XTAL32M_TRIM bit field) directly control ten binary weighted capacitors, as shown in [Figure 37](#).

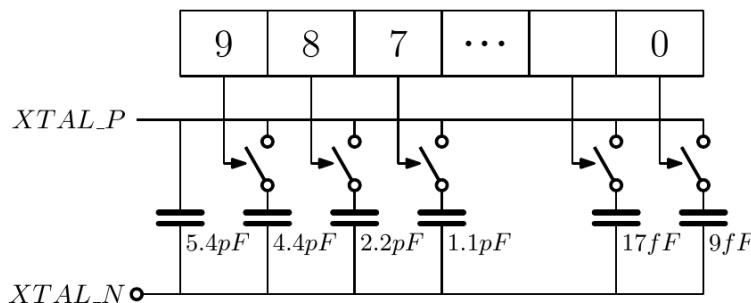


Figure 37: XTAL32MHz Oscillator Frequency Trimming

17.3 RC Oscillators

The DA1469x has three RC oscillators. These are:

- RC32M generates 32 MHz
- RC32K generates 32 kHz and
- RCX generates ~15 kHz

The 32 MHz RC oscillator is powered by the Digital LDO VDD, which is available for the core logic during Active or Sleep Mode. The output clock is significantly slower than 32 MHz if untrimmed and is used to clock the CPU and the digital part of the chip during power up or wake up, while the XTAL32M oscillator is settling.

The simple RC oscillator (RC32K) operates on VDD and generates 32 kHz. The main usage of the RC32K oscillator is for internal clocking during power up or startup. It clocks the HW state machine which brings up the power management system of the chip. The enhanced RC oscillator (RCX) generates 15 kHz. The RCX oscillator can be used to replace the 32.768 kHz crystal, since it has a precision of < 500 ppm, while its output frequency is quite stable over temperature.

17.3.1 Frequency Calibration

The output frequency of the 32 kHz crystal oscillator and the three RC-oscillators can be measured relative to the DivN clock, using the on-chip reference counter.

The measurement procedure is as follows:

- REF_CNT_VAL = N (the higher N, the more accurate and longer the calibration will be)
- CLK_REF_SEL_REG[REF_CLK_SEL] = 0 (RC32K) or
CLK_REF_SEL_REG[REF_CLK_SEL] = 1 (RC32M) or
CLK_REF_SEL_REG[REF_CLK_SEL] = 2 (XTAL32K) or
CLK_REF_SEL_REG[REF_CLK_SEL] = 3 (RCX)
CLK_REF_SEL_REG[REF_CLK_SEL] = 4 (RCOSC)
- Start the calibration: CLK_REF_SEL_REG[REF_CAL_START] = 1
- Wait until CLK_REF_SEL_REG[REF_CAL_START] = 0
- Read CLK_REF_VAL_REG = M (32-bits value)
- Frequency = (N/M) * 32 MHz

In the case of using the RCX as a sleep clock, the frequency calibration should be implemented at a frequency that ensures BLE connection maintenance.

17.4 PLL

The low power PLL multiplies the XTAL32M clock to produce a 96 MHz clock with very high precision within a few us.

Changing the system's clock in to the PLL output can be done dynamically without affecting the operation of the chip. Its main purpose is to:

Multi-core Bluetooth 5.2 SoC family with system PMU

- Provide a divided by 2, 48 MHz required for the operation of the USB Controller
- Provide more processing power to the CPU, enabling 144 dMIPS, for computational hungry applications

18 OTP Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

18.1 Introduction

The OTP controller realizes all functions of the 4 kB OTP macro cell. The controller facilitates all data transfers (reading and programming). The controller comprises two AHB slave interfaces, one for the configuration registers and one for the actual OTP memory cell. The OTP memory space will be transparently read by the CPU, but not transparently programmed.

The OTP controller operates on the AHB system clock. It takes care of the timing requirements of the various OTP cell operations using system clock tick values in configurable registers. The actual accesses to the OTP cell are performed at no more than 20 MHz, with the controller making sure this frequency constraint is always respected.

Features

- Implements all timing constraints for any access to the physical memory cell in a configurable manner
- Automatic Error Code Correction (ECC) – 6 bits (implemented by the OTP cell)
- 32-bits read in a single clock cycle from the OTP cell
- Single word buffer for programming. No burst programming supported
- Empty words are 0xFFFFFFFF. Zeros are programmed per 32-bit word

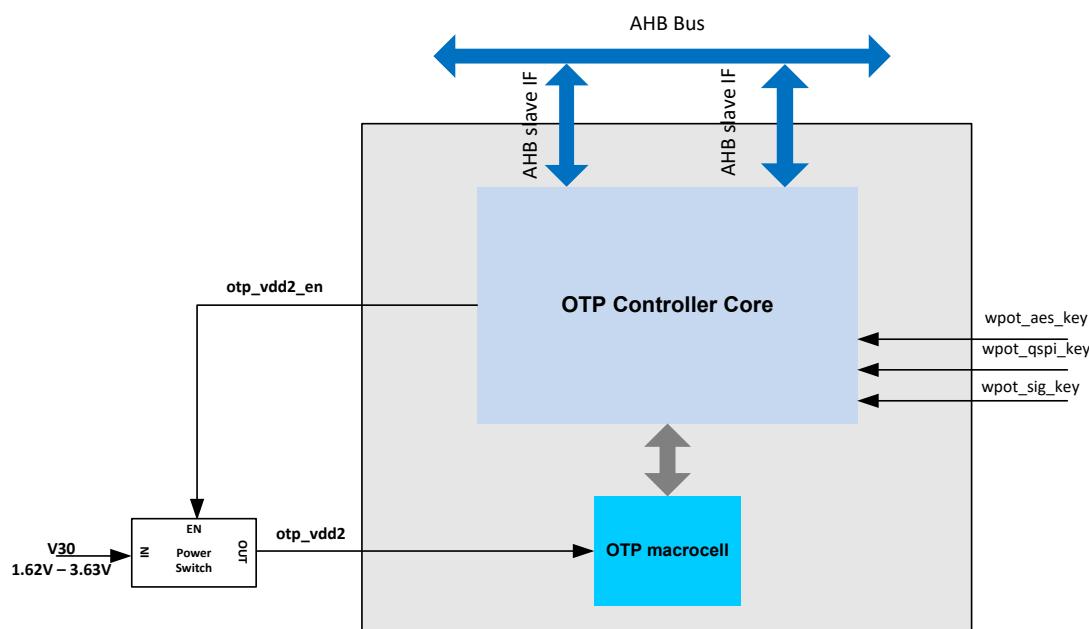


Figure 38: OTP Controller Block Diagram

18.2 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.

A power switch is used to control the power of the OTP cell driven by the digital logic.

The internal organization of the OTP cell is 32 bits data + 6 bits ECC for each of the 1024 addressable positions. The 6 bits of the ECC are not accessible outside of the OTP cell. The ECC is generated during the programming by the OTP cell, and is used during the reading again by the OTP cell, with a transparent way.

Multi-core Bluetooth 5.2 SoC family with system PMU

The system supplies three signals that enables write protection of three address ranges in the OTP memory. This protection prevents three specific address ranges of the OTP memory from being programmed. The request comes from three interface signals. The three address ranges (in byte address) and their corresponding interface signal that enables protection are:

- Range1: 0x0B00 – 0x0BFF, is protected when PROT_QSPI_KEY_WRITE is asserted
- Range2: 0x0A00 – 0x0AFF, is protected when PROT_AES_KEY_WRITE is asserted
- Range3: 0x08C0 – 0x09BF, is protected when PROT_SIG_KEY_WRITE is asserted

These three address ranges host various encryption keys in the OTP cell: The encryption key of the AES algorithm in the QSPIC, the encryption key for the general-purpose AES engine and the signature public key.

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

- **Power Down Mode (PDOWN).** In this mode of operation, the OTP cell is not powered by the VDD2 pin and the internal LDO is inactive. The otp_vdd2_en signal is 0 and the otp_vdd2 should be at 0 V. In this mode, the OTP cell consumes the least possible power, while the system is in active power state.
- **Deep Standby Mode (DSTBY).** In this mode, the required power supplies are applied to the OTP cell. However, the internal LDO of the OTP cell is inactive. The otp_vdd2_en signal is 1 and the otp_vdd2 should be at a functional voltage level (the VDD2 of the OTP cell is powered).
- **Standby 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 is not the lowest possible, but is less than the power consumption in active mode. This is the state from which any active mode of operation (READ, PROG, PVFY, RINI) happens with the least delay.
- **Read Mode (RD).** In this mode, the contents of the OTP cell are read at the respective AHB address space. This mode can be used for software execution in place (XIP). The OTP controller translates a read request into the corresponding control sequence for the OTP cell, to retrieve the requested data.
- **Programming Mode (PROG).** The PROG mode provides the functionality for programming 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 6-bits of the ECC information. Programming is performed only for bits equal to 0. Bits that are equal to 1 are bypassed to save on programming time. Due to the fact that the ECC value is unknown to the controller, there are always 6 extra programming pulses applied for the ECC bits. Programming is done by issuing a programming request stored in the Programming Buffer (PBUF). PBUF consists of 2 configuration registers storing the 32-bit data value and the 10-bit address in the OTP cell where the value should be programmed. A new request can only be stored in PBUF when the previous is served. A status bit indicates if this has already been done, and should therefore be monitored by SW before issuing a new programming request.
- **Programming Verification Mode (PVFY).** The PVFY mode forces the OTP cell to enter in 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 that the programmed data will be retrieved correctly under all the 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 will be applied. This test verifies that the OTP cell is empty (all the bits are equal to 1). The OTP controller will send to the OTP cell the required control sequence in order to enable the test mode. Following that, 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. This specific read mode is a margin read, which means that it is not equivalent with the normal read and should only be used for this purpose.

18.3 Programming

There is a simple sequence of steps that needs to be followed 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 standby mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x2).
3. Wait OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
4. Set OTP speed by writing:
 - a. OTPC_TIM1_REG =:
 - i. hclk = 16 MHz: 0x0999000F.
 - ii. hclk = 32 MHz: 0x0999101F (default).
 - iii. hclk = 48 MHz: 0x0999202F.
 - iv. hclk = 96 MHz: 0x0999515F.
 - b. OTPC_TIM2_REG = 0xA4040409.
5. Perform an OTP access:
 - a. Programming:
 - i. Setup OTP write mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x4).
 - ii. Wait OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
 - iii. Check OTPC_STAT_REG[OTPC_STAT_PBUF_EMPTY] = 1
 - iv. Write OTPC_PWORD_REG the data to be programmed.
 - v. Write OTPC_PADDR_REG the address that the data will be programmed to.
 - 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] = 0x5).
 - viii. Wait OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
 - ix. Read back and compare the data written.
 - x. Put the OTP in standby mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x2).
 - xi. Wait OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
 - b. Reading:
 - i. Setup OTP read mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x3).
 - ii. Wait OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).
 - iii. Read OTP word.
 - iv. Put the OTP in standby mode (OTPC_MODE_REG[OTPC_MODE_MODE] = 0x2).
 - v. Wait OTP mode to change (OTPC_STAT_REG[OTPC_STAT_MRDY] = 1).

Note: When VDD = 0.9V (POWER_CTRL_REG[VDD_LEVEL] = 0x0) and hclk = 32 MHz, OTPC_TIM1_REG[OTPC_TIM1_CC_T_RD] should be equal to 2.

19 Quad SPI FLASH Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

19.1 Introduction

The Quad SPI Controller (QSPIC) provides a low pin count interface to FLASH memory devices. The QSPIC supports the standard Serial Peripheral Interface (SPI) and a high performance Dual/Quad SPI Interface. The QSPIC gives the ability to read data from a quad FLASH memory, transparently through the SPI bus. This Execute-In-Place (XIP) feature combined with the CPU cache, provides comparable performance to executing code from standard parallel FLASH. In this case the QSPIC generates all the control signals for the SPI bus that are needed to read data from the serial FLASH memory. Additionally, software can easily control the serial FLASH memory via a memory mapped register file which is contained in the QSPIC. All instructions supported by the FLASH memory, can be programmed using the above register file. A special feature of the QSPIC enables for automated re-initialization of the FLASH device right after power up, without the CPU being involved, thus reducing initialization time and consequently power dissipation. A small initialization memory of sixteen 32-bit retainable words, contains an encoded sequence of commands which are shifted in to the FLASH memory right after waking up from power down modes.

Features

- SPI Modes:
 - Single: Data transfer via two unidirectional pins.
 - Dual: Data transfer via two bidirectional pins.
 - Quad: Data transfer via four bidirectional pins.
- Auto Mode: up-to 32 Mbyte transparent Code access for XIP (Execute-In-Place) and Data access with 3-byte and 4-byte addressing modes.
- Manual Mode: Direct register access using the QSPIC register file
- Up to 96 MHz QSPI clock. Clock modes 0 and 3. Master mode only
- Vendor independent Instruction Sequencer
- Support for single access and high-performance burst mode, in combination with the cache controller (in Auto Mode)
- Use of a special read instruction in the case of a specific (programmable) wrapping burst access
- Erase suspend/resume to Support for Code and Data storage
- Hardware initialization state machine based on uCode commands
- Decrypt on-the-fly (AES-256b-CTR) capability while in auto mode operation

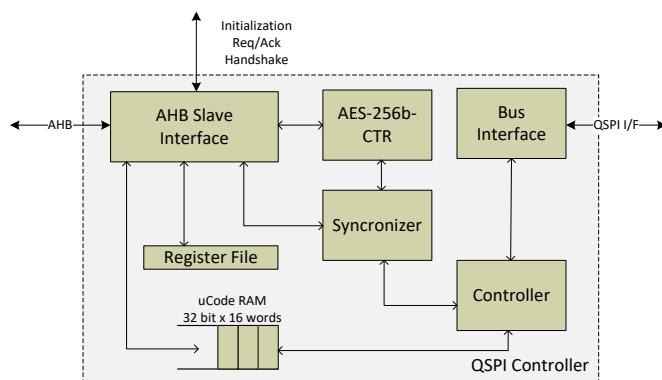


Figure 39: QSPIC FLASH Controller Block Diagram

19.2 Architecture

The AHB slave block implements two AHB Slave interfaces which enable access to the register file and the uCode memory. The Controller implements all protocol related to the functionality of the FLASH memory. It contains a finite state machine (FSM) that generates all necessary signaling to the QSPI bus and realizes all features of the Auto mode operation. Moreover, it manages all data transfers between the two interfaces (the AHB and the QSPI).

The Bus Interface block controls the QSPI signals at the lowest level while the Synchronizer implements "stretching" or "shortening" of the signals that cross the two clock domains.

The uCode memory is sixteen words x 32 bits. It contains the microcode for the initialization of the FLASH memory even before the CPU has been waken up. The signaling between the FIFO and the PMU is done through the request/acknowledge signals.

19.2.1 Interface

The Quad SPI Controller uses the following signals:

- QSPI_SCK: output serial clock
- QSPI_CS: Active Low output Chip select.
- QSPI_IO0:
 - DO (output) in Single SPI mode
 - IO0 (bidirectional) in Dual/Quad SPI mode.
- QSPI_IO1:
 - DI (input) in Standard SPI mode
 - IO1 (bidirectional) in Dual/Quad SPI mode.
- QSPI_IO2:
 - General purpose (output) (e.g. WPn Write Protect) in Standard SPI mode
 - IO2 (bidirectional) in Quad SPI mode.
- QSPI_IO3:
 - General purpose (output) (e.g. HOLDn) in Single SPI mode
 - IO3 (bidirectional) at Quad SPI mode.
- The output drive of the pads is programmable via register bits
`QSPI_GP_REG[QSPI_PADS_DRV]` and slew via `QSPI_GP_REG[QSPI_PADS_SLEW]`

The Quad SPI Controller (QSPIC) drives all data pins constantly except for the case when a read is performed. The time for changing the direction of the pads is at least $1.5 \times \text{QSPI_CLK}$ (`QSPI_CLK` being the clock that the FLASH operates on). In this way, data lines are always terminated thus reducing unnecessary power consumption.

The default state of the `QSPI_IOx` pins is 1. This state is applied at the pins as soon as the QSPIC clock is enabled even if no access to the external FLASH has yet been triggered. The value of the pins might be changed by programming the respective registers (`QSPIC_IOx_DAT`, `QSPIC_IOx_OEN`). This value will be valid only after the `QSPI_CS` is pulled low, i.e. an access to the external FLASH occurs.

19.2.2 Initialization FSM

Since the QSPIC is used in an ultra-low-power SoC, it is possible that the FLASH memory will be either totally powered off, or set into deep power-down mode when the system goes to any of the sleep modes.

However, upon power-up or wake-up, the FLASH device requires a few commands to get at a state where the CPU can execute code from. This initialization should be done prior to the CPU wakeup. The QSPIC contains a hardware state machine, which decodes a few commands in a sixteen 32-bit word retainable FIFO and initializes the FLASH automatically, even before the CPU is waken up.

Multi-core Bluetooth 5.2 SoC family with system PMU

The command FIFO will be initialized upon cold boot of the system by the CPU with the commands residing in the FLASH header. The start address of the RAM to be programmed with the uCode is 0x38000040. The command encoding is presented in [Table 99](#):

Table 99: Initialization Command Encoding

| Bit | Name | Description |
|---------------------------------|---------------|--|
| Byte 0 | | |
| 7:3 | CMD_NBYTES | The number of payload bytes to be sent |
| 2:1 | CMD_TX_MD | QSPI bus mode when transmitting the command: 0x0: single SPI 0x1: Dual SPI 0x2: Quad SPI 0x3: Reserved |
| 0 | CMD_VALID | 1: the command record is valid 0: the command record is not valid |
| Byte 1 | | |
| 7:0 | CMD_WT_CNT_LS | Number of clock cycles to wait after applying the command (least significant byte) |
| Byte 2 | | |
| 7:0 | CMD_WT_CNT_MS | Number of clock cycles to wait after applying the command (most significant byte) |
| Byte 3 to (CMD_NBYTES+2) | | |
| | | The actual data bytes to be sent within a CS envelope |

The first byte (LSByte) in the word of the FIFO contains the flag of the command being valid or not, the bus mode of operation and the number of bytes contained in the payload to be sent. The second and third byte define the amount of clock cycles that the QSPIC must wait, after applying the command. The clock to be used is the RC32M (~32 MHz), which results in a maximum of 4 ms waiting time. If more time is required by the FLASH, then multiple identical commands might be issued.

Example:

Considering 0xAB to be the opcode for releasing the FLASH from deep power-down mode, the FIFO would be initialized with the following sequence:

Table 100: FLASH Initialization uCode Example

| Byte | Value | Description |
|------|-------|---|
| 0 | 0x11 | Valid command record, single SPI mode, 2 bytes of payload |
| 1 | 0x01 | 1 clock cycle wait after command is sent |
| 2 | 0x00 | |
| 3 | 0xAB | Actual FLASH command opcode |

19.2.3 SPI Modes

The Quad SPI Controller (QSPIC) supports the following SPI standards:

- Single: Data transfer via two unidirectional pins. The QSPIC supports communication to any single/dual or Quad SPI FLASH memory. However, the Single SPI interface does not support bus modes 1 and 2, full-duplex communication and any SPI slave mode

Multi-core Bluetooth 5.2 SoC family with system PMU

- Dual: Data transfer via two bidirectional pins
- Quad: Data transfer via four bidirectional pins

19.2.4 Access Modes

The access to a serial FLASH connected to the QSPI can be done in two modes:

- Auto mode
- Manual mode

These modes are mutually exclusive. The serial FLASH can operate only in one of the two modes. In auto mode, 3-bytes and 4-bytes addressing modes are supported. With QSPIC_CTRLMODE_REG[QSPIC_USE_32BA]=0, up to 16 MBytes QSPI (3-bytes addressing) can be accessed. If QSPIC_USE_32BA=1, 4-bytes addressing is enabled for accessing up to 32 Mbyte QSPI FLASH.

Auto mode

In auto mode, a read access from the serial FLASH memory is fully transparent to the CPU. A read access at the interface is translated by the QSPIC into the respective SPI bus control commands needed for the FLASH memory access. When the Auto Mode is disabled, any access (reading or writing) will be ignored. When the Auto Mode is enabled, only read access is supported. A write access causes hard fault. A read access can be single access, incremental burst or wrapping burst. Wrapping burst is supported even when the FLASH device doesn't support any special instruction for wrapping burst. A special read instruction can be used in the case of a specific (programmable) wrapping burst access. When a FLASH supports a special instruction for wrapping burst access, it reduces access time (less wait states). For maximizing the utilization of the bus and minimizing the number of wait states, it is recommended to use burst accesses. However, non-sequential random accesses are supported with cost of more wait states.

Manual Mode

In manual mode the FLASH memory is controlled via a register file. All instructions that are supported by a FLASH memory can be programmed using the register file. Moreover, the mode of interface (SPI, Dual SPI, Quad SPI) and the mode of operation (Auto or Manual Mode) can be configured via this register file. The register file supports the following data sizes for reading and writing accesses: 8 bits, 16 bits and 32 bits.

19.2.5 Endianess

The QSPIC operates in little-endian mode. For 32-bit or 16-bit access (for read and write operations) to a serial FLASH memory, the least-significant byte comes first. For 32-bit access, the byte ordering is: data [7:0], data [15:8], data [23:16], data [31:24] while for 16-bit access the byte ordering is: data [7:0], data [15:8].

19.2.6 Erase Suspend/Resume

The QSPI FLASH can be used for Data Storage, combining the EEPROM functionality + Program storage in one single device. For this purpose, the QSPI ERASE/SUSPEND ERASE/RESUME commands are automatically executed as shown in [Figure 40](#).

To store data in QSPI FLASH, execution from QSPI must temporary be stopped by running directly from RAM or from a cached program part. The sector selected for storage must be erased first, in case it contained data already. The process is implemented in a HW FSM and consists of the following steps:

1. The controller is in Auto mode and read requests are served. The Erase procedure is initiated by setting QSPIC_ERASE_EN=1. The address of the sector that will be erased is defined at QSPIC_ERS_ADDR. When an Erase procedure is requested, the controller jumps to state 2.
2. Read requests are still served. As soon as the Read requests stop (also possible due to late bus master change, e.g DMA) and there is no any new Read request for a few AHB clock cycles

Multi-core Bluetooth 5.2 SoC family with system PMU

equal to QSPIC_ERSRES_HLD, then QSPIC_WEN_INST and QSPIC_ERS_INST instructions are sent to the FLASH. The QSPIC_RESSUS_DLY counter is started and the controller jumps to state 3.

3. Erasing is in progress in FLASH and the QSPI controller waits until one of the following events occur:
 - a. A status check request. This request can be forced by writing SPIC_CHCKERASE_REG. The QSPI controller will then read the status of the FLASH memory and check if erasing has finished. Reading of the status is delayed by QSPIC_RESSTS_DLY cycles or by QSPIC_RESSUS_DLY cycles. The first is based on the clock of the SPI bus, while the latter on an internal 222 KHz clock. The selection between the two delays is configured by QSPIC_STSDLY_SEL bit. If erasing has finished, the QSPI controller returns to the normal operation (state 1) and sets QSPIC_ERASE_EN=0, otherwise it remains at state 3.
 - b. A FLASH read data request on the AHB bus. The QSPI controller reads the status of the FLASH memory and checks if erasing is done. The reading of the status will be delayed again as in the previous case by QSPIC_RESSTS_DLY or QSPIC_RESSUS_DLY. If erasing has ended, the controller returns to normal operation (state 1) and sets QSPIC_ERASE_EN=0. The read request will be served as soon as the controller reaches state 1. If erasing has not ended, the controller proceeds to state 4.
4. The QSPIC_SUS_INST is sent as soon as the QSPIC_RESSUS_DLY/QSPIC_RESSTS_DLY counter is 0. The controller jumps to state 5.
5. The controller polls the FLASH status register, until the FLASH device becomes ready (erasing is suspended). The controller will then proceed to state 6.
6. The erasing process in the FLASH is now suspended and the controller may read the FLASH. The requested data are retrieved from the FLASH device. If the reading on the AHB stops, (for example, Cache hit), and there is no new Read requests for a number of AHB clock cycles equal to QSPIC_ERSRES_HLD, the controller goes to state 7.
7. The QSPIC_RES_INST instruction is applied, and the controller jumps back to state 3. Also, the QSPIC_RESSUS_DLY counter is started. As a result, the erase procedure is resumed.

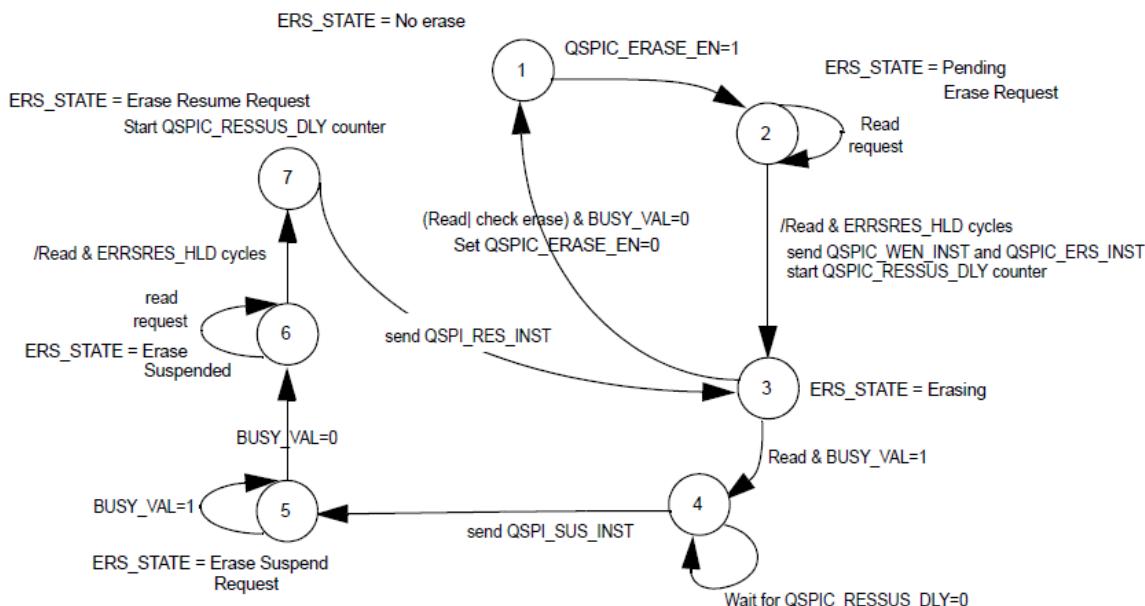


Figure 40: Erase Suspend/Resume in Auto Mode

Note that, QSPIC_RESSTS_DLY counts QSPI_CLK cycles, so before changing the QSPI_CLK, make sure that QSPIC_RESSTS_DLY is set large enough to meet the timing parameter requirements of the FLASH device used.

19.2.7 On-the-fly Decryption

The QSPIC supports decryption of the data retrieved from the FLASH device, only when in Auto mode. FLASH contents should be encrypted already, using the same algorithm.

The address range that will be decrypted automatically by the controller, is defined in the QSPIC by using two configuration registers, one for the start address (ENC_START_ADDR), and one for the end address (ENC_END_ADDR). The defined address range is 1024 bytes aligned. All addresses that are outside of this range, will not be automatically decrypted, but fetched as is.

The on-the-fly decryption feature is based on the CTR mode of the AES encryption algorithm. The AES algorithm processes blocks of 128 bits. That means that the input and the output block of the AES is 128 bits or 16 bytes and as a result, data to be processed by the algorithm, are fragmented into blocks of 16 byte. The key size that is used for the AES algorithm is 256 bits.

For the AES-256-CTR algorithm, only the cipher part of the AES algorithm is required. The general idea of the encryption process is based on the encryption of a 128-bit counter block (CTR). The initial value of the CTR is labeled as CTR0. The first counter block (CTR0) is encrypted with the help of the AES cipher, and the encrypted result is XORed with the first 16 bytes of the plaintext data (P0) to be encrypted. The counter block is incremented by one ($CTR1 = CTR0 + 1$) and is encrypted again. The result is XORed, with the next 16 bytes of the plaintext (P1), and so on, until all plaintext data are encrypted.

The AES CTR decryption is the same process as the encryption. By XORing again the ciphertext with the same encrypted counter value, the plaintext is retrieved. The decryption process can be described by equations as bellow:

For $j=1$ to m , do $CTR_j = CTR_{j-1} + 1$

For $j=0$ to m , do $P'_j = AES_CIPH_k(CTR_j) \oplus C_j$

Because accesses to the FLASH memory are random, the structure of the CTR block is selected to simplify the process. The total size of the counter block is 128 bits or 16 bytes namely: CTR_{B_0} , CTR_{B_1} , CTR_{B_2} , CTR_{B_3} , ..., $CTR_{B_{14}}$, $CTR_{B_{15}}$.

The first 8 bytes (CTR_{B_0} - CTR_{B_7}) of the counter block comprise the NONCE value and are programmed in the QSPI Controller in configuration registers. This is typically a random value and is the same for all the CTR_i blocks.

The next 4 bytes of the counter block (CTR_{B_8} - $CTR_{B_{11}}$) are always zero.

The last 4 bytes of the counter block ($CTR_{B_{12}}$ - $CTR_{B_{15}}$), are produced automatically by the hardware, based on the 32-bit address offset $OFFSET_ADDR[31:0]$ inside the encrypted range, where the data that should be decrypted are placed. If $FLASH_ADDR[31:0]$ is the absolute address of a specific byte inside the encrypted range, the offset address is $OFFSET_ADDR = FLASH_ADDR - ENC_START_ADDR$. The four least significant bits of the address offset are truncated and the four most significant bits of the $CTR_{B_{12}}$ are padded with zeros. Thus, the zero value in bytes $CTR_{B_{12}}$ - $CTR_{B_{15}}$ of the CTR is used for the first 16 bytes of the address range that is encrypted, the value 1 is used for the second 16 bytes of the address range etc.

The final form of the counter block is the following:

{64 bits NONCE, 32 bits 0x0, 4 bits 0x0, $OFFSET_ADDR[31:4]$ }

The four least significant bits of the address offset $OFFSET_ADDR[3:0]$ define which of the $AES_CIPH_k(CTR_j)$ byte should be used for the decryption of a specific byte of the encrypted block C_i .

In this way the CTR block that should be used for the decryption of a specific byte, can be calculated immediately by the address of the byte in the FLASH and the start address of the encrypted range. This counter block supports up to 4 GB data, which covers the maximum supported size for the FLASH devices.

19.2.8 Timing

This section contains timing diagrams for input and output signals

Multi-core Bluetooth 5.2 SoC family with system PMU

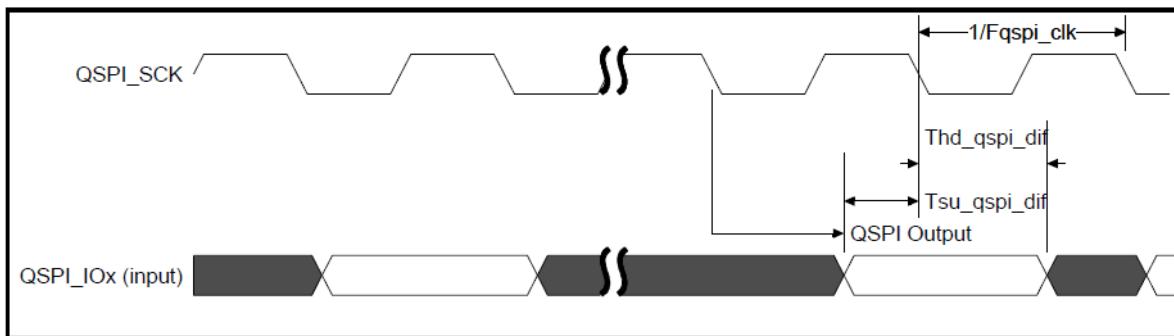


Figure 41: QSPI Input Timing

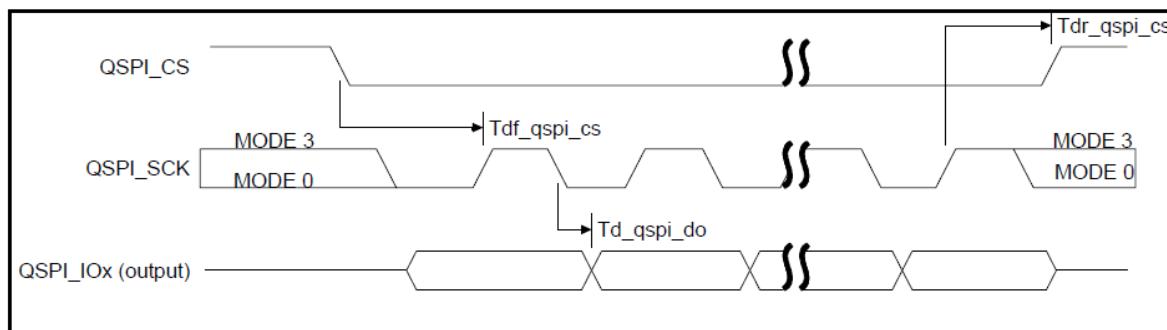


Figure 42: QSPI Output Timing

Table 101: QSPI Timing Parameters

| Parameter | Description | Conditions | Min | Typ | Max | Units |
|---------------|---|---------------------------------|------|-----|-------------------|-------|
| Fqsipi_sck | QSPI_SCK frequency | | | | 96 | MHz |
| Td_qsipi_do | Delay QSPI_SCK to QSPI_IOx | | -1.4 | | 3 | ns |
| Tdr_qsipi_cs | Delay QSPI_SCK to QSPI_CS | Tqspi_sck=1/Fqsipi_sck - 2.5 | | | Tqspi_sck +2.0 | ns |
| Tdf_qsipi_cs | Delay QSPI_CS to QSPI_SCK | Tqspi_sck=1/Fqsipi_sck - 2.5 | | | Tqspi_sck +3.5 | ns |
| Tsu_qsipi_dif | Setup time QSPI_IO to QSPI_SCK falling edge with variable readpipe sample clock delay QSPI_RPIPE_EN=1 | QSPIC_PCLK_MD=6 | 2.6 | | | ns |
| Thd_qsipi_dif | Hold time QSPI_SCK falling edge to QSPI_IO with variable readpipe sample clock delay QSPI_RPIPE_EN=1 | QSPIC_PCLK_MD=6 | -0.3 | | | ns |

Note 1 Note: Total Delay QSPI FLASH output + PCB delay + Tsux_qsipi_dif < 1/Fqsipi_sck. E.g Winbond output + PCB delay = 6ns, -> Tsux_qsipi_dif < 1/96 MHz - 7 = 3.41 -> QSPIC_PCLK_MD= 6, QSPI_RPIPE_EN=1 is recommended value for all QSPI_SCK frequencies and shall be set before the maximum frequency is applied

19.3 Programming

19.3.1 Auto Mode

In the case of Auto Mode of operation, the QSPIC generates a sequence of control signals in SPI BUS. This sequence of control signals is analyzed to the following phases: instruction phase, address phase, extra byte phase, dummy clocks phase, and read data phase. These phases can be programmed via registers

- QSPIC_BURSTCMDA_REG
- QSPIC_BURSTCMDB_REG

Bits QSPIC_INST are used to set the selected instruction for the cases of incremental burst or single read access. If bit QSPIC_WRAP_MD is equal to 1, bit QSPIC_INST_WB can be used to set the used instruction for the case of a wrapping burst read access of length and size described by the bits QSPIC_WRAP_LEN and QSPIC_WRAP_SIZE respectively. In all other cases the QSPIC_INST is the selected instruction.

If the instruction must be transmitted only in the first access after the selection of Auto Mode, then the QSPIC_INST_MD must be equal to 1.

To enable the extra byte phase set QSPIC_EXT_BYTE_EN=1 register. The transmitted byte during the extra byte phase is specified by the QSPIC_EXT_BYTE register. To disable (hi-Z) the output pads during the transmission of bits [3:0] of extra byte, set QSPIC_EXT_HF_DS =1.

The number of dummy bytes during the dummy clocks phase is specified by register QSPIC_DMY_NUM and enabled by QSPIC_DMY_FORCE.

The SPI BUS mode during each phase can be set with register bits:

- QSPIC_INST_TX_MD for the instruction phase
- QSPIC_ADR_TX_MD for the address phase
- QSPIC_EXT_TX_MD for the extra byte phase
- QSPIC_DMY_TX_MD for the dummy byte phase
- QSPIC_DAT_RX_MD for the read data phase

If the Quad SPI mode is selected in any of the above phases, write 0 to the QSPIC_IO3_OEN and QSPIC_IO2_OEN.

If the FLASH Memory needs to be accessed with any instruction but the read instruction, then the Manual Mode must be used.

The final step to enable the use of Auto Mode of operation, is to set the QSPIC_AUTO_MD equal to 1.

19.3.2 Manual Mode

For the Manual Mode QSPIC_AUTO_MD must be equal to zero. Manual operation of the bus signals is done via QSPIC_CTRLBUS_REG:

- The start/end of an access can be controlled using bits QSPIC_EN_CS and QSPIC_DIS_CS respectively
- The SPI bus mode of operation can be configured with bits QSPIC_SET_SINGLE, QSPIC_SET_DUAL and QSPIC_SET_QUAD

Writing to QSPIC_WRITEDATA register, is generating a data transfer from the QSPIC to the SPI bus. A read access at QSPIC_READDATA register, is generating a data transfer from the SPI bus.

Writing to QSPIC_DUMMYDATA register, is generating a few dummy clock pulses to the SPI bus.

When access to the SPI bus via QSPIC_WRITEDATA, QSPIC_READDATA and QSPIC_DUMMYDATA is very slow, most probably the delay in accessing the internal AHB is large. In this case, set the QSPIC_HRDY_MD register equal to 1 to increase priority when accessing the required registers. All masters of the SoC can access the AHB bus interface without waiting for the SPI Bus access completion. Polling of the QSPIC_BUSY register must be done to check the end of

Multi-core Bluetooth 5.2 SoC family with system PMU

the activity at the SPI bus, before issuing any more accesses. If a read transaction is finished, QSPIC_RECVDATA contains the received data.

The state and the value of the QSPI_IO[3:2] is specified by the following registers bits:

- QSPIC_IO3_OEN, QSPIC_IO3_DAT (Used for the WPn, Write Protect function)
- QSPIC_IO2_OEN, QSPIC_IO2_DAT respectively (Used for the HOLDn function)

19.3.3 Clock Selection

The SPI clock mode as set with bit QSPIC_CLK_MD. The supported modes for the generated SPI clock is:

- 0 = Mode 0. The QSPI_SCK is low, when the bus is idle (QSPI_CS is high)
- 1= Mode 3. The QSPI_SCK is high, when the bus is idle (QSPI_CS is high)

The QSPI_CLK frequency has a programmable divider CLK_AMBA_REG[QSPI_DIV] which divides either XTAL32 or PLL by 1,2,4,8.

The QSPI_CLK can be faster or slower than HCLK.

19.3.4 Receiving Data

The standard method to sample received data, is by using the positive edge of the QSPI_SCK. However, when the output delay of the FLASH memory is high, a timing problem on the read path is very likely. For this reason, the QSPIC can be programmed to sample the received data with the negative edge of the QSPI_SCK. This is configured with the QSPIC_RXD_NEG register.

Furthermore, the receive data can be pipelined by setting QSPI_RPIPE_EN=1 and the sampling clock can be delayed using QSPI_PCLK_MD. This enables sampling the received data later than the actual clock edge allows.

19.3.5 Delay Line Configuration

When VDD = 0.9 V (POWER_CTRL_REG[VDD_LEVEL] = 0x0) and QSPI_CLK = 32 MHz, the QSPIC_CTRLMODE_REG[QSPIC_PCLK_MD] bit field should be equal to 2. On the contrary, if VDD = 1.2 V then QSPIC_CTRLMODE_REG[QSPIC_PCLK_MD] bit field should be equal to 7.

20 Quad SPI RAM/FLASH Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | x | ✓ | ✓ | ✓ |

20.1 Introduction

The Quad SPI Controller (QSPIC2) provides a low pin count interface to serial QSPI FLASH and RAM memory devices. The QSPIC2 supports the standard Serial Peripheral Interface (SPI) and a high performance Dual/Quad SPI Interface. The QSPI RAM feature provides a low-cost RAM extension for infrequently used data.

The QSPIC2 automatically generates all the control signals for the QSPI bus needed to access data from the serial quad memory. The controller has a vendor independent register file that provides a rich set of control fields for wide range of FLASH and RAMs.

The QSPIC2 provides transparent memory mapped RAM access.

Features

- SPI modes:
 - Single: Data transfer via two unidirectional pins
 - Dual: Data transfer via two bidirectional pins
 - Quad: Data transfer via four bidirectional pins
- Auto mode: up-to 32 MB memory mapped Read/Write Data access with 3-byte and 4-byte addressing modes
- Manual mode: Direct register access using the QSPIC2 register file
- QSPI clock up-to 96 MHz. Clock modes 0 and 3. Master mode only
- Vendor independent Instruction Sequencer
- In Auto mode the FLASH control signals are fully programmable
- Use of a special read instruction in the case of a specific (programmable) wrapping burst access
- Erase suspend/resume to Support for Code and Data storage

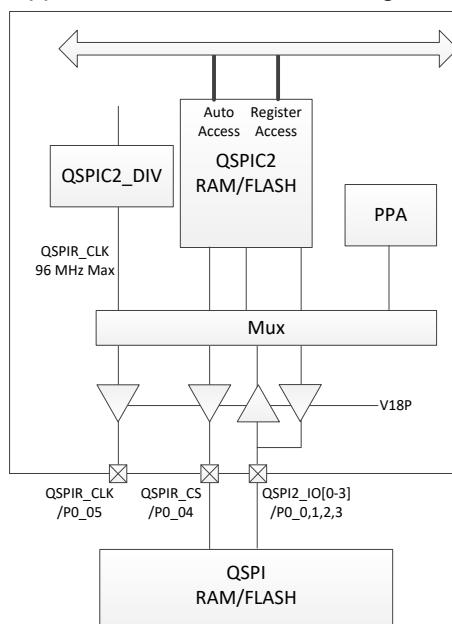


Figure 43: Quad SPI RAM/FLASH Controller

20.2 Architecture

20.2.1 Interface

- QSPI_SCK: output serial clock
- QSPI_CS: Active Low output Chip select
- QSPI_IO0:
 - DO (output) in Single SPI mode
 - IO0 (bidirectional) in Dual/Quad SPI mode
- QSPI_IO1:
 - DI (input) in Standard SPI mode
 - IO1 (bidirectional) in Dual/Quad SPI mode
- QSPI_IO2:
 - General purpose (output) (e.g. WPn Write Protect) in Standard SPI mode
 - IO2 (bidirectional) in Quad SPI mode
- QSPI_IO3:
 - General purpose (output) (e.g. HOLDn) in Single SPI mode
 - IO3 (bidirectional) at Quad SPI mode

20.2.2 SPI Modes

The Quad SPI Controller (QSPIC2) supports the following SPI standards:

- Single: Data transfer via two unidirectional pins. The QSPIC2 supports communication to any single/dual or Quad SPI FLASH memory. In contradiction to the Standard SPI interface, the supported Single SPI interface does not support the bus modes 1 and 2, does not support full-duplex communications and does not support any SPI slave mode
- Dual: Data transfer via two bidirectional pins
- Quad: Data transfer via four bidirectional pins

20.2.3 Access Modes

Access to a serial memory (FLASH or RAM) connected to the QSPIC2 can be done in two modes:

- Auto mode
- Manual mode

These modes are mutually exclusive. The serial memory can be controlled only in one of the two modes. The registers which control the mode of operation can be used at any time.

In Auto mode, 3-byte and 4-byte addressing modes are supported. With QSPIC2_USE_32BA=0, up to 16 MB serial memory (3-byte addressing) can be accessed. If QSPIC2_USE_32BA=1, the 4-byte addressing is enabled for accessing up to 32 MB serial memory.

Auto Mode FLASH Access

In auto mode (QSPIC2_AUTO_MD=1), the read access to a serial FLASH memory is performed in a fully transparent way through the SPI bus. A read access to the memory space, where the external memory is mapped, is translated by the controller to the respective SPI bus command sequence, which is needed for the retrieving of the requested data from the serial FLASH memory.

When the auto mode is disabled (QSPIC2_AUTO_MD =0), any access (reading or writing) to the mapped memory space is ignored by the controller.

Only read accesses are supported when the connected external device is a FLASH memory (QSPIC2_SRAM_EN=0). A write access causes a hard fault at the CPU.

Multi-core Bluetooth 5.2 SoC family with system PMU

The read access can be single access or incremental burst or wrapping burst access. The wrapping burst is supported even when the controlled serial FLASH doesn't support any special instruction for wrapping burst. A special read instruction can be used in the case of a specific (programmable) wrapping burst access. When a serial FLASH supports a special instruction for wrapping burst access, this feature saves access time (less wait states). For maximizing the utilization of the bus and minimizing the number of wait states, it is recommended to be used burst accesses. However, non-sequential random accesses are supported with the cost of more wait states.

Auto Mode RAM Access

In the case where it is connected to a serial RAM device, the QSPIC2 controller is capable of provide read/write functionality.

The special configuration register must be programmed in order to enable the RAM functionality (QSPIC2_SRAM_EN=1). As in the case where the external device is a FLASH, the auto mode must also be enabled (QSPIC2_AUTO_MD=1). In the case where the auto mode is disabled (QSPIC2_AUTO_MD=0), any access (reading or writing) in the memory space, where the external device has been mapped, is ignored by the QSPI controller.

The read access in the memory space of the external serial RAM memory, is done in a fully transparent way through the QSPI bus. The capability of the controller to handle the various types of read accesses, is the same as in the case of the FLASH device. Single access, incremental burst or wrapping burst are all supported.

A write access to the memory space where the external memory is mapped, does not cause a hard fault to the Cortex-M0. In the contrary, the write access is interpreted by the QSPIC2 in the respective QSPI bus protocol and the write data is stored in the external RAM device.

The controller is capable of handling write accesses of all kinds of burst: single access, incremental burst or wrapping burst access. The throughput that can be achieved, varies depending on the burst length, the word width, the cost of the protocol of the external memory device, and the frequency of the QSPI clock.

Burst accesses provides the highest throughput. The non-sequential random accesses are supported at the cost of more wait states. The maximum throughput that can be achieved, depends on the burst length.

Manual Mode

In manual mode, the external serial memory is controlled via a register file. All instructions that are supported by the serial memory can be programmed by using the register file. Moreover, the mode of interface (SPI, Dual SPI, Quad SPI) and the mode of operation (Auto or Manual mode), can be configured via this register file. The register file supports the following data sizes for reading and writing accesses: 8-bits, 16-bits, and 32 bits.

20.2.4 Endianness

The QSPI controller operates in little-endian mode. For 32-bit or 16-bit access (for read and write operations) to a serial memory, the least-significant byte comes first. For 32-bit access, the byte ordering is: data [7:0], data [15:8], data [23:16], data [31:24] and for 16-bit access the byte ordering is: data [7:0], data [15:8].

20.2.5 Erase Suspend/Resume

A QSPI FLASH memory can be used for data storage, combining the EEPROM functionality + Program storage in one single device.

For this purpose the QSPI ERASE/SUSPEND ERASE/RESUME are automatically executed as shown in Figure 44.

To store data in QSPI FLASH memory, the sector designated for storage must be erased first.

The ERASE/SUSPEND ERASE/RESUME process is only meaningful if the external device is a serial FLASH memory (QSPIC2_SRAM_EN=0).

Erase procedure

Multi-core Bluetooth 5.2 SoC family with system PMU

1. The controller is in Auto mode and the read requests are served. The Erase procedure is initiated by setting QSPIC2_ERASE_EN=1. The address of the sector that is erased, is defined by the QSPIC2_ERS_ADDR. When an Erase procedure is requested, the controller enters in state 2.
2. The read requests are still served. As soon as the Read requests stop (also possible due to late bus master change, e.g. DMA) and there is no new Read request for a number of AHB clock cycles equal to QSPIC2_ERSRES_HLD, the QSPIC2_WEN_INST and the QSPIC2_ERS_INST instructions, are sent to the QSPI FLASH. The QSPIC2_RESSUS_DLY counter is started. After this, the controller enters into state 3.
3. The erasing is in progress in the QSPI FLASH memory. The QSPI controller waits for one of the following:
 - a. A status check request. This request can be forced by writing QSPIC2_CHCKERASE_REG. This makes the QSPIC2 controller read the FLASH memory status and checks the end of erasing. Reading the status is delayed by QSPIC2_RESSTS_DLY cycles, or by QSPIC2_RESSUS_DLY cycles. The QSPIC2_RESSTS_DLY delay is based on the SPI bus clock, while the QSPIC2_RESSUS_DLY delay is based on a 288 kHz clock. The selection between the two delays, is configured with the help of the QSPIC2_STSDLY_SEL bit. After erasing, the QSPI controller returns to the normal operation (state 1) and sets QSPIC2_ERASE_EN= 0, otherwise it remains in state 3.
 - b. A read data request on the AHB bus. The QSPI controller reads the status of the FLASH memory and checks the end of erasing. Status reading is delayed again by QSPIC2_RESSTS_DLY cycles, or QSPIC2_RESSUS_DLY cycles. The QSPIC2_STSDLY_SEL bit does the Selection between two delays .At the end of erasing , the controller returns to normal operation (state 1) and sets QSPIC2_ERASE_EN= 0. The read request will be served in state 1. If the erasing has not ended, the controller proceeds to state 4.
4. The QSPIC2_SUS_INST is sent as soon as the QSPIC2_RESSUS_DLY counter is 0. The controller enters into state 5.
5. The controller reads the status register until the FLASH device is ready (erasing is suspended). When the FLASH device is ready, the controller enters state 6.
6. Erasing process in the FLASH is suspended and the controller can read the FLASH. The requested data is retrieved from the FLASH device. If the reading on the AHB stops (e.g. Cache hit) and there is no new Read request for a number of AHB clock cycles equal to QSPIC2_ERSRES_HLD, the controller enters state 7.
7. The QSPIC2_RES_INST instruction is applied and state 3 is entered. Also, the QSPIC2_RESSUS_DLY counter is started. The QSPIC2_RES_INST makes sure the erase procedure is continued (erase resume) in the flash device.

Multi-core Bluetooth 5.2 SoC family with system PMU

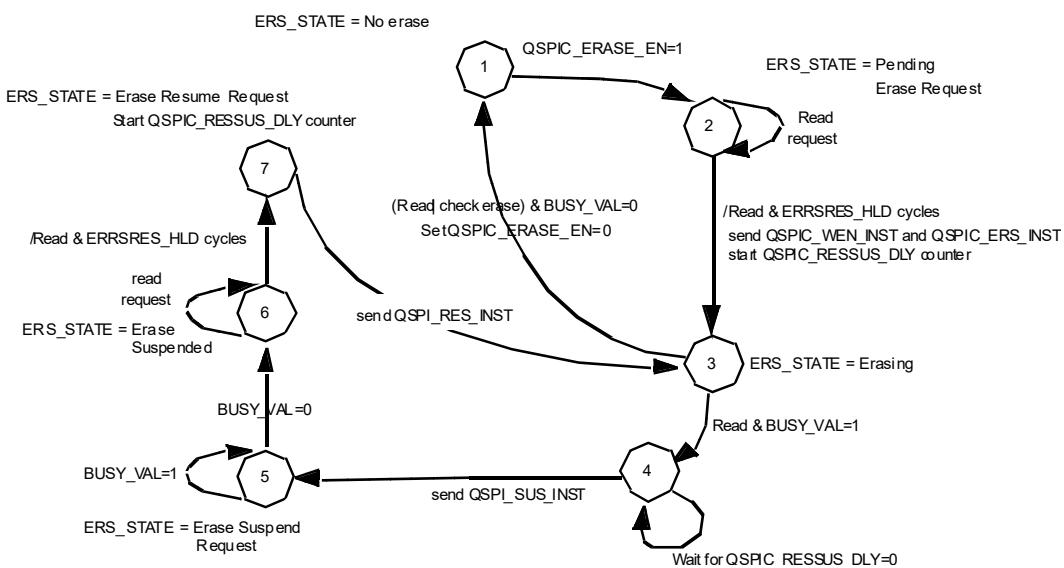


Figure 44: Erase Suspend/Resume in Auto Mode

Note that QSPI_RESSTS_DLY is counted with the QSPI_CLK, so before changing the QSPI_CLK, make sure that QSPI_RESSTS_DLY is set large enough to meet the timing parameter requirements.

QSPI FLASH Programming Procedure

Sectors are programmed in manual mode by polling the status bit in the QSPI FLASH. During programming, the CPU MUST run from shared or non-shared RAM. Also, interrupts must be disabled while executing the write command to the FLASH.

Byte programming is relatively short, so a polling loop could be acceptable to meet system latency requirements.

20.2.6 Low Power Considerations

In order to reduce the power dissipation in the QSPI FLASH, the QSPI_CLK must always be the highest possible system clock to keep the burst access to the FLASH as short as possible. The CPU must run as slow as possible for minimum power.

For lowest power with slow CPU (for example, 2 MHz) and high QSPI_CLK (for example, 32 MHz) bit QSPIC2_CTRLMODE_REG[QSPIC2_FORCENSEQ_EN] must be set to 1. This enables split burst mode, reducing the power dissipation during active burst only, while disabling the FLASH when the burst is done compared to high efficiency burst. These two modes are explained in the following two figures:

Multi-core Bluetooth 5.2 SoC family with system PMU

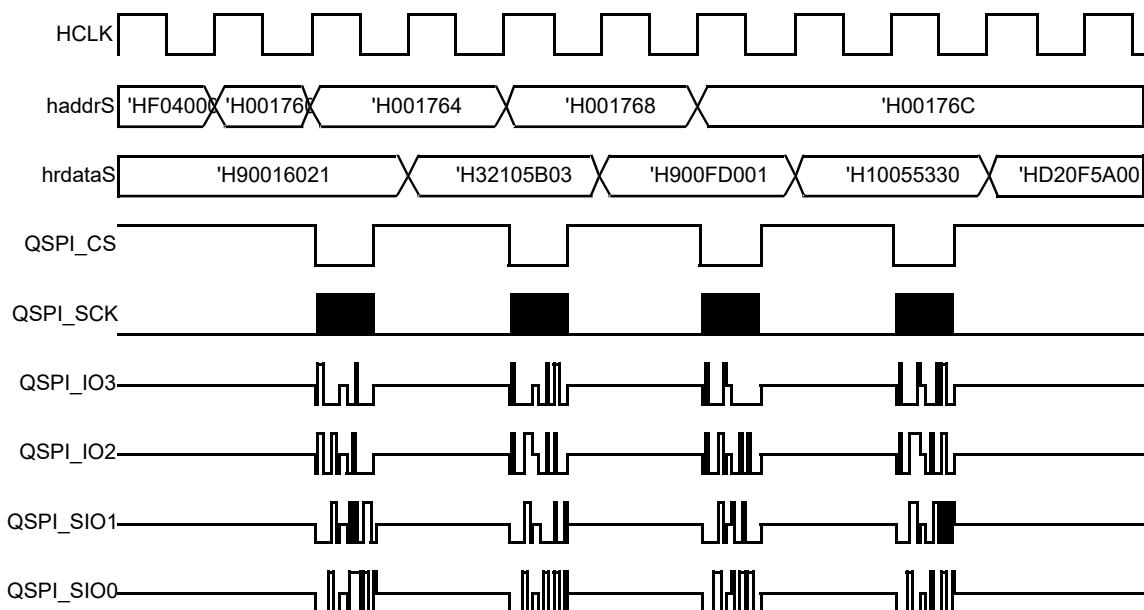


Figure 45: QSPI Split Burst Timing for Low Power (QSPI_FORENSEQ_EN=1)

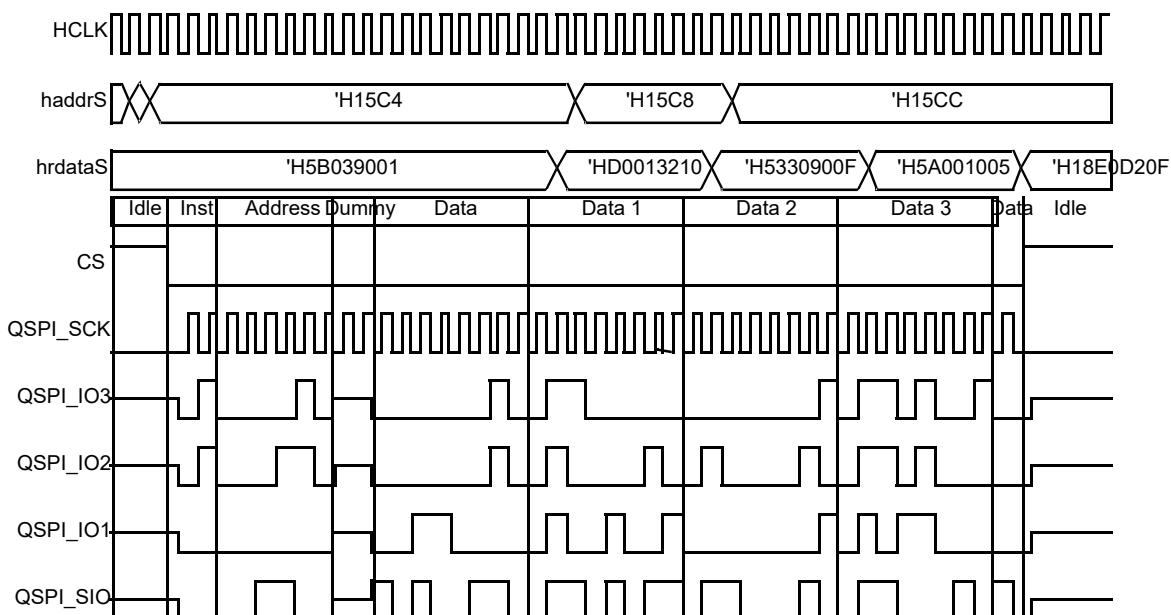


Figure 46: QSPI Burst Timing for High Performance (QSPI_FORENSEQ_EN=0)

If QSPI_FORENSEQ_EN=0, the QSPIC2 reads data in a burst address1/extra/dummy/data1, data2, dataN, keeping the QSPI_CS low during the complete burst. If set to '1', the burst is split into non-sequential accesses address1/extra/dummy/data1, address2/extra/dummy/data2, etc. making the QSPI_CS high between the accesses.

20.3 Programming

20.3.1 Auto Mode

Chip Selection

In auto mode QSPI executes from address 0. See Arm chapter remap function

Multi-core Bluetooth 5.2 SoC family with system PMU

Notice that certain PSRAM RAM (for example, APmemory APS3204J) connected to the QSPI interface have a minimum time t_{CEM} that the #CE may stay low. SW must make sure that the QSPI CLK is not going too slow during a burst, otherwise RAM data might get lost.

Read Burst in Auto Mode

In the case of a read from the external device, in Auto mode, the QSPI controller generates a sequence of control signals. This is analyzed in the following phases: instruction phase, address phase, extra byte phase, dummy clocks phase and read data phase. These phases can be programmed via registers QSPIC2_BURSTCMDA_REG and QSPIC2_BURSTCMDB_REG.

Bits QSPIC2_INST are used to set the selected instruction for the cases of incremental burst or single read access. If bit QSPIC2_WRAP_MD is equal to 1, bits QSPIC2_INST_WB can be used to set the used instruction for a wrapping burst read access. The length and size is described by the bits QSPIC2_WRAP_LEN and QSPIC2_WRAP_SIZE respectively. In all other cases, the QSPIC2_INST is the selected instruction.

If instruction is to be transmitted only during the first access after the selection of Auto mode, the QSPIC2_INST_MD must be equal to 1.

To enable the extra byte phase, set 1 to the QSPIC2_EXT_BYTE_EN register. The transmitted byte during the extra byte phase is specified from the QSPIC2_EXT_BYTE register. To disable (hi-z) the output pads during the transmission of bits [3:0] of extra byte, write 1 to the QSPIC2_EXT_HF_DS register.

The number of dummy bytes during the dummy clocks phase is specified at QSPIC2_DMY_NUM.

The SPI BUS mode during each phase can be configured as follows:

- QSPIC2_INST_TX_MD for the instruction phase
- QSPIC2_ADR_TX_MD for the address phase
- QSPIC2_EXT_TX_MD for the extra byte phase
- QSPIC2_DMY_TX_MD for the dummy byte phase
- QSPIC2_DAT_RX_MD for the read data phase.

If the Quad SPI mode is selected in any of the above phases, write 0 to the QSPIC2_IO3_OEN and QSPIC2_IO2_OEN.

If the serial FLASH Memory must be prepared for reading with the use of any instruction except the read instruction, then the Manual mode must be used for the programming of the above instructions.

The final step to enable the use of Auto mode of operation is to set the QSPIC2_AUTO_MD equal to 1.

Write Bursts in Auto Mode

In the case where the connected memory is a serial PSRAM, the controller can serve requests for write accesses. This is implemented, as in the case of the read burst, when the Auto mode of operation is active (QSPIC2_AUTO_MD=1). Additionally, the external device must be declared to the QSPI controller as a serial RAM (QSPIC2_SRAM_EN=1).

Under these conditions the QSPI controller generates a sequence of control signals in SPI BUS, for each request for write burst access towards the external device. This sequence of control signals is analyzed in the following phases: instruction phase, address phase, extra byte phase, and write data phase. These phases can be programmed via the register QSPIC2_AWRITECMD_REG.

Bits QSPIC2_WR_INST are used to define the write instruction. This instruction is used for all the cases of bursts: single access, incremental burst or wrapping burst. The controller will handle them accordingly to implement all the lengths of the bursts.

The SPI BUS mode during each phase can be set by the register bits:

- QSPIC2_WR_INST_TX_MD for the instruction phase
- QSPIC2_WR_ADR_TX_MD for the address phase
- QSPIC2_WR_DAT_TX_MD for the read data phase

Multi-core Bluetooth 5.2 SoC family with system PMU

If the serial RAM has to be configured with a special command sequence prior to the write instruction, Manual mode must be used.

20.3.2 Manual Mode

For the Manual mode, QSPIC2_AUTO_MD must be equal to zero. Manual operation of the bus signal is done via QSPIC2_CTRLBUS_REG:

- Start /End of an access can be controlled using bits QSPIC2_EN_CS and QSPIC2_DIS_CS respectively
- SPI mode configured with bits QSPIC2_SET_SINGLE, QSPIC2_SET_DUAL and QSPIC2_SET_QUAD

Writing to QSPIC2_WRITEDATA register generates a data transfer from the QSPIC2 to the SPI bus. A read access at QSPIC2_READDATA register generates a data transfer from the SPI bus. Writing to QSPIC2_DUMMYDATA register generates a number of clock pulses to the SPI bus. During this activity in the SPI bus, the QSPI_IO data pads are in *hi-z* state.

When an access to the SPI bus via QSPIC2_WRITEDATA, QSPIC2_READDATA and QSPIC2_DUMMYDATA is very slow, the delay in access to the AHB is very high. In this case, set the QSPIC2_HRDY_MD register equal to 1. With this feature, the *hready* signal of the SB slave interface is always equal to 1, when accessing the WriteData, ReadData, and DummyData registers. All masters can access the AHB bus without waiting for transmission completion on SPI Bus. A read of the QSPIC2_BUSY register must be done to check the end of the activity at the SPI bus, before any more access is triggered. In this case, register QSPIC2_RECVDATA contains the received data at the end of a read access.

20.3.3 Clock Selection

The SPI clock mode is set with bit QSPIC2_CLK_MD. The supported modes for the generated SPI clock are:

- 0 = Mode 0. The QSPI_SCK is low, when the bus is idle (QSPI_CS is high)
- 1= Mode 3. The QSPI_SCK is high, when the bus is idle (QSPI_CS is high)

The QSPI_CLK frequency has a programmable divider CLK_AMBA_REG[QSPIC2_DIV] which divides either XTAL or PLL by 1,2,4,8

20.3.4 Received Data

The standard method to sample the received data, is by using the positive edge of the QSPI_SCK. However, when the output delay of the serial FLASH is high, timing issues at the read path are very likely. For this reason, the QSPIC2 can be programmed to sample the received data with the negative edge of the QSPI_SCK. This is specified with the QSPIC2_RXD_NEG register.

Furthermore, the receive data can be pipelined by setting QSPI_RPIPE_EN=1 and the sample clock can be delayed using QSPI_PCLK_MD. Refer to the timing chapter for detailed QSPI Timing information.

20.3.5 Delay Line Configuration

When VDD = 0.9V (POWER_CTRL_REG[VDD_LEVEL] = 0x0) and QSPI_CLK = 32 MHz, the QSPIC2_CTRLMODE_REG[QSPIC_PCLK_MD] bit field should be equal to 2. On the contrary, if VDD = 1.2 V then QSPIC2_CTRLMODE_REG[QSPIC_PCLK_MD] bit field should be equal to 7.

21 Step Motor Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | x | x | x | ✓ |

21.1 Introduction

The Motor Controller drives the motor that moves the hands in a watch. It is very flexible in defining commands for forward or backward hand movements. Up to four GPIOs can be used to drive a hand, and as much as five hands are supported. Up to 64 commands can be stored into a memory mapped RAM which serves as a command FIFO. The actual waveforms are pre-defined and stored in a 48-places register based memory.

Features

- A 30.5 μ s granularity for defining a slot used for waveform generation
- Supports up to five independent 4-signal pattern-generators for hands control
- Implements a 64 command FIFO and a separate memory for storing up to 12 different wave definitions (the number depends on the wave format)
- Keeps the commands read pointer in retained registers and triggers system CPU for storing the last movement to external NVM
- Can be triggered by CPU, a divided version of the sleep clock or an external programmable event from RTC
- Supports any kind of wave generation on signals-in-single, pair or quad mode
- Generates Interrupt to system CPU in case of a FIFO underrun or overflow
- Operates on $pclk$ and uses the sleep clock for the pattern generation. Note that, the pattern generation clock should always be slower than $pclk$

Figure 47 shows a system overview of the Motor Controller.

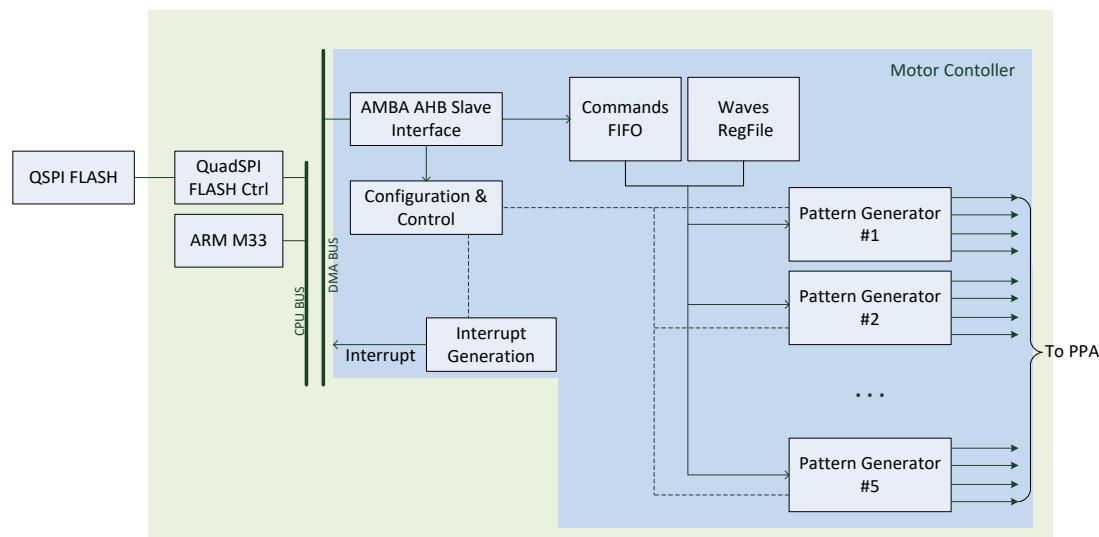


Figure 47: Motor Controller Block Diagram

21.2 Architecture

The architecture is comprised of a number of sub blocks. These are:

- **Pattern Generator (PG):** Each PG is a hardware state machine, which generates a signal waveform on the output. This is based on specific parameters needed to drive the motor of a watch's hand. Depending on the amount of hands that need to be supported, the DA1469x can support up to five PGs. Each Pattern Generator will control four signals that can generate a waveform according to specific commands and waveform formats
- **Commands FIFO and Waves RegFile:** The Commands FIFO and the Waves RegFile are basically a hardware state machine that combines commands of a certain format, written in a Command buffer with wave definitions, which represents the waveform each PG has to implement on the four different signals
- **Interrupt Generator:** This block is responsible for generating an interrupt
- **AMBA interface, Configuration and Control registers:** The registers that are supposed to control the flow and configure the engine according to the requirement parameters presented in the later section

21.2.1 Commands and Waves

Figure 48 explains the commands and waves format.

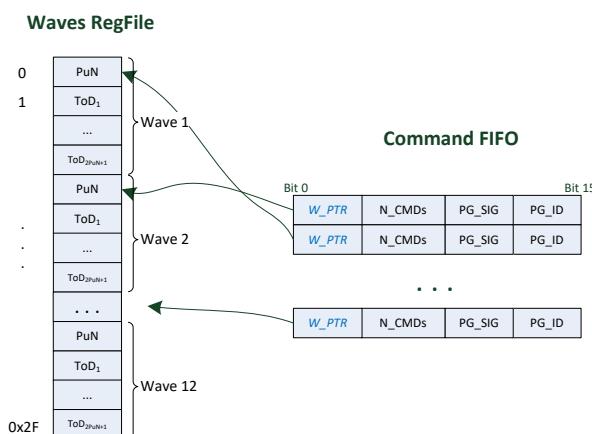


Figure 48: Commands and Waveform Format

The Commands FIFO is a memory mapped (for reading) RAM of 64x16bits. Writing this memory occurs in a FIFO mode (same address, Run through mode). Each Command FIFO word adheres to the following format:

Table 102: Commands Format

| Symbol | Bits | Description |
|--------|-------|---|
| W_PTR | 5:0 | This is a pointer to the waves memory space; supports up to 12 different waves. |
| N_CMDs | 10:6 | Defines the number of additional commands to be pushed to PGs on a single trigger. |
| PG_SIG | 12:11 | Pattern Generator signal; defines which signal (out of the four) is the command addressing. |
| PG_ID | 15:13 | Pattern Generator Identification; defines which PG the current command is for; supports up to 5 different PGs |

Multi-core Bluetooth 5.2 SoC family with system PMU

Each command addresses a Pattern Generator (PG), one of the four signals within this PG indicates if more commands should be pushed along with it to this specific PG, and finally points to the wave format that needs to be decoded and implemented by the PG.

The Command FIFO can be exercised in two modes:

1. **Run through mode:** The read pointer is continuously increased until the Command FIFO is empty.
2. **Cyclic mode:** The read pointer will revert to the beginning of the FIFO after reaching a programmable value, the maximum value is 63.

The Wave format, followed by the contents of the Waves RegFile, is presented below:

Table 103: Waves Format

| Symbol | Bits | Description |
|--------|--------|---|
| PuN | 4:0 | Pulses Number: Defines the amount of active high pulses that the wave contains; up to 23 pulses are supported. |
| ToDN | 5 bits | Toggle Duration: Defines the amount of time before the wave toggles from high to low, or low to high. The value indicates the number of "slots". A "slot" is a programmable value, a multiple of the 30.5 µs clock period. The amount of ToD, N is defined by the following formula: $N = 2^{*}PuN+1$. For example, if a PuN=2 is selected, 5 ToDs must be defined. Since the waves are always return-to-zero, the last ToD is not indicating toggling of the wave but just elapsed time from last toggle to the end of the wave. |

The waveform granularity which defines “the slot”, is a programmable value with a minimum of 30,5 µs per slot (based on the sleep clock period). The “slot” can be configured in form of five bits, therefore, maximum $32^{*}30.5\text{ }\mu\text{s}=976\text{ }\mu\text{s}$ (0 means $1^{*}30.5\text{ }\mu\text{s}$).

The maximum waveform time is $(2^{*}23+1) = 47$ ToDs. Each ToD can be 32 slots max., therefore, $47^{*}32\text{ }*488\text{ }\mu\text{s/slot} = 1468\text{ ms}$.

Minimum waveform time will be $(2^{*}1+1) = 3$ ToDs. Each ToD can be 1 slot min, hence $3^{*}1\text{ }*30\text{ }\mu\text{s/slot} = 90\text{ }\mu\text{s}$.

The commands will pop from the FIFO on every trigger. A trigger can be one of three below:

1. A SW writing to a register bit (auto-cleared).
2. A clock tick (a divided version of the sleep clock. The division is maximum 64, resulting in a ~1 ms trigger period).
3. An external signal coming from RTC, which can be programmed to be asserted on multiples of 10 ms

Upon a trigger, one or more commands will be read out of the FIFOs, depending on the N_CMDS field. The reading of the commands happens on the fast clock (sys_clk), while the wave generation happens on the slow clock (mc_clk: lp_clk divided by MC_CLK_DIV).

The read pointer of the commands FIFO is a register which consists of retention flip-flops (i.e. the content will not be lost if the power domain, where the Motor Controller resides, is powered down). Moreover, this pointer will not be reset by HW Reset, but by PORReset only. Hence, in cases of Brown-Out Detection events (HW reset) the value of the pointer will be retained.

Both read and write pointers are readable by the system CPU.

The waves should be loaded using the WAVETABLE_BASE register. The writing of the register should be done in 5-bit chunks, according to the four corresponding bit fields of the WAVETABLE_BASE register. Each 5-bit chunk is a PuN or ToD within the Waves RegFile. [Figure 49](#) describes the way of arranging and loading the waves into the Waves memory space:

Multi-core Bluetooth 5.2 SoC family with system PMU

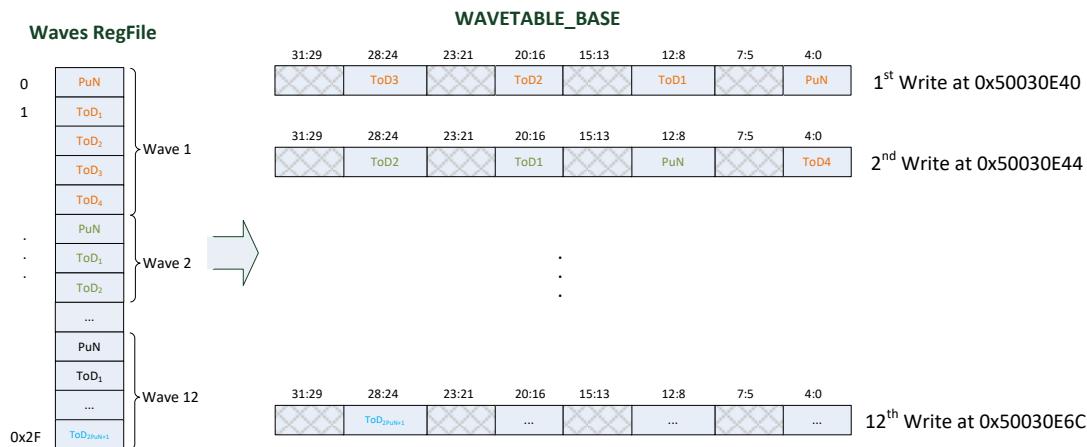


Figure 49: Loading Waves into Waves Memory Space

21.2.2 Pattern Generators (PGs)

All PGs monitor the output of the Commands FIFO & Wave memory. They all decode the PG_ID and if not addressed, they stop parsing; ignoring the rest of the message.

The PG addressed, will store the Waves table index of the start of the waveform. According to its programming, it should start implementing the waveform. It might also wait until one or more signal is instructed with commands, before it starts. The example in Figure 48 shows that two signals (SIG0 and SIG1) need a command before they start generating the waveform.

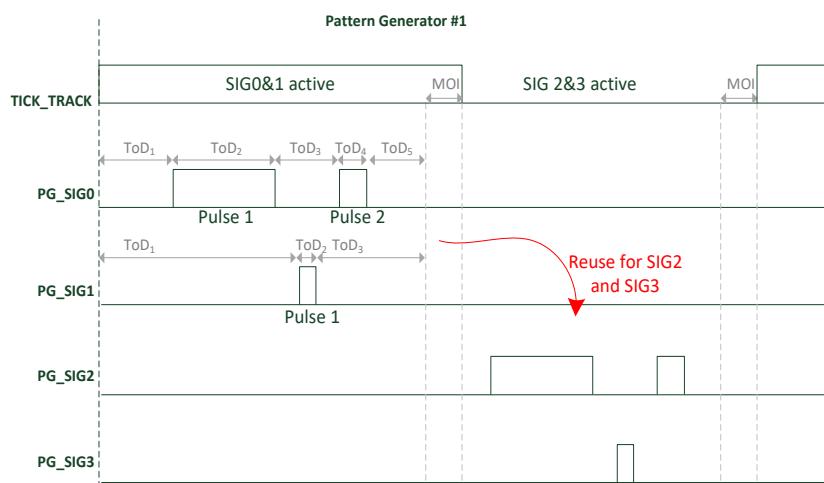


Figure 50: Waveform Generation in Signal Pairs

In the previous example, the waveform parameters for PG_SIG0 are: PuN=2, and the number of ToDs = 5. Another parameter that is configurable, is Mol (stands for **M**otor **I**dle) and indicates a number of slots where nothing happens. This time is always added at the end of a command and will be common for all five PGs. This parameter is 10 bits wide and 0 is allowed.

The PG can be configured to operate in “mirroring mode”, that is:

1. A single signal will replicate the wave of another one after Mol has elapsed.
2. A pair of signals will replicate the waves of the other pair after Mol has elapsed.

In the previous example, the PG is configured to operate in mirroring mode and more specifically in pairs.

Multi-core Bluetooth 5.2 SoC family with system PMU

As soon as the PG starts generating the waveform, a busy signal indicates to the Commands & Wave controller that this specific PG is busy, so no more waves can be received. If the next command were to address the same PG, it is discarded until the busy signals are de-asserted. If the next command(s) address other PGs, they are broadcasted. The busy signal should be de-asserted as soon as the waveform generation is finished.

21.2.3 Interrupt Generator

An interrupt is generated when:

1. A PG has just started the waveform implementation. The BUSY signal positive edge is used as a notification. This interrupt source is disabled in cases of hands animations.
2. Read pointer – Write pointer < Threshold, where threshold is a programmable value indicating the amount of commands still left in the Command FIFO.

The idea is that the CPU is notified after a successful start of the waveform, meaning the hand movement, so that the last state of the read pointer can be stored into a non-volatile FLASH. In case of a power failure, the system should be able to move the hand back to its last position. The following cases can be observed:

Table 104: Hand Position and Restoration Error

| Hand | Read Pointer Storage | Remarks |
|-----------------------|-----------------------|---|
| Started, not finished | Started, not finished | Hand will be restored at the actual position OR one tick away in case the wave has proceed enough to move the hand |
| Started, finished | Started, not finished | Hand will be restored one tick away from the real position |
| Started, not finished | Started, finished | Hand will be restored one tick away from the real position OR at the actual position in case the wave has proceed enough to move the hand |
| Started, finished | Started, finished | Hand will be restored at the actual position |

Storing the value of the Read Pointer into external FLASH, also requires some time, so does the restoration process. Hence, the above error of one tick is depending on the relation of the tick period and the write/read process from the FLASH.

21.3 Programming

There is a simple sequence of steps that needs to be followed to program the Motor Controller:

1. Select the clock source of the system by writing the CLK_CTRL_REG[SYS_CLK_SEL] bit field and the low power (sleep) clock source by writing the CLK_CTRL_REG[LP_CLK_SEL] bit field.
2. Setup the clock divider for the motor controller slot using CLK_PER_REG[MC_CLK_DIV] bit field.
3. Configure GPIOs used for the Pattern Generator signals by writing the appropriate Px_yy_MODE_REG[PID] = 43 (PG).
4. Setup triggers for automatic mode (SMOTOR_CTRL_REG[TRIG_RTC_EVENT_EN] or SMOTOR_CTRL_REG[MC_LP_CLK_TRIG_EN]).
5. Setup FIFO operation mode (normal or cyclic) using SMOTOR_CTRL_REG[CYCLIC_MODE] bit field.
6. Setup the FIFO level thresholds (SMOTOR_CTRL_REG[SMOTOR_THRESHOLD]).
7. Setup IRQs depending on the case: for example, SMOTOR_CTRL_REG[SMOTOR_THRESHOLD_IRQ_EN], SMOTOR_CTRL_REG[SMOTOR_GENSTART_IRQ_EN], SMOTOR_CTRL_REG[SMOTOR_GENEND_IRQ_EN], PGx_CTRL_REG[GENSTART_IRQ_EN], PGx_CTRL_REG[GENEND_IRQ_EN]

Multi-core Bluetooth 5.2 SoC family with system PMU

8. Setup FIFOs IRQs depending on the case: for example,
SMOTOR_CTRL_REG[SMOTOR_FIFO_UNR_IRQ_EN],
SMOTOR_CTRL_REG[SMOTOR_FIFO_OVF_IRQ_EN]
9. Setup pattern generator signals triggers PGx_CTRL_REG[PG_START_MODE].
10. Load Waves memory (WAVETABLE_BASE) with the available wave patterns.
11. Enable the Motor Controller block by setting the CLK_PER_REG[MC_CLK_EN] bit.
12. Enable Pattern Generator signals by setting the PGx_CTRL_REG[SIGx_EN] bit.
13. Push commands in to the FIFO (SMOTOR_CMD_FIFO_REG).
14. Generate patterns manually by popping commands (SMOTOR_TRIGGER_REG) or automatically by the configured trigger sources (RTC or divided sleep clock).

22 LCD Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | x | ✓ | ✓ | ✓ |

22.1 Introduction

The DA1469x is equipped with a LCD Controller, capable of supporting Parallel and Serial (SPI3/4) interfaces.

A global register file controls the parameters of the controller. After applying timing parameters, a dedicated DMA engine fetches data. Depending on the target screen, output can be formatted to different types as described in the following sections.

The architecture of the LCD Controller is depicted in [Figure 51](#).

Features

- 1-layer support
- Dedicated DMA engine
- Parallel and Serial (SPI3/4) I/Fs support
- Programmable IRQ

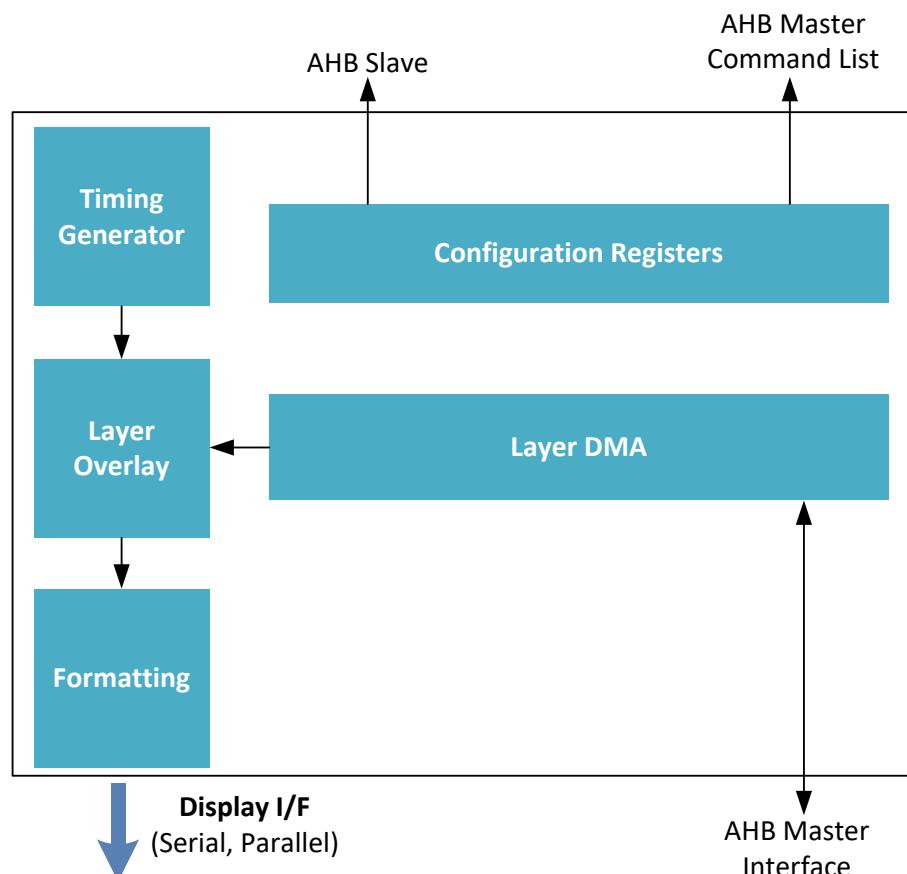


Figure 51: LCD Controller Block Diagram

22.2 Architecture

22.2.1 Parallel LCD Interfaces

22.2.1.1 HSYNC / VSYNC Parallel Interface

The signal HSYNC is asserted on every scanline and the signal VSYNC is asserted on every frame. The polarity of the output *clock* signal can be defined as positive or negative, and the length of each pulse is programmable. Typical waveforms of operation are depicted in [Figure 52](#) and [Figure 53](#).

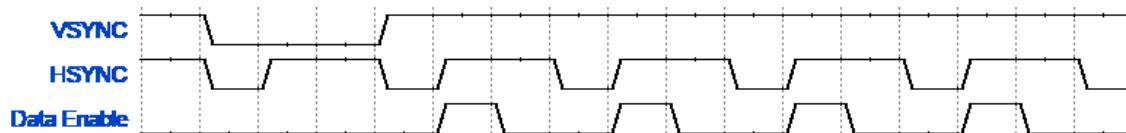


Figure 52: HSYNC/VSYNC Typical Waveform

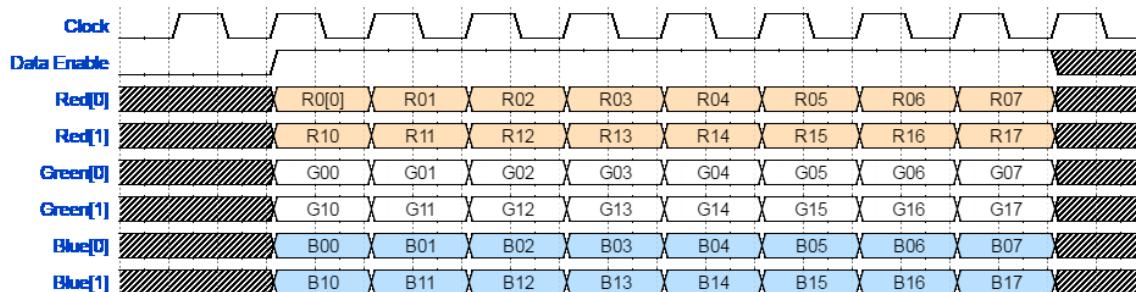


Figure 53: HSYNC/VSYNC Color Bits Waveform

The available signals of the interface are shown in [Table 105](#).

Table 105: HSYNC/VSYNC Parallel I/F Pinout

| Pin Name | Type | Description | Source |
|------------------|--------|---|----------------|
| CLK | Output | LCD Clock | LCD Controller |
| ENAB | Output | Indicate valid data on the LCD data bus | LCD Controller |
| HSYNC | Output | Reset the LCD column pointer to the edge of the display | LCD Controller |
| VSYNC | Output | Reset the LCD row pointer to top of the display | LCD Controller |
| RED0 (DATA[0]) | Output | Red image data | LCD Controller |
| RED1 (DATA[1]) | Output | Red image data | LCD Controller |
| GREEN0 (DATA[2]) | Output | Green image data | LCD Controller |
| GREEN1 (DATA[3]) | Output | Green image data | LCD Controller |
| BLUE0 (DATA[4]) | Output | Blue image data | LCD Controller |
| BLUE1 (DATA[5]) | Output | Blue image data | LCD Controller |
| Backlight | Output | LED driver signals | GPIO |

22.2.1.2 JDI Parallel Interface

The available signals of the interface are shown in [Table 106](#).

Table 106: JDI Parallel I/F Pinout

| Pin Name | Type | Description | Source |
|---------------------------------|--------|---|----------------|
| HST | Output | Start signal for the horizontal driver | LCD Controller |
| HCK | Output | Shift clock for the horizontal driver | LCD Controller |
| ENB | Output | Write enable signal for the pixel memory | LCD Controller |
| VST | Output | Start signal for the vertical driver | LCD Controller |
| XRST | Output | Reset signal for the horizontal and vertical driver | LCD Controller |
| VCK | Output | Shift clock for the vertical driver | LCD Controller |
| RED0 | Output | Red image data | LCD Controller |
| RED1 | Output | Red image data | LCD Controller |
| GREEN0 | Output | Green image data | LCD Controller |
| GREEN1 | Output | Green image data | LCD Controller |
| BLUE0 | Output | Blue image data | LCD Controller |
| BLUE1 | Output | Blue image data | LCD Controller |
| VCOM (Note 1) | Output | Common electrode driving signal (60Hz) | MCU GPIO |
| FRP (Note 1) | Output | Liquid crystal driving signal (60Hz) | MCU GPIO |
| XFRP | Output | Liquid crystal driving signal (60Hz inverted) | MCU GPIO |
| Backlight | Output | LED driver signals | MCU GPIO |

22.2.2 Serial LCD Interfaces

22.2.2.1 SPI3/4 Serial Interface

The LCD controller supports the SPI serial interface with three or four distinct signals. The 3-lines serial interface use the CS (chip enable) signal, the SCLK (serial clock) signal and the DO (MOSI, serial data output) signal. The 4-lines serial interface use an additional signal, the DCS (data/command select). The DO signal is regarded as a command when DCS is low and as data when DCS is high. The serial clock (SCLK) can be stopped when no communication is necessary.

During the write mode, the LCD controller sends the write commands and data to the LCD display. In the 4-lines serial mode, the data packet (DO) contains only the transmission byte (8 bits). The control bit is transferred by the DCS signal. In the 3-lines serial mode, the data packet (DO) contains a control bit and a transmission byte (9 bits). In both cases, the MSB is transmitted first.

The CS signal is configurable. It can be set high or low to indicate the start of the data transmission. Data can be sampled either by the falling or the rising edge of the SCLK, depending on the configuration. The clock polarity is also configurable (clock starts at high or low edge).

If the CS signal remains active (high or low, depending the configuration) after the last bit of the transmitted data packet, the LCD controller will transmit the MSB of the next byte at the next rising/falling edge of SCLK.

The available signals of the interface are shown in [Table 107](#).

Table 107: SPI3/4 Serial I/F Pinout

| Pin Name | Type | Description | Source |
|----------------------|--------|----------------------------------|----------------|
| SCLK | Output | Serial clock signal | LCD Controller |
| MOSI | Output | Serial data | LCD Controller |
| CS | Output | Chip select | LCD Controller |
| DCS | Output | Data/Command select (when SPI4) | LCD Controller |
| EXTCOMIN (Note 1) | Output | COM Inversion Signal Input (1Hz) | MCU GPIO |
| RST | Output | Reset display | MCU GPIO |
| DISP | Output | Display ON/OFF Control | MCU GPIO |
| Backlight | Output | LED driver signals | MCU GPIO |

22.2.2.2 JDI SPI Serial Interface

The available signals of the interface are shown in [Table 108](#).

Table 108: JDI SPI Serial I/F Pinout

| Pin Name | Type | Description | Source |
|----------------------|--------|--|----------------|
| SCLK | Output | Serial clock signal | LCD Controller |
| MOSI | Output | Serial data | LCD Controller |
| CS | Output | Chip select | LCD Controller |
| DCS | Output | Data/Command select (when SPI4) | LCD Controller |
| TE | Input | Tearing effect. Use to synchronize MCU to frame memory writing | LCD Controller |
| EXTCOMIN (Note 1) | Output | COM Inversion Signal Input (1Hz) | MCU GPIO |
| RST | Output | Reset display | MCU GPIO |
| DISP | Output | Display ON/OFF Control | MCU GPIO |
| Backlight | Output | LED driver signals | MCU GPIO |

Note 1 CLK_SYS_REG[LCD_EXT_CLK_SEL]. Note that the clock output when CLK_SYS_REG[LCD_EXT_CLK_SEL] is set is available in sleep only if the PD_COM is kept active

Multi-core Bluetooth 5.2 SoC family with system PMU

22.2.3 Color Input Formats

The following color input formats are supported by the controller:

Table 109: L1 Grayscale/Palette

| |
|---|
| L |
|---|

Note 2 Values are 0 (white) and 1 (black).

Table 110: L4 Grayscale/Palette

| | | | |
|----|----|----|----|
| L3 | L2 | L1 | L0 |
|----|----|----|----|

Note 3 Values range from 0 (black) to 16 (white).

Table 111: L8 Grayscale/Palette

| | | | | | | | |
|----|----|----|----|----|----|----|----|
| L7 | L6 | L5 | L4 | L3 | L2 | L1 | L0 |
|----|----|----|----|----|----|----|----|

Note 4 Values range from 0 (black) to 255 (white).

Table 112: 8-bit RGB332

| | | | | | | | |
|----|----|----|----|----|----|----|----|
| R2 | R1 | R0 | G2 | G1 | G0 | B1 | B0 |
|----|----|----|----|----|----|----|----|

Table 113: 16-bit RGB565

| | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R4 | R3 | R2 | R1 | R0 | G5 | G4 | G3 | G2 | G1 | G0 | B4 | B3 | B2 | B1 | B0 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Table 114: 16-bit RGBX5551

| | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R4 | R3 | R2 | R1 | R0 | G4 | G3 | G2 | G1 | G0 | B4 | B3 | B2 | B1 | B0 | A0 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Table 115: 32-bit RGBX8888

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 116: 32-bit XRGB8888

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| X | X | X | X | X | X | X | X | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Table 117: 32-bit ABGR8888

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Table 118: 32-bit BGRA8888

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

22.2.4 Color Output Formats

22.2.4.1 SPI Output Formats

The supported SPI output data formats are depicted below:

Table 119: SPI3/4 – 8-bit RGB-332

| Byte | Pixel | DCS | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----------|----|----|----|----|----|----|----|----|
| Byte1 | n pixel | (Note 5) | R2 | R1 | R0 | G2 | G1 | G0 | B1 | B0 |

Table 120: SPI3/4 – 8-bit RGB-444

| Byte | Pixel | DCS | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|-------------|----------|----|----|----|----|----|----|----|----|
| byte0 | n pixel | (Note 5) | R3 | R2 | R1 | R0 | G3 | G2 | G1 | G0 |
| byte1 | n/n+1 pixel | (Note 5) | B3 | B2 | B1 | B0 | R3 | R2 | R1 | R0 |
| byte2 | n+1 pixel | (Note 5) | G3 | G2 | G1 | G0 | B3 | B2 | B1 | B0 |

Table 121: SPI3/4 – 8-bit RGB-565

| Byte | Pixel | DCS | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----------|----|----|----|----|----|----|----|----|
| byte0 | n pixel | (Note 5) | R4 | R3 | R2 | R1 | R0 | G5 | G4 | G3 |
| byte1 | n pixel | (Note 5) | G2 | G1 | G0 | B4 | B3 | B2 | B1 | B0 |

Table 122: SPI3/4 – 8-bit RGB-666

| Byte | Pixel | DCS | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----------|----|----|----|----|----|----|----|----|
| byte0 | n pixel | (Note 5) | R5 | R4 | R3 | R2 | R1 | R0 | | |
| byte1 | n pixel | (Note 5) | G5 | G4 | G3 | G2 | G1 | G0 | | |
| byte2 | n pixel | (Note 5) | B5 | B4 | B3 | B2 | B1 | B0 | | |

Table 123: SPI3/4 – 8-bit RGB-888

| Byte | Pixel | DCS | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----------|----|----|----|----|----|----|----|----|
| byte0 | n pixel | (Note 5) | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| byte1 | n pixel | (Note 5) | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 |
| byte2 | n pixel | (Note 5) | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |

Note 5 Available only in SPI3 mode.

22.2.4.2 JDI SPI Output Formats

The supported JDI SPI output data formats are depicted below:

Table 124: JDI SPI – 8-bit RGB-111-1 (3-bit mode)

| Byte | Pixel | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----|----|----|----|----|----|----|----|
| byte0 | n pixel | | | R0 | G0 | B0 | R1 | G1 | B1 |
| byte1 | n pixel | | | R2 | G2 | B2 | R3 | G3 | B3 |
| byte2 | n pixel | | | R4 | G4 | B4 | R5 | G5 | B5 |

Note 6 The Data order is as follows, MSB = D7, LSB = D0. Pixel Data is MSB = R0, LSB = B5 for Red, Green, Blue data.

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 125: JDI SPI – 8-bit RGB-111-2 (4-bit mode)

| Byte | Pixel | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----|----|----|----|----|----|----|----|
| byte0 | n pixel | R0 | G0 | B0 | | R1 | G1 | B1 | |
| byte1 | n pixel | R2 | G2 | B2 | | R3 | G3 | B3 | |
| byte2 | n pixel | R4 | G4 | B4 | | R5 | G5 | B5 | |

Note 7 The Data order is as follows, MSB = D7, LSB = D0. Pixel Data is MSB = R0, LSB = Dummy data for Red, Green, Blue and dummy data.

Table 126: JDI SPI – 8-bit RGB-111-3 (3-bit mode)

| Byte | Pixel | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----|----|----|----|----|----|----|----|
| byte0 | n pixel | R0 | G0 | B0 | R1 | G1 | B1 | R2 | G2 |
| byte1 | n pixel | B2 | R3 | G3 | B3 | R4 | G4 | B4 | R5 |
| byte2 | n pixel | G5 | B5 | R6 | G6 | B6 | R7 | G7 | B7 |

Note 8 The Data order is as follows, MSB = D7, LSB = D0. Pixel Data is MSB = R0, LSB = B7 for Red, Green, Blue data.

Table 127: JDI SPI – 8-bit RGB-111-4 (1-bit mode)

| Byte | Pixel | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-------|---------|----|----|-----|-----|-----|-----|-----|-----|
| byte0 | n pixel | d0 | d1 | d2 | d3 | d4 | d5 | d6 | d7 |
| byte1 | n pixel | d8 | d9 | d10 | d11 | d12 | d13 | d14 | d15 |

Note 9 The Data order is as follows, MSB = D7, LSB = D0. Pixel Data is MSB = d0, LSB = d15 for black and white data, where '0' indicates black and '1' indicates white.

22.3 Programming

Refer to the SDK for the full featured LCD Software Library.

23 DMA Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

23.1 Introduction

The DMA controller has 8 Direct Memory Access (DMA) channels for fast data transfers to and from SPI, UART, I2C, SRC, PCM, USB, GP_ADC and SD_ADC to and from any on-chip RAM. The DMA controller off-loads the Arm interrupt rate, if an interrupt is generated after a programmable amount of transfers. A number of peripheral requests is multiplexed on the 8 available channels, to increase utilization of the DMA. The block diagram of the DMA controller is depicted in [Figure 36](#).

Features

- Eight channels with optional peripheral trigger
- Full 32-bit source and destination pointers
- Flexible interrupt generation
- Programmable transfer length
- Flexible peripheral request per channel
- Option to initialize memory
- Programmable Edge-Sensitive request support
- Programmable support of AHB burst reads/writes, supporting both Memory-to-Memory and Memory-to-Peripheral transfers
 - Burst lengths supported are 8-beat (INCR8) and 4-beat (INCR4)
- Programmable bus error detection support and IRQ generation upon detection
- FREEZE support also in on-going Memory-to-Memory transfers
- Support of “secure transfer” mode by a dedicated, conditionally secure DMA channel (DMA7)

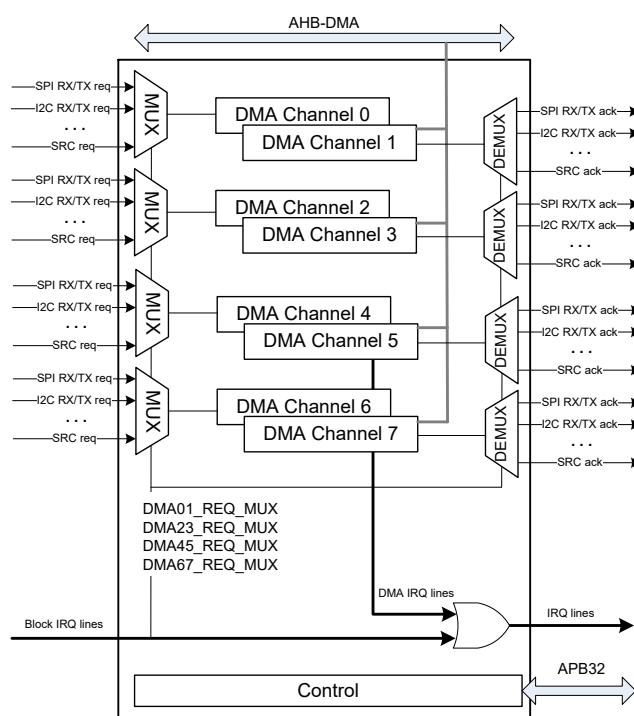


Figure 54: DMA Controller Block Diagram

23.2 Architecture

23.2.1 DMA Peripherals

The list of peripherals that can request a DMA service, is presented in [Table 128](#).

Table 128: DMA served peripherals

| Block | Direction(s) | Supported Access Rate |
|------------|--------------|---------------------------------|
| SPI | RX & TX | Single |
| SPI2 | RX & TX | Single |
| QSPI RAM | Read & Write | Burst |
| QSPI FLASH | Read | Burst |
| UART | RX & TX | Single/Burst |
| UART2 | RX & TX | Single/Burst |
| UART3 | RX & TX | Single/Burst |
| I2C | RX & TX | Single/Burst |
| I2C2 | RX & TX | Single/Burst |
| USB | RX & TX | Single |
| RAM | Read & Write | Burst |
| GP_ADC | Read | Single |
| SD_ADC | Read | Single |
| SRC | Left & Right | Single/Burst (Left or Right) |
| PCM | Left & Right | Single |

The Edge-sensitive requests cannot be used when DMA services the USB Rx path (USB-to-Memory). The same applies in the case of the SPI_CTRL_REG[SPI_PRIORITY] = '1' (either Rx or Tx FIFO is enabled). Nevertheless, it is strongly recommended for the case of USB Tx path (Memory-to-USB) and when serving the UART and I2C peripherals (Memory-to-Peripheral).

23.2.2 Input/Output Multiplexer

The multiplexing of peripheral requests is controlled by DMA_REQ_MUX_REG. So, if DMA_REQ_MUX_REG[DMAxy_SEL] is set to a certain (non-reserved) value, the TX/RX request from the corresponding peripheral will be routed to DMA channels x (TX request) and y (RX request) respectively.

Similarly, an acknowledging de-multiplexing mechanism is applied.

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

23.2.3 DMA Channel Operation

A DMA channel is switched on with bit DMA_ON. This bit is automatically reset if the DMA transfer is finished. The DMA channels can either be triggered by software, or by a peripheral DMA request. If DREQ_MODE is 0, then a DMA channel is immediately triggered. If DREQ_MODE is 1 the DMA channel can be triggered by a Peripheral request.

If 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 bits counter DMAx_IDX_REG, which is set to 0 if

Multi-core Bluetooth 5.2 SoC family with system PMU

the DMA transfer starts and is compared with the DMA_LEN_REG after each transfer. The register value is multiplied according to the AINC, BINC and BW values before it is added to DMA_B_START_REG and DMA_B_START_REG. AINC or BINC must be 0 for register access.

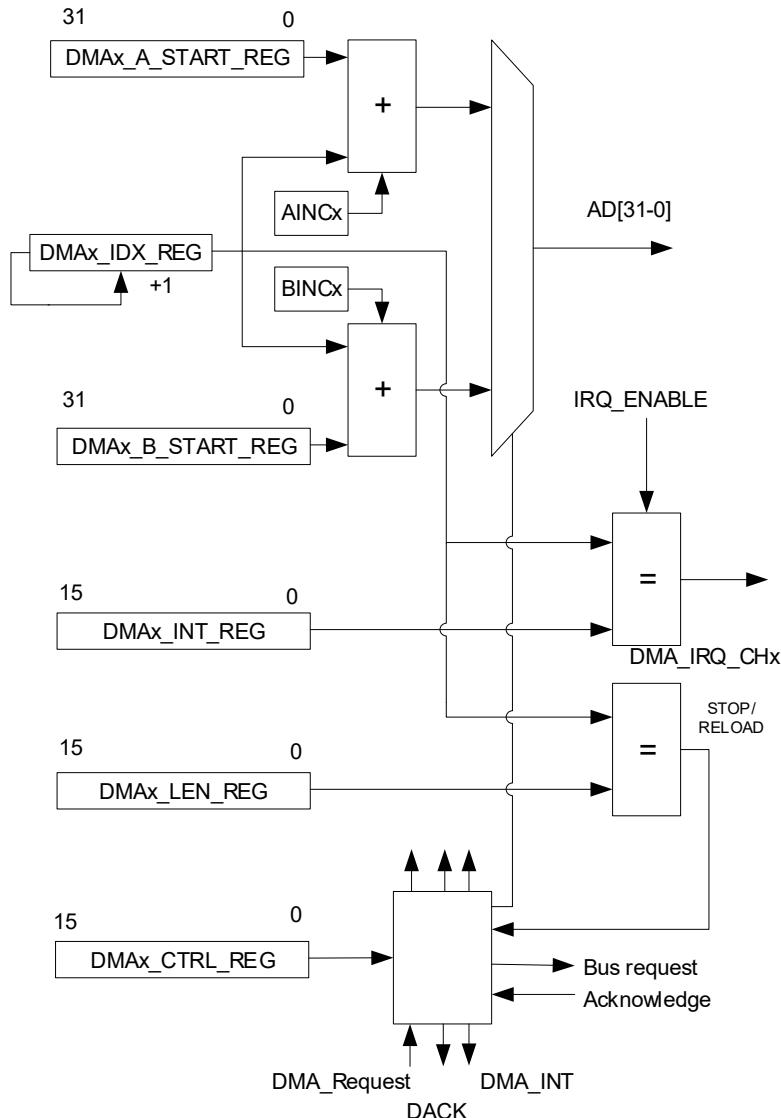


Figure 55: DMA channel diagram

If, at the end of a DMA cycle, the DMA start condition is still true, the DMA continues. The DMA stops when the transfer length is reached and, either DREQ_MODE is low, or DMAx_LEN_REG is equal to the internal index register. This condition also clears the DMA_ON bit.

If 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 Cortex-M33. If the DMA controller is started with DREQ_MODE = 0, the DMA will always stop, regardless of the state of CIRCULAR.

Each DMA channel can generate an interrupt if DMAx_INT_REG is equal to DMAx_IDX_REG. After the transfer and before DMAx_IDX_REG is incremented, the interrupt is generated.

Example: if DMA_x_INT_REG=0 and DMA_x_LEN_REG=0, there will be one transfer and an interrupt.

It should be noted that there is no hardware protection from erroneous programming of the DMA registers. However, the software can be notified by an interrupt when a bus error has occurred in either the read or write cycle of the DMA transfer, if DMAx_CTRL_REG[BUS_ERROR_DETECT] is set.

23.2.4 DMA Arbitration

The priority level of a DMA channel can be set with bits DMA_PRIO[2:0]. These bits determine which DMA channel will be activated in case 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 channel 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 channel, until the transfer is completed, regardless if DMA_IDLE is set. The purpose of DMA_INIT is to initialize a specific memory block with a certain value, fetched also from memory, without any interruption from other active DMA channels that may request the bus at the same time. Consequently, it should be used only for memory initialization, while when the DMA transfers data to/ from peripherals, it should be set to '0'. Note that AINC must be set to '0' and BINC to '1', when DMA_INIT is enabled.

It should be noted that memory initialization could also be performed without having the DMA_INIT enabled and by simply setting AINC to '0' and BINC to '1', provided that the source address memory value will not change during the transfer. However, it is not guaranteed that the DMA transfer will not be interrupted by other channels of higher priority, when these request access to the bus at the same time.

23.2.5 Freezing DMA Channels

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

To enable the channels again, a 1 to bits at the RESET_FREEZE_REG must be written.

It is noted that the on-going Memory-to-Memory transfers (DREQ_MODE='0') can also be interrupted (freeze).

23.2.6 Secure DMA Channel

If the security flag is enabled in the OTP header, then DMA channel #7 becomes a secure channel. This channel is then only used to move keys from the Symmetric Key Area to the AES block for encryption/decryption or the QSPI FLASH Controller for on-the-fly decryption without the CPU being able to intervene.

23.3 Programming

There is a simple sequence of steps that needs to be followed to configure the DMA Controller.

23.3.1 Memory to Memory Transfer

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. Enable the IRQ of the used DMA channel if needed (DMA_INT_MASK_REG).
6. 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[BURST_MODE]: Enable DMA read/write bursts, if needed.
7. Start the DMA transfer by setting the DMAx_CTRL_REG[DMA_ON] bit.
8. Wait until transfer is finished (DMAx_CTRL_REG[DMA_ON] = 0) or the corresponding interrupt.
9. Clear the IRQ if it was enabled (DMA_INT_STATUS_REG).

23.3.2 Peripheral to Memory Transfer

1. Set the length of data to be transferred (DMAx_LEN_REG).
2. Set the source address (DMAx_A_START_REG) equal to the peripheral Rx register (e.g. I2C_DATA_CMD_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. 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 channel priority.
 - f. DMAx_CTRL_REG[BURST_MODE]: Enable DMA read/write bursts, if needed.
Note that SPI does not support burst transfers.
7. Enable the IRQ of the used DMA channel (DMA_INT_MASK_REG).
8. Start the DMA transfer by setting the DMAx_CTRL_REG[DMA_ON] bit.
9. Enable peripheral's DMA request (e.g. I2C_DMA_CR_REG[TDMAE]).

24 Crypto Engine

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

24.1 Introduction

The Crypto engine aims to accelerate the algorithm calculations that are needed in order to implement the RFC4835. It implements AES in ECB, CBC and CTR modes. It also comprises HASH functions (SHA-1, SHA224/256/384/512, MD5). It supports AES128, AES 256 as well as HMAC-SHA-256 authentication protocol.

The AES/HASH engine uses a DMA engine for transferring encrypted/decrypted data to a shared memory in the AHB bus. The control registers of the IP are connected to the AHB bus.

The AES/HASH engine gives more flexibility to the way input data can be provided to the module. A calculation can be applied on fragmented input data, but not on data residing at a specific memory space, by means of successive register programming in the internal DMA engine.

Features

- AES (Advanced Encryption Standard) with 128, 192 or 256 bits key cryptographic algorithm
- HASH functions: MD5, SHA-1, SHA224/256/384/512 bits
- Modes of operation
 - ECB (Electronic Code Book)
 - CBC (Cipher Block Chaining)
 - CTR (Counter)
- AHB Master DMA machine for data manipulation
- AHB Slave register file for configuration

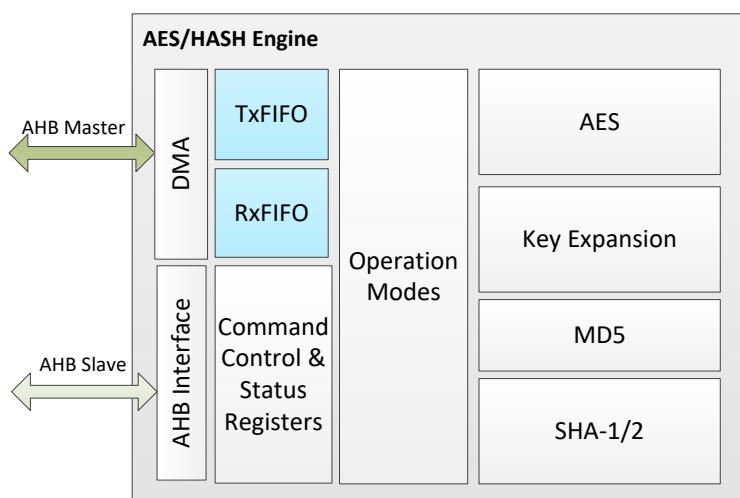


Figure 56: AES/HASH Architecture

24.2 Architecture

The AES/HASH includes a DMA engine (AHB Master Interface) for transferring data between the IP and a shared memory. The control registers of the AES/HASH are connected to AHB interface (AHB Slave interface).

The “Modes” controls the AES by implementing the respective mode that the selected encryption algorithms will operate each time. Also, “Modes” communicates with DMA via two FIFO’s (RxFIFO and TxFIFO), which isolate the operation of the AES/HASH IP from the current status of the AHB-

Multi-core Bluetooth 5.2 SoC family with system PMU

AMBA bus and also enable parallel transmission of data in bursts. By using burst transmission, the bus is utilized better because the bus access requests are reduced.

The “Ctrl FSM” block checks the FIFO’s and DMA status continuously and decides for: the amount of data traffic, plus which of the FIFO’s will be used. It also decides the “switching off” of the AES/HASH after transferring all results to the memory. The “HASH” block contains all the logic required for the realization of the hash algorithms calculations, as well as circuitry for the padding of data. It also contains glue logic for the transfer of the results to the “Modes” block.

24.2.1 AES

This part of the architecture implements the AES algorithm described in the AES-FIPS PUB 197. The capability it offers, is the encryption and decryption of 128 bits data blocks by using 128, 192 or 256 bits encryption key.

24.2.2 Operation Modes

The block “Modes” uses the AES in order to implement the following modes of operations:

- ECB (Electronic Code Book)
- CBC (Cipher Block Chaining)
- CTR (Counter)

Padding requirements of the algorithms, to convert all data to multiples of 16 bytes (for AES), must be addressed by software.

By applying successive programming of AES-CBC encryptions using software, the realization of the HMAC-XCBC-AES-96 algorithm is possible.

The implementation of the AES-CCM is feasible, just like the implementation of AES-CTR algorithms for encryption, and AES-CBC for authentication.

24.2.3 HASH

Padding at the input data is automatically applied as required by the hash algorithms. Two types of padding are implemented, due to the different algorithms supported. The purpose is to ensure that the message is a multiple of 512 or 1024 bits, depending on the algorithm. Padding is done in a similar way in both cases. After the last data byte, one extra byte of value 0x80 is added. Next, a number of bytes (0x00) is added, so that the overall size of the data block (including the extra bytes) mod 512/1024 is 448/896, depending on the algorithm. Following that, a 64/128-bits big-endian number is attached, which represents the size of the data block, in bits (without the padding). While in this process, TX/RX FIFOs are switched into 8-bytes mode.

The block packetizes the algorithm result (128 to 512 bits) into blocks of 64 bytes, so that they can be shifted to the TX FIFO. The following hash algorithms are implemented:

1. MD5: RFC1321.
2. SHA-1: FIPS PUB 180-4.
3. SHA-224/256: FIPS PUB180-4. In this case only initialization changes.
4. SHA-384/512: FIPS PUB 180-4.

24.3 Programming

24.3.1 AES Engine Programming

There is a simple sequence of steps that needs to be followed to program the AES Engine:

5. Enable the clock by setting the CLK_AMBA_REG[AES_CLK_ENABLE] bit.
6. Select the AES mode (CRYPTO_CTRL_REG[CRYPTO_HASH_SEL] = 0).

Multi-core Bluetooth 5.2 SoC family with system PMU

7. Define the mode of operation of the AES algorithm (CRYPTO_CTRL_REG[CRYPTO_ALG_MD]). For the CBC/CTR mode of operation the corresponding IV/CTR block should be defined in the CRYPTO_MREGx_REG registers.
8. Select the AES algorithm (CRYPTO_CTRL_REG[CRYPTO_ALG]).
9. Select the size of AES Key (CRYPTO_CTRL_REG[CRYPTO_AES_KEY_SZ]).
10. Use AES keys expansion if needed (CRYPTO_CTRL_REG[CRYPTO_AES_KEXP]). If the key expansion is enabled, define the (basic) key in the local CRYPTO_KEYS_START memory.
11. Setup data fetching address (CRYPTO_FETCH_ADDR_REG).
12. Setup data destination address (CRYPTO_DEST_ADDR_REG).
13. Set data length (CRYPTO_LEN_REG).
14. Select to Encrypt or Decrypt (CRYPTO_CTRL_REG[CRYPTO_ENCDEC]).
15. Enable the process by setting the CRYPTO_START_REG[CRYPTO_START] bit.
16. Wait the process to finish (CRYPTO_STATUS_REG[CRYPTO_INACTIVE] = 1).

24.3.2 HASH Engine Programming

There is a simple sequence of steps that needs to be followed to program the HASH Engine:

1. Enable the clock by setting the CLK_AMBA_REG[AES_CLK_ENABLE] bit.
2. Select the HASH mode (CRYPTO_CTRL_REG[CRYPTO_HASH_SEL] = 1).
3. Set the number of bytes which will be saved at the memory by the DMA (CRYPTO_CTRL_REG[CRYPTO_HASH_OUT_LEN], see register description).
4. Define the mode of operation of the HASH algorithm (CRYPTO_CTRL_REG[CRYPTO_ALG_MD], see register description).
5. Select the HASH algorithm (CRYPTO_CTRL_REG[CRYPTO_ALG], see register description).
6. Setup data fetching address (CRYPTO_FETCH_ADDR_REG).
7. Setup data destination address (CRYPTO_DEST_ADDR_REG).
8. Set data length (CRYPTO_LEN_REG).
9. Enable the process by setting the CRYPTO_START_REG[CRYPTO_START] bit.
10. Wait the process to finish (CRYPTO_STATUS_REG[CRYPTO_INACTIVE] = 1).

For both cases, an interrupt can be enabled upon the completion of the processing, by setting 1 to the CRYPTO_IRQ_EN (the interrupt should be enabled also in the CPU).

The clock of the Crypto Block should be disabled after the completion of each operation, by clearing the CLK_AMBA_REG[AES_CLK_ENABLE] bit, if there is no other operation to be performed.

25 True Random Number Generator

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

25.1 Introduction

The TRNG is a non-deterministic Random Number generator used to provide the seed for encryption processes. Its output can be used as entropy input for a FIPS 140-2 approved deterministic random number generation process which is handled by SW and HW accelerators of the DA1469x.

The TRNG contains oscillator rings in digital logic. When combined, they create metastability on a Flip-Flop that eventually becomes the source of the entropy bits.

Features

- Optional NIST SP800-90A Hash_DRBG post processing, with SHA-256 function that uses the on-chip HW accelerators
- Random numbers access through 32x32 bits FIFO on AHB bus
- Dedicated TRNG_IRQ Interrupt line
- Start-up time is 512 pclk cycles per 32 random bits
- Clock enable signals for optimal power saving

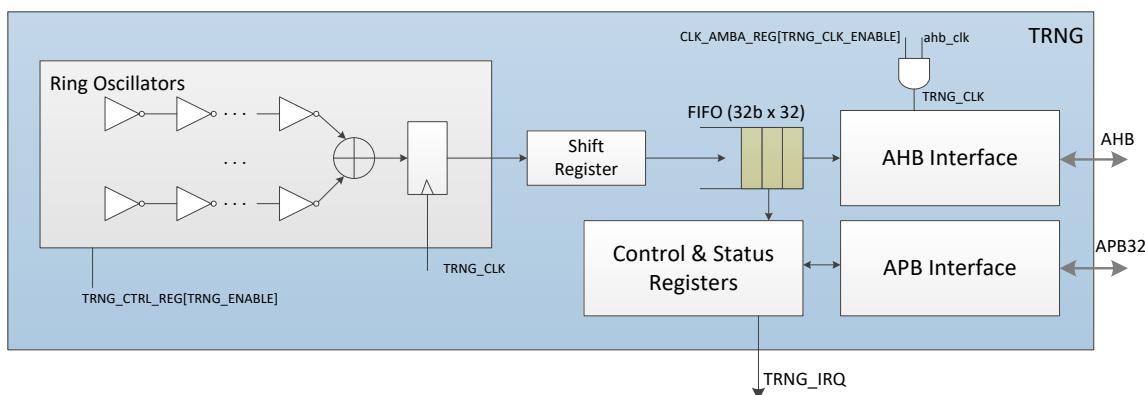


Figure 57: TRNG Block Diagram

25.2 Architecture

The TRNG is made up of a number of oscillator rings consisting of several inverters each. The output of the oscillator rings is accumulated in a shift register for whitening, before it is stored in a 32x32 bits deep FIFO. The oscillator rings are enabled when the TRNG_CTRL_REG[TRNG_ENABLE] is set and the FIFO is not full.

The 32-bit random numbers are accessible via an AHB interface, while the registers are accessible via the APB32 interface.

25.3 Programming

There is a simple sequence of steps that needs to be followed to program the TRNG engine:

1. Set CLK_AMBA_REG[TRNG_CLK_EN]=1 to enable the AHB bus access.
2. Set TRNG_CTRL_REG[TRNG_ENABLE]=1 to start the random number generation. This signal is ignored when the FIFO is already full.

Multi-core Bluetooth 5.2 SoC family with system PMU

3. Poll TRNG_FIFOLVL_REG which provides the amount of data in the FIFO, or wait for the TRNG_IRQ.
4. Read the random number from the TRNG FIFO at address 0x30050000 (TRNG_M)
5. To save power, set TRNG_CLK_EN=0 and set TRNG_ENABLE=0. The FIFO can only be accessed if TRNG_CLK_EN=1.

Note that, all signals are handled following the little-endian format. That means that the Least Significant Byte (LSB) is stored at the lowest address.

After TRNG_CTRL_REG[TRNG_ENABLE] is set, 512 pclk clock cycles will be needed per FIFO entry to be filled. So, for 32 FIFO entries $512 \times 32 = 16K$ pclk clock cycles or 500 μ s if the 32 MHz clock is used.

26 Wake-Up Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

26.1 Introduction

The Wake-Up Controller can be programmed to wake up the DA1469x from the Extended/Deep Sleep (clocked) mode as well as from Hibernation (clockless) mode. It consists of two parallel circuits that can be programmed regarding the GPIO edge monitored one with a debouncing counter usually used for buttons and another that indicates which GPIO has toggled.

The block diagram illustrating the Wake-Up function is depicted in [Figure 58](#).

Features

- Monitors any GPIO state change
- Implements debouncing time from 0 up to 63 ms
- Latches the status of the monitored lines
- Generates three interrupts to Arm Cortex-M33
- Generates a signal bus towards PDC indicating which GPIO has toggled
- Wakes up the system from Hibernation, Deep or Extended Sleep

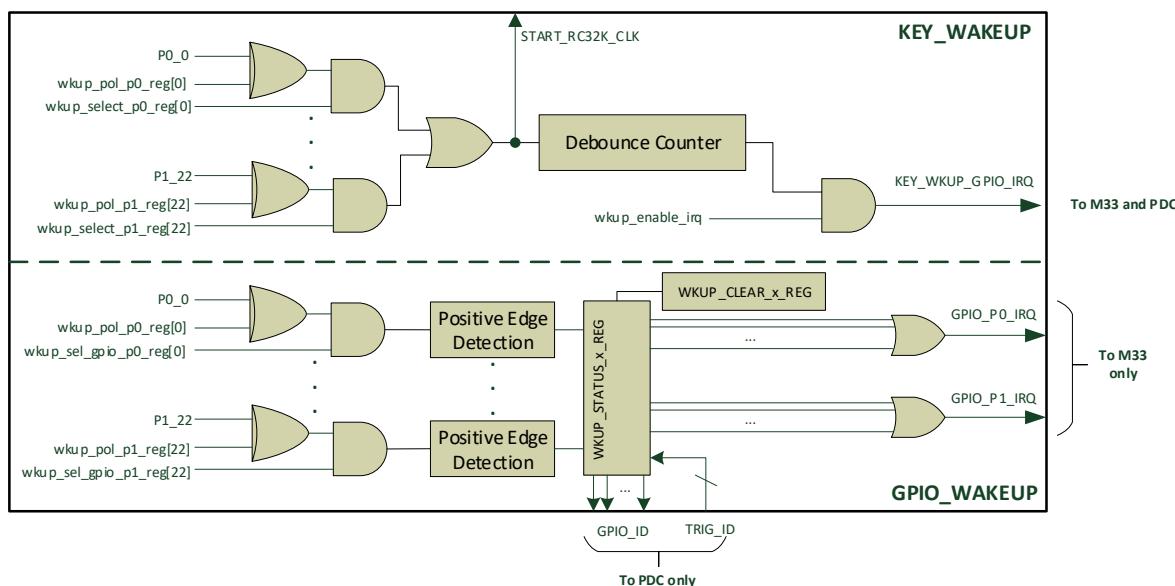


Figure 58: Wake-Up Controller Block Diagram

26.2 Architecture

The Wake-up controller can monitor all GPIO lines for an event. A line of XOR gates defines the polarity of the signal to be monitored. Two parallel structures are implemented explicitly separated in Figure 58 by a dashed line:

- **KEY_WAKEUP:** It can be programmed to monitor a number of GPIOs regarding their edge or level. After a GPIO toggles, a debounce counter can be triggered to debounce the key press before generating an interrupt. The KEY_WKUP_GPIO_IRQ will be generated towards the Cortex-M33 and the PDC (named as Debounced_IO in the PDC). The debouncing time can be programmed to be up to 63 ms. If debouncing is selected to be 0, the interrupt will be issued on an edge detection, but the external signal needs to be at least 2 x RC32K clock periods asserted

Multi-core Bluetooth 5.2 SoC family with system PMU

considering the system being active. If the system is in Hibernation, the external signal needs to be at least 4 x RC32K periods asserted before it actually triggers the interrupt.

This circuit can wake up the system from Hibernation (clock-less sleep) as well as Deep or Extended (clocked) Sleep. Especially for Hibernation, this circuit must be programmed to wake the system up, as it is the only one that starts the RC32K oscillator as shown in the figure. Since the interrupt is kept asserted until acknowledged by SW, the Cortex-M33 will be able to receive it after its power domain has been powered up by PDC. Of course, a respective PDC entry needs to be in place for waking up Cortex-M33.

When the system is active, the circuit can keep on being used as a button press indicator with debouncing. Note that, if multiple GPIOs have been toggled there is no indication which one has generated the interrupt.

- **GPIO_WAKEUP:** This circuit can be programmed to monitor edges (positive or negative) and latch them into a status register so source is known by the application. This status register will be delivered to the PDC in form of a signal bus for being used as a trigger for the PDC entries. Moreover, an OR of each GPIO port signal will form a level interrupt towards the Cortex-M33, namely GPIO_P0_IRQ and GPIO_P1_IRQ.

This circuit can wake up the system from any clocked sleep (Deep or Extended) but not from Hibernation. Since both interrupt lines are kept asserted until acknowledged by SW, the M33 will be able to receive them after its power domain has been powered up by PDC. Of course, a respective PDC entry needs to be in place for waking up Cortex-M33.

When the system is active, the circuit can keep on being used as an interrupt generator towards Cortex-M33 storing the GPIO state that has caused the interrupt in the first place.

26.3 Programming

There is a simple sequence of steps that needs to be followed to program the Wake-Up controller:

1. Enable the Wake-Up Controller by setting the CLK_TMR_REG[WAKEUPCT_ENABLE] bit.
2. Set up triggering polarity (WKUP_POL_Px_REG).
3. If debouncing is needed, define debounce time in WKUP_CTRL_REG[WKUP_DEB_VALUE].
4. Register PDC and/or KEY Interrupts:
 - a. Add PDC Entries (needed when the device goes to sleep)
 - b. Clear any latched values of the GPIOs (WKUP_CLEAR_Px_REG).
 - c. Add Wake-Up events (WKUP_SELECT_Px_REG)
 - d. Add GPIO interrupts (WKUP_SEL_GPIO_Px_REG)
5. If needed, add the corresponding ISRs and enable the interrupts (KEY_WKUP_GPIO IRQn, GPIO_P0_IRQn, GPIO_P1_IRQn, PDC_IRQn).

27 General Purpose ADC

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

27.1 Introduction

The DA1469x 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 and bipolar (differential) mode. The ADC has its own voltage regulator (LDO) of 1.2 V, which represents the full-scale reference voltage.

Features

- 10-bit dynamic ADC with 125 ns conversion time
- Maximum sampling rate 4 Msample/s
- Ultra-low power (13 μ A typical supply current at 100 ksample/s)
- Single-ended as well as differential input with two input ranges via 3x attenuator
- Eight single-ended in VBGA100 (four in VBGA86) or four differential external input channels (in both VFBGA100 and VFBGA86)
- Oversampling up to 128 steps providing effectively up to 11.2 bits accuracy (ENOB)
- Battery monitoring function
- Chopper function
- Offset and zero scale adjust
- Common-mode input level adjust
- DMA support

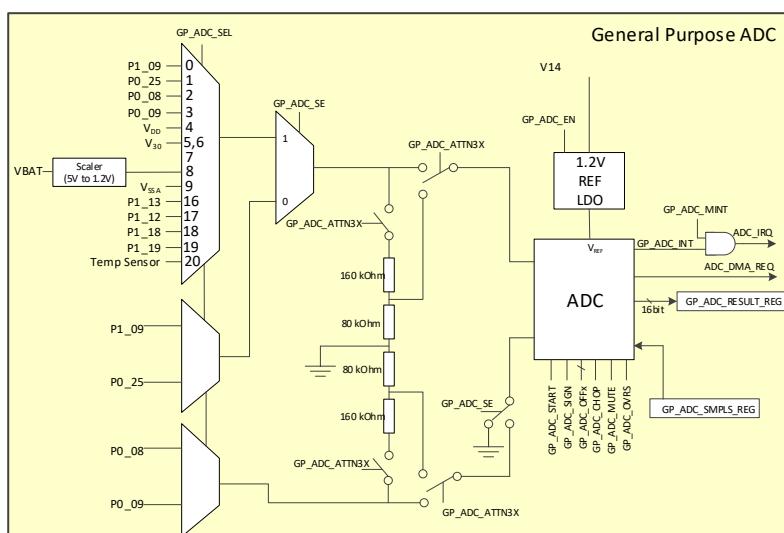


Figure 59: General-Purpose ADC Block Diagram

27.2 Architecture

The ADC architecture shown in Figure 59 has the following sub blocks:

- Analog to Digital converter (ADC)
 - ADC analog part internally clocked with 100 MHz(default) or the ADC_CLK selected with GP_ADC_CTRL_REG[GP_ADC_CLK_SEL]
 - ADC logic part clocked with the ADC_CLK (16 MHz or 96 MHz) selected with CLK_PER_REG[GPADC_CLK_SEL]

Multi-core Bluetooth 5.2 SoC family with system PMU

- 1.2 V Reference LDO for the ADC supply with a high PSRR enabled with GP_ADC_CTRL_REG[GP_ADC_EN]
- 1.2 V Digital LDO for the ADC digital logic supply enabled with GP_ADC_CTRL_REG[LDO_VDDD_HIGH_ENABLE]. Note that, this LDO supply comes from V18P which needs to be activated before the GPADC is used
- 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 eight specific GPIO ports, battery voltage (VBAT1) and the analog ground level (AVS) can be measured

27.2.1 Input Channels and Input Scale

[Table 129](#) summarizes the single-ended or differential operation for the external channels via bit GP_ADC_CTRL_REG[GP_ADC_SE].

Input voltages up to 3.45 V can be handled if bit GP_ADC_CTRL2_REG[GP_ADC_ATTN3X] is set to 1. The VBAT scaler scales from 5 V to 1.2 V, so 1 LSB corresponds with $5 / 1023 = 4.88 \text{ mV}$.

Table 129: GPADC Input Channels and Voltage Range

| GP_ADC_ATTN3X | GP_ADC_SE | GP_ADC_SEL | Input Channels | Input Scale | Input Limits |
|---------------|-----------|-------------------------------|--|------------------|--------------------|
| 0 | 1 | 0, 1, 2, 3, 16, 17, 18, 19 | P0_08, P0_09, P0_25, P1_09, P1_12, P1_13, P1_18, P1_19 | 0 V to +1.2 V | -0.1 V to +1.3 V |
| 0 | 0 | 0 | [P1_09, P0_25] | -1.2 V to +1.2 V | -1.3 V to +1.3 V |
| 0 | 0 | Not 0 | [P0_08, P0_09] | -1.2 V to +1.2 V | -1.3 V to +1.3 V |
| 1 | 1 | 0, 1, 2, 3, 16, 17, 18, 19 | P0_08, P0_09, P0_25, P1_09, P1_12, P1_13, P1_18, P1_19 | 0 V to +3.6 V | -0.1 V to +3.45 V |
| 1 | 0 | 0 | [P1_09, P0_25] | -3.6 V to +3.6 V | -3.45 V to +3.45 V |
| 1 | 0 | Not 0 | [P0_08, P0_09] | -3.6 V to +3.6 V | -3.45 V to +3.45 V |

There are a few more channel inputs to GP ADC apart from the GPIOs listed below:

- VDD: this is the 0.9/1.2V core supply rail of the ADC circuit.
- V30: this is the V30 supply rail
- VSSA: this is the analog ground
- Tempsensor: this is reading the output of the die temperature sensor

27.2.2 Operation

The ADC has the following modes of operation as shown in [Figure 59](#):

- Manual mode (See section [27.2.3.1](#))
- Continuous mode (See section [27.2.3.2](#))

In both modes the ADC performance can be increased by enabling conversion modes:

- Oversampling mode (See section [27.2.4.2](#))
- Chopper mode (See section [27.2.4.3](#))

Multi-core Bluetooth 5.2 SoC family with system PMU

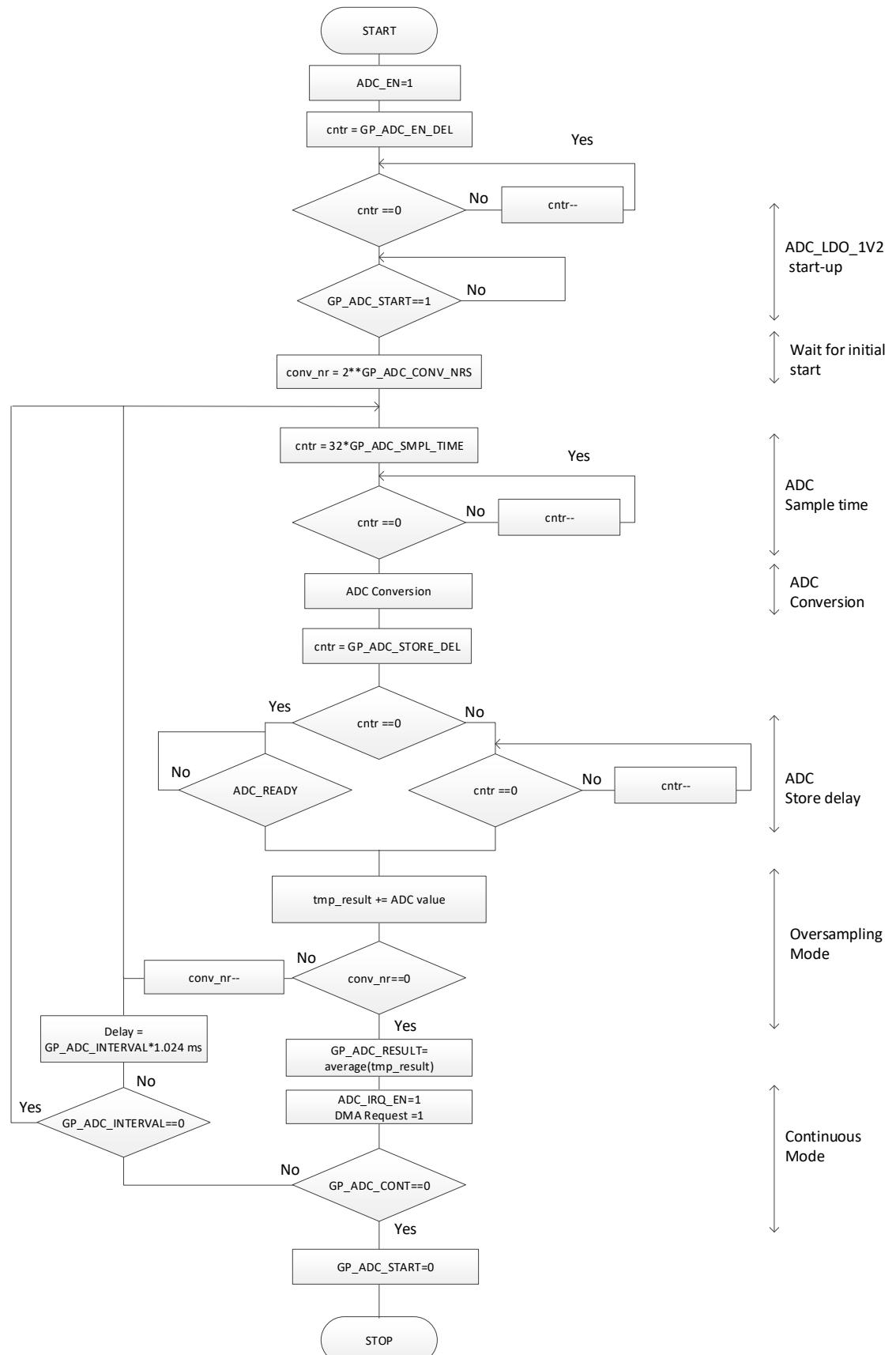


Figure 60: GPADC Operation Flow Diagram

27.2.2.1 Enabling the ADC

The enabling or disabling of the ADC is triggered by configuring bit GP_ADC_CTRL_REG[GP_ADC_EN]. When the latter is set to 1, the LDO is first enabled, then after the delay value set in GP_ADC_CTRL3_REG[GP_ADC_EN_DEL] (typically 20 µs to account for the LDO settling time) the ADC will be enabled, and an AD-conversion can be started. See [Table 130](#) for recommended values.

Table 130: ADC_LDO_1V2 Start-Up Delay

| f _{ADC_CLK} | GP_ADC_EN_DEL | Delay (µs) |
|----------------------|---------------|------------|
| 32 MHz | 0x16 | 22 |
| 96 MHz | 0x40 | 21.3 |

Formula:

$$GP_ADC_EN_DEL = 20E-6 * f_{ADC_CLK} / 32$$

This value must be rounded up.

The GPADC is a dynamic ADC and consumes no static power, except for the ADC_LDO_1V2 that consumes approximately 5 µA. Therefore, GP_ADC_EN must be set to 0 if the ADC is not used.

27.2.3 Operating Modes

27.2.3.1 Manual Mode

If GP_ADC_CTRL_REG[GP_ADC_START] is set to 1 the conversion is started. After a conversion, bit GP_ADC_START is set to 0, bit GP_ADC_INT is set to 1 (interrupt) and GP_ADC_RESULT_REG contains the valid ADC value. Software should always check that bit GP_ADC_START = 0 before starting a new conversion.

27.2.3.2 Continuous Mode

Setting GP_ADC_CTRL_REG[GP_ADC_CONT] to 1 enables the Continuous mode, which automatically starts a new ADC conversion after the current conversion is done. The GP_ADC_START bit is only needed once to trigger the first conversion. While Continuous mode is active, GP_ADC_RESULT_REG always contains the latest ADC value.

To correctly terminate Continuous mode, the GP_ADC_CONT bit must first be disabled. It must 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, Continuous mode must be disabled by setting bit GP_ADC_CONT to 0, while waiting until bit GP_ADC_START = 0.

At full speed, the ADC consumes approximately 75 µA. If the data rate is less than 100 ksample/s, the current consumption would be about 13 µA.

Programmable Delay

The time interval between two successive ADC conversions is programmable with GP_ADC_CTRL3_REG [GP_ADC_INTERVAL] in units of 1.024 ms. If GP_ADC_INTERVAL = 0, the conversion will restart immediately. If GP_ADC_INTERVAL is non-zero, it can take up to 1 ms before the first conversion is executed, because the ADC first synchronizes to the delay clock.

27.2.4 Conversion Modes

27.2.4.1 ADC Conversion

Each ADC conversion has three phases:

- Sampling
- Conversion
- Storage

Multi-core Bluetooth 5.2 SoC family with system PMU

The ADC 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. When the conversion is ready, the ADC value is stored after a delay set in GP_ADC_CTRL2_REG[GP_ADC_STORE_DEL].

Sampling Phase

The sampling time constant τ_{ADC} depends on the output impedance of the source, the internal resistive dividers, and the internal sampling capacitor.

Table 131: ADC Sampling Time Constant

| ADC Input | τ_{ADC} |
|---------------------------------------|--|
| VBAT | 500 ns |
| GPADC0, GPADC1 (GP_ADC_ATTN3X = 0) | $R_{OUT} * 0.5 \text{ pF}$ |
| GPADC0, GPADC1 (GP_ADC_ATTN3X = 1) | $(R_{OUT} + 160 \text{ k}\Omega) * 0.5 \text{ pF}$ |

The settling time is approximately $7 * \tau_{ADC}$ for 10 bits accuracy and $9 * \tau_{ADC}$ for 12 bits accuracy. The settling time is clocked with $f_{ADC_CLK} / 32$ and is counted with GP_ADC_CTRL_REG[GP_ADC_SMPL_TIME].

Formula:

$$GP_ADC_SMPL_TIME = 7 * \tau_{ADC} * (f_{ADC_CLK} / 32)$$

This value must be rounded up.

For a settling time less than one ADC_CLK cycle, GP_ADC_SMPL_TIME can be set to 0.

Example: To measure VBAT with $\tau_{ADC} = 500 \text{ ns}$ and $f_{ADC_CLK} = 32 \text{ MHz}$:

$$GP_ADC_SMPL_TIME = 7 * 500E-9 * 32E6 / 32 = 3.5 \Rightarrow 4.$$

Conversion and Storage Phase

The ADC conversion time is around 125 ns, done with a fast, internal oscillator (100 MHz typ.). The conversion can be handshaked or a storage delay of a fixed number of cycles can be programmed.

Handshake Mode (GP_ADC_STORE_DEL = 0)

In Handshake mode, the conversion result is available in GP_ADC_RESULT_REG after the conversion time (in ADC_CLK cycles) plus the synchronization time. Conversion and storage time (in ADC_CLK cycles):

$f_{ADC_CLK} = 32 \text{ MHz}: 4 + 3$

$f_{ADC_CLK} = 96 \text{ MHz}: 12 + 3$

Fixed Mode (GP_ADC_STORE_DEL > 0)

In Fixed mode, the conversion result is available in GP_ADC_RESULT_REG after the programmed storage delay, whether or not the conversion is ready.

Conversion and storage time (in ADC_CLK cycles):

$GP_ADC_STORE_DEL + 1$

The total ADC conversion time is shown in [Table 132](#).

Table 132: Total ADC Conversion Time

| GP_ADC_STORE_DEL | ADC Conversion Time (ADC_CLK cycles) |
|--------------------------------|--|
| 0 (recommended) | $(4 \text{ or } 12) + 32 * GP_ADC_SMPL_TIME + 3$ |
| > 0 (Note 1) | $32 * GP_ADC_SMPL_TIME + (GP_ADC_STORE_DEL + 1)$ |

Multi-core Bluetooth 5.2 SoC family with system PMU

Note 1 For $f_{GP_ADC_SMPL_TIME} = 0$ and $ADC_CLK = 96$ MHz the $GP_ADC_STORE_DEL$ value must at least be 11.

See section [27.2.10](#) to determine the value of $GP_ADC_SMPL_TIME$ based on the input impedance.

27.2.4.2 Oversampling Mode

In Oversampling mode, multiple successive conversions are executed, and the results are averaged to increase the Effective Number Of Bits (ENOB). The number of conversions is programmable with $GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS]$ and is given by $2^{GP_ADC_CONV_NRS}$.

The resulting $GP_ADC_RESULT_REG$ is 16 bits left-aligned and the effective number of significant bits increases with the number of conversions, as shown in [Table 133](#).

Table 133: Oversampling Mode Effective Number of Bits

| GP_ADC_CONV_NRS | Effective Number Of Bits (Left-Aligned) in GP_ADC_RESULT_REG |
|------------------------|---|
| 0 | 9.05 |
| 1 | 9.45 |
| 2 | 9.83 |
| 3 | 10.21 |
| 4 | 10.52 |
| 5 | 10.85 |
| 6 | 11.10 |
| 7 | 11.27 |

The preferred settings for acquiring the results of [Table 133](#) are given in [Table 134](#).

Table 134: Preferred Settings for ENOB Measurements

| Description | Description Register Setting |
|---|--|
| Run digital at PLL speed | $PLL_SYS_CTRL1_REG[PLL_EN] = 1$ |
| Use internal 100 MHz clock as SAR clock | $GP_ADC_CTRL_REG[GP_ADC_CLK_SEL] = 0$ |
| Use auto zero and reference sampling in the LDO to suppress noise | $GP_ADC_CTRL_REG[GP_ADC_LDO_ZERO] = 1$ |
| Disable chopping (default) | $GP_ADC_CTRL_REG[GP_ADC_CHOP] = 0$ |
| Sign is not inverted (default) | $GP_ADC_CTRL_REG[GP_ADC_SIGN] = 0$ |
| Measure single ended | $GP_ADC_CTRL_REG[GP_ADC_SE] = 1$ |
| Single-Ended: (e.g. P1_09) | $GP_ADC_CTRL_REG[GP_ADC_SEL] = 0$ |
| Enable dynamic LDO current load | $GP_ADC_CTRL2_REG[GP_ADC_IDYN] = 1$ |
| Enable static LDO 20A current load | $GP_ADC_CTRL2_REG[GP_ADC_I20U] = 1$ |
| 32 ADC clock cycles sampling time | $GP_ADC_CTRL2_REG[GP_ADC_SMPL_TIME] = 1$ |
| Negative Offset centered (zero offset) | $GP_ADC_OFFN_REG = 0x200$ |
| Positive Offset centered (zero offset) | $GP_ADC_OFFP_REG = 0x200$ |

27.2.4.3 Chopper Mode

By enabling Chopper mode, the internal ADC offset is cancelled by swapping the inputs and outputs between conversions. This method also smooths other non-ideal effects and is recommended for DC and slowly changing signals.

Multi-core Bluetooth 5.2 SoC family with system PMU

Chopper mode can be enabled in Manual, Continuous and Oversampling mode. When Oversampling mode is not active, two samples are taken with opposite signs. In the other cases, an even number of conversions is done, based on the value of GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS].

Chopper mode is enabled by setting bit GP_ADC_CTRL_REG[GP_ADC_CHOP] to 1. The mid-scale value of the ADC is the 'natural' zero point of the ADC (ADC result = 511.5 = 1FF or 200 Hex = 01.1111.1111 or 10.0000.0000 Bin). Ideally, this corresponds to $V_i = 1.2 \text{ V} / 2 = 0.6 \text{ V}$ in single-ended mode and $V_i = 0.0 \text{ V}$ in differential mode.

27.2.5 Additional Settings

If bit GP_ADC_CTRL2_REG[GP_ADC_ATTN3X] is set to 1, the input range is scaled by a factor of three, and the zero-point changes accordingly (1.8 V single-ended and to 0.0 V differential).

With bit GP_ADC_CTRL_REG[GP_ADC_MUTE] = 1, the ADC input is switched to the mid-scale input level, so the ADC result ideally should be 511.5. If instead a value of 515 is observed, the output offset is +3.5.

With bit GP_ADC_CTRL_REG[GP_ADC_SIGN] = 1 the sign of both the ADC input and output is inverted. Two sign changes have no effect on the signal path, but the sign of the ADC offset will change.

At the same output offset of +3.5 the ADC result with opposite GP_ADC_SIGN will be 508. The sum of these values equals $515 + 508 = 1023$. This is the mid-scale value of an 11-bit ADC, so one additional bit, due to the oversampling by a factor of two.

The LSB of this 11-bit word should be ignored when a 10-bit word is preferred. In that case the result is 511.5, so the actual output value will be either 511 or 512.

27.2.6 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 slightly 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 or by offset calibration.

The ADC result will also include some noise. If the input signal itself is noise free (inductive effects included), the average noise level will be $\pm 1 \text{ LSB}$. Taking more samples and calculating the average value will reduce the noise and increase the resolution. This can be done by programming GP_ADC_CTRL2_REG[GP_ADC_CONV_NRS] to a non-zero value.

With a 'perfect' input signal (e.g. 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. Since the DA1469x 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] and GP_ADC_CTRL2_REG[GP_ADC_IDYN] to 1. Bit GP_ADC_I20U enables a constant $20 \mu\text{A}$ load current at the regulator output, so that the current will not drop to zero. Bit GP_ADC_IDYN enables a $10 \mu\text{A}$ load current during sampling phase, so that the load current during sampling and conversion phase becomes approximately the same.

27.2.7 Offset Calibration

A relative high offset E_{OFS} (up to 20 mV, so approximately 20 LSB) is caused by a very small dynamic comparator. This offset 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 can be compensated in the ADC network itself. To calibrate the ADC follow the steps in [Table 135](#).

Table 135: 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 |

Note 2 The average of GP_ADC_OFFP and GP_ADC_OFFN should be 0x200 (with a margin of E_{OFFS})

To increase 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.

It is recommended to implement the above calibration routine during the initialization phase of DA1469x. 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.

27.2.8 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.

27.2.9 Common Mode Adjustment

The common mode level of the differential signal must be 0.6 V (or 1.8 V with GP_ADC_ATTN3X = 1). If the common mode input level of 0.6 V cannot be achieved, the common mode level of the GPADC can be adjusted via GP_ADC_OFFP_REG and GP_ADC_OFFN_REG according to [Table 136](#). The GPADC can tolerate a common mode margin of up to 50 mV.

Table 136: Common Mode Adjustment

| CM Voltage (V _{cm}) | GP_ADC_OFFP = GP_ADC_OFFN |
|-------------------------------|---------------------------|
| 0.3 V | 0x300 |
| 0.6 V | 0x200 |
| 0.9 V | 0x100 |

Any other common mode levels between 0.0 V and 1.2 V can be calculated from the table above. 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.

Note: The input voltage limits for the ADC in differential mode are: -1.3 V to +1.3 V (for GP_ADC_ATTN3X = 0, see [Table 129](#)). The differential input range of the ADC is: -1.2 V < V[P0_09, P0_25] < +1.2 V. Therefore, if V_{cm} < 0.5 V or V_{cm} > 0.7 V, the input can no longer cover the whole ADC range.

27.2.10 Input Impedance, Inductance and Input Settling

The GPADC has no input buffer stage. During the sampling phase, a capacitor of 0.5 pF is switched to the input line. The precharge of this capacitor is at midscale level so the input impedance is infinite.

Multi-core Bluetooth 5.2 SoC family with system PMU

During the sampling phase, a certain settling time is required. A 10-bit accuracy requires at least seven time-constants τ_{ADC} , determined by the output impedance of the input signal source, the internal resistive dividers and the 0.5 pF sampling capacitor. See [Table 131](#).

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).

To measure the noise level of the ADC and the LDO, bit GP_ADC_CTRL_REG[GP_ADC_MUTE] must be set to 1. The internal noise level should be less than ± 2 LSB on average. When a higher noise level is observed on the input channel(s), applying external filter capacitor(s) will reduce the noise.

27.2.11 Reading Temperature Sensors

The General-Purpose ADC can read all temperature sensors integrated on the chip. The following temperature sensors are available:

1. The one used by the charger circuitry that provides a pretty good accuracy of the die temperature. To enable reading from it, in the GP_ADC_CTRL_REG, GP_ADC_DIFF_TEMP_EN must be 0, GP_ADC_SEL must be 20, and GP_ADC_DIFF_TEMP_SEL must be 2. The formula that converts the ADC reading into temperature is as follows:
 - a. For uncalibrated measurements: $T = (GPADC_value - 734) / 2.3$ (10-bits resolution)
 - b. For measurements, using a single-point calibration: Store the GPADC-readout ($GPADC_{TCAL}$) at a well-defined temperature " T_{CAL} " (e.g. at room temperature)
 $T = T_{CAL} + (GPADC_value - GPADC_{TCAL}) / 2.3$
2. Three more simple diode-based sensors that are spread in the die and can be used for relative measurements only. They are placed next to the radio, the charger and the bandgap respectively. To enable reading from each one, in the GP_ADC_CTRL_REG, GP_ADC_DIFF_TEMP_EN must be 1, GP_ADC_SEL must be 20 and GP_ADC_DIFF_TEMP_SEL defines which sensor will be read. The formula for converting the sensor reads into temperature difference, is as follows:
 $dT(t1 - t0) = (GPADC_value(t1) - GPADC_value(t0)) / 82$,
 where $t0$ and $t1$ designate the successive reading time points.

Please note that, while measuring and/or calibrating, the system's power dissipation should be kept the same, otherwise the measurement is affected by the internal thermal gradient.

27.3 Programming

There is a simple sequence of steps that should be followed to program and use the General-Purpose AD Converter:

1. Enable the reference LDO by setting the GP_ADC_CTRL_REG[GP_ADC_EN] bit. This bit will also enable the digital part of the ADC.
2. Setup the GPIO input (Px_yy_MODE_REG[PID] = 26).
3. Select the input channel GP_ADC_CTRL_REG[GP_ADC_SEL].
4. Select the sampling mode (differential, single ended) by writing the GP_ADC_CTRL_REG[GP_ADC_SE] bit.
5. Select between manual and continuous mode of sampling (GP_ADC_CTRL_REG[GP_ADC_CONT]).
6. Setup extra options (see GP_ADC_CTRLx_REG description)
7. Start the conversion by setting GP_ADC_CTRL_REG[GP_ADC_START] bit.

Multi-core Bluetooth 5.2 SoC family with system PMU

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.

28 ΣΔ ADC

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

28.1 Introduction

The DA1469x is equipped with a 2nd order 14 bits ΣΔ ADC. The conversion time for a 14-bit resolution is about 1 ms, allowing for a sampling rate equal to 1000 samples per second.

The reference voltage can be selected between the internal 1.2 V reference or an external voltage reference. It also supports VBAT measuring as well as 8 external channels for sensing.

Features

- 14-bit resolution
- Maximum sampling rate of 1000 sample/s
- Single-ended as well as differential input with two input scales
- Eight single-ended in VBGA100 (four in VBGA86) or four differential external input channels in VFBGA100 (two in VFBGA86)
- Battery monitoring function
- DMA support with the GP DMA

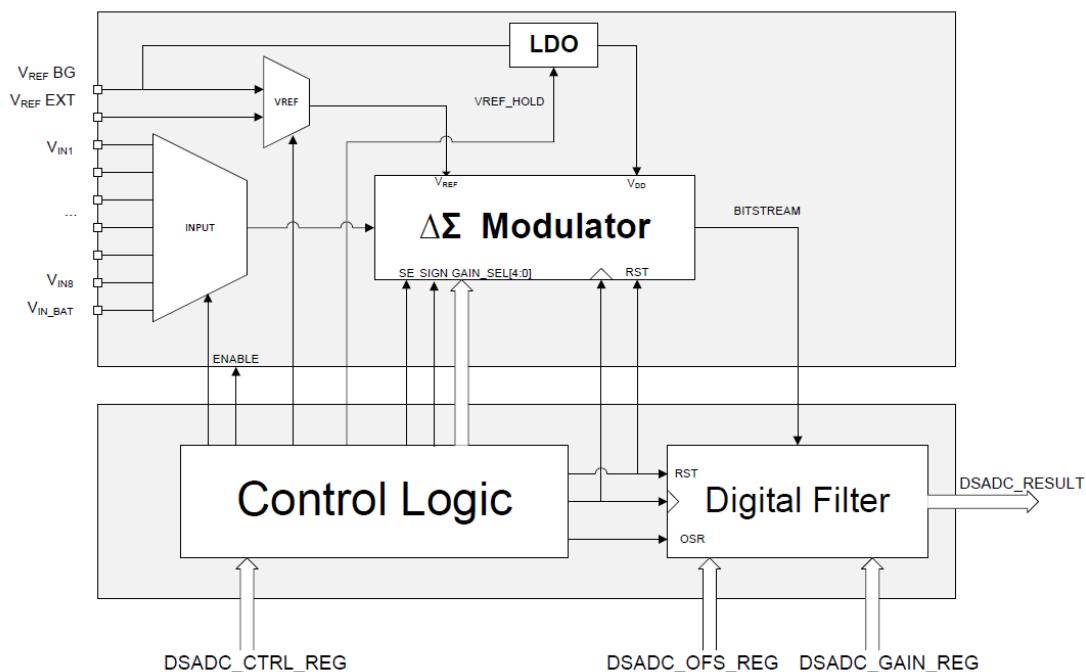


Figure 61: Block diagram of the $\Delta\Sigma$ ADC

28.2 Architecture

The block comprises an analogue part including the $\Delta\Sigma$ -Modulator and auxiliary circuitry, and a digital part containing the digital filter and control logic.

An 8-input multiplexer and a separate VBAT input is placed in front of the modulator. A reference selector is also added that allows an external reference voltage to be used. The internal reference voltage of 1.2 V is selected by default.

Multi-core Bluetooth 5.2 SoC family with system PMU

The analogue block is a modulator allowing 14 bits accuracy to be achieved (thermal noise floor). For single shot measurements, a higher oversampling rate (OSR) is required than for normal Sigma-Delta operation due to step-response behavior.

The digital block includes the digital filter and control logic for analogue. It takes care of the chopper and dynamic element matching algorithm. It also provides the clock and reset to the ADC and control signals to the input mux and reference-selection mux.

An internal divider of 160 kOhm and 3*160 kOhm is utilized for reading the battery voltage.

An external reference voltage can be applied at pins SDADC_REF (P0_16) and SDADC_GND (P0_06) of no more than 1.2 V to improve precision.

28.3 Programming

There is a simple sequence of steps that needs to be followed to program and use the $\Sigma\Delta$ AD Converter:

1. Enable the SDADC block by setting the SDADC_CTRL_REG[SDADC_EN] bit.
2. Setup the GPIO input (Px_yy_MODE_REG[PID] = 26).
3. Select the input channel for the positive side (SDADC_CTRL_REG[SDADC_INP_SEL]) and the negative side (SDADC_CTRL_REG[SDADC_INN_SEL], negative side is ignored in single ended mode).
4. Select the Voltage reference (SDADC_CTRL_REG[SDADC_VREF_SEL]).
5. Select the sampling mode (differential, single ended) by writing the SDADC_CTRL_REG[GP_ADC_SE] bit.
6. Select between manual and continuous mode of sampling (SDADC_CTRL_REG[SDADC_CONT]).
7. Setup extra options (see SDADC_CTRL_REG description)
8. Start the conversion by setting SDADC_CTRL_REG[SDADC_START] bit.
9. Wait for SDADC_CTRL_REG[SDADC_START] to become 0 or interrupt being triggered (when used).
10. Clear the ADC interrupt by writing any value to SDADC_CLEAR_INT_REG.
11. Get the ADC result from the SDADC_RESULT_REG.

29 Audio Unit (AU)

| Device | DA14691 | DA14695 | DA14697 | DA14699 |
|----------------------|---------|---------|---------|---------|
| Feature Availability | ✓ | ✓ | ✓ | ✓ |

29.1 Introduction

The Audio Unit is made up of two digital interfaces, namely: a PDM and PCM. Its also has a Sampling Rate Converter (SRC), which is used for adjusting the sampling rate of audio samples between the two interfaces and memory.

The PDM interface provides a serial connection for one stereo or two mono input devices (e.g. MEMS microphones) or output devices. The interface has a single clock PDM_CLK and one input/output PDM_DI/PDM_DO that can carry two channels in a time divided manner.

The PCM controller implements up to 192 kHz synchronous interface to external audio devices, ISDN circuits, and serial data interfaces. It enables master and slave modes, and also supports I2S and TDM formats.

The AU has dedicated DMA channels for the PCM and PDM streams. The PCM data flow is further supported by an internal dedicated 8x32-bit FIFO that cannot be used in stereo mode.

Features

- Supported conversions:
 - SRC_IN (32 bits) to SRC_OUT (32 bits)
 - PDM_IN (1bit) to SRC_OUT (32 bits)
 - SRC_IN (32 bits) to PDM_OUT (1 bit)
- SRC_IN, SRC_OUT Sample rates 8 kHz to 192 kHz
- SNR > 100 dB
- Single Buffer I/O with DMA support
- Automatic mode to adjust sample rate to the applied frame sync (e.g. PCM_FSC)
- Manual mode to generate interrupts at the programmed sample rate. Adjustment is done by SW based on buffer pointers
- PCM (Master/Slave) interface
 - PCM_FSC
 - Master/slave 4 kHz to 96 kHz
 - Strobe Length 1, 8, 16, 24, 32, 40, 48 and 64 bits
 - PCM_FSC before or on the first bit. (In Master mode)
 - 2x32 channels
 - Programmable slot delay up-to 31*8 bits
 - Formats
 - PCM mode
 - I2S mode (Left/Right channel selection) with N*8 for Left and N*8 for Right
 - IOM2 mode (double clock per bit)
 - Programmable clock and frame sync inversion
- PDM interface
 - PDM_CLK frequency 62.5 kHz - 4 MHz
 - Down-sampling to 32 bits in SRC
 - PDM_CLK on/off to support Sleep mode
 - PDM_DATA
 - (input): 1 Channel in stereo format

Multi-core Bluetooth 5.2 SoC family with system PMU

- (output): 2 Channels in mono format, 1 Channel in stereo format
- Programmable Left/Right channel selection

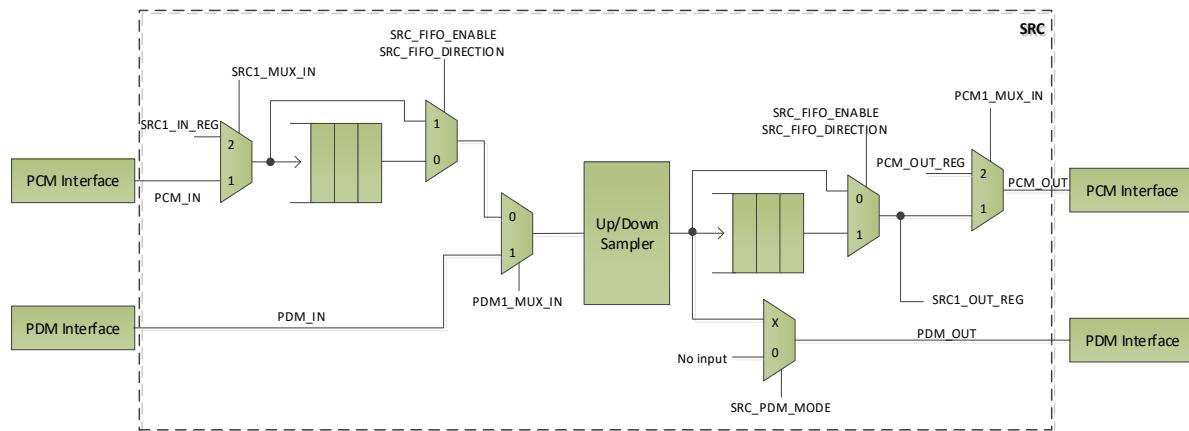


Figure 62: Audio Unit Block Diagram

29.2 Architecture

29.2.1 Data Paths

The SRC block converts two 32-bit channels into, either a stereo pair, or two mono streams. PCM linear data pairs are transferred to SRC1_IN1/2; the output is 2x32-bit left-aligned on SRC1_OUT1/2. The two 1-bit PDM data inputs are received on PDM_IN and are converted to 2x32 bits, left-aligned to SRC_OUT.

The SRCx_IN input multiplexer (Figure 62) is controlled by APU_MUX_REG[PDM1_MUX_IN]. The input of these multiplexers comes from the audio interfaces or the SRC1_IN1/2_REG. The data to these registers is left-aligned, bits 31-8 are mapped on bits 23-0 of the SRC.

The 32 bits SRC outputs can be read in SRC1_OUT1_REG and SRC1_OUT2_REG and can also be routed to the PCM interface. The input selection of these multiplexers is also controlled by APU_MUX_REG[PCM1_MUX_IN].

An 8x32-bit FIFO can be connected to the SRC input or output data path when not in stereo mode. When the FIFO services the SRC input data path, an SRC input event triggers a FIFO write, while a CPU read access triggers a FIFO read. SRC_IN_IRQ is issued when the FIFO level increments to four samples. When the FIFO services the SRC output data path, an SRC output event triggers a FIFO read, while a CPU write access triggers a FIFO write. SRC_OUT_IRQ is issued when the FIFO level drops below five samples. FIFO operation can be completely disabled.

The SRC can be configured to operate in two different modes of operation:

- Manual mode
- Automatic mode

In **manual mode**, the input/output sampling rate is determined by SRC1_IN_FS_REG / SRC1_OUT_FS_REG registers.

In **automatic mode**, the input/output sampling rate is automatically derived from the external synchronization signals and can only be read back at SRC1_IN_FS_REG / SRC1_OUT_FS_REG registers.

When PDM is used (input/output), SRC operates in automatic mode. The sampling rate reported in SRC1_IN_FS_REG register is PDM_CLK/64, 64 being the default oversampling ratio.

29.2.2 Up/Down Sampler

The Up/Down Sampler performs the required arbitrary resampling by polynomial interpolation at an 8x oversampled input rate and 16x oversampled output rate.

For maximum flexibility, a generic single cycle multiplier facilitates variable coefficient multiplications. The multiplier is combined with optional pre and post adders into an arithmetic unit.

29.2.3 PCM Interface

29.2.3.1 Channel Access and Delay

The PCM interface has two 32-bit registers for TX and RX, namely PCM1_IN1/OUT1_REG and PCM1_IN1/OUT2_REG for input and output directions respectively. These registers can be arranged as eight channels of 8 bits each, named channel 1 to channel 8. By a configurable clock inversion, channel delay and strobe length adjustment, various formats like PCM, I2S, TDM and IOM2 can be supported.

The 8 PCM channels can be delayed with a maximum delay of 31x8bits by configuring PCM1_CTRL_REG[PCM_CH_DEL]. Note that a high delay count in combination with a slow clock, can lead to the PCM_FSC sync occurring before all channels are shifted in or out. The received bits of the current channel may not be properly aligned in that case.

29.2.3.2 Clock Generation

The PCM clock (PCM_CLK) must be generated according to the required sample rate. There are two ways of generating the clock:

- The Fractional Option.** Dividing the system clock by an integer and a fractional part (i.e. inserting jitter in the clock pulse train). This is programmed in the PCM_DIV_REG and PCM_FDIV_REG respectively.
- The Integer Only option.** Approximate the sample rate by adding more clock pulses than bits required. These extra pulses are ignored. This approach is used when external slave devices cannot tolerate the inserted jitter on the clock line. It is configured in PCM_DIV_REG.

The PCM_DIV_REG[PCM_DIV] is a 12 bits field which holds the integer part of the desired clock divider. The fractional part of the divider is stored in the 16 bits PCM_FDIV_REG register. The value of the register is calculated in the following way:

- The position of the left most '1' of the value in binary format defines the denominator
- The amount of '1's define the numerator of the fraction as explained in the example of [Table 137](#).

Table 137: PCM_FDIV_REG Programming Example

| PCM_FDIV_REG (Hex) | PCM_FDIV_REG (Binary) | Numerator | Denominator | Fraction |
|--------------------|-----------------------|-----------|-------------|----------|
| 0x0110 | 0b100010000 | 2 | 9 | 2/9 |
| 0x0101 | 0b100000001 | 2 | 9 | 2/9 |
| 0x1ABC | 0b1101010111100 | 8 | 13 | 8/13 |
| 0xBEEF | 0b101111011101111 | 13 | 16 | 13/16 |
| 0xFEEE | 0b111111011101110 | 13 | 16 | 13/16 |

The FSC pulse is generated from the PCM_CLK by further dividing it by PCM_FSC_DIV.

Both clock generation options are explained in the following table, with 8 bits, 16 bits, 32 bits and 48 bits in various sample rates.

Table 138: Fractional and Integer Only Clock Divisors for Various PCM Frequencies and Sample Rates

| Sample Rate (kHz) | Bits | Desired Bit Clock (kHz) | XTAL (KHz) | | | | | PLL (KHz) | | | | |
|-------------------|------|-------------------------|-----------------|-------------------|--------------|---------------------|-------------------------|-----------------|-------------------|--------------|---------------------|-------------------------|
| | | | 32000 | | | | | 96000 | | | | |
| | | | | Fractional Option | | Integer Only Option | | | Fractional Option | | Integer Only Option | |
| Sample Rate (kHz) | Bits | Desired Bit Clock (kHz) | Desired Divider | PCM_DIV_REG | PCM_FDIV_REG | PCM_DIV_REG | Actual Word size (Bits) | Desired Divider | PCM_DIV_R EG | PCM_FDIV_REG | PCM_DIV_R EG | Actual Word size (Bits) |
| 8 | 1*8 | 64 | 500 | 500 | | 500 | 8 | 1500 | 1500 | | 1500 | 8 |
| 8 | 1*16 | 128 | 250 | 250 | | 250 | 16 | 750 | 750 | | 750 | 16 |
| 8 | 1*24 | 192 | 166.667 | 166 | 2/3 | 160 | 25 | 500 | 500 | | 500 | 24 |
| 8 | 1*32 | 256 | 125 | 125 | | 125 | 32 | 375 | 375 | | 375 | 32 |
| 8 | 2*8 | 128 | 250 | 250 | | 250 | 8 | 750 | 750 | | 750 | 8 |
| 8 | 2*16 | 256 | 125 | 125 | | 125 | 16 | 375 | 375 | | 375 | 16 |
| 8 | 2*24 | 384 | 83.333 | 83 | 1/3 | 80 | 25 | 250 | 250 | | 250 | 24 |
| 8 | 2*32 | 512 | 62.5 | 62 | 1/2 | 50 | 40 | 187,5 | 187 | 1/2 | 150 | 40 |
| 16 | 1*8 | 128 | 250 | 250 | | 250 | 8 | 750 | 750 | | 750 | 8 |
| 16 | 1*16 | 256 | 125 | 125 | | 125 | 16 | 375 | 375 | | 375 | 16 |
| 16 | 1*24 | 384 | 83.333 | 83 | 1/3 | 80 | 25 | 250 | 250 | | 250 | 24 |
| 16 | 1*32 | 512 | 62.5 | 62 | 1/2 | 50 | 40 | 187,5 | 187 | 1/2 | 150 | 40 |
| 16 | 2*8 | 256 | 125 | 125 | | 125 | 8 | 375 | 375 | | 375 | 8 |
| 16 | 2*16 | 512 | 62.5 | 62 | 1/2 | 50 | 20 | 187,5 | 187 | 1/2 | 150 | 20 |
| 16 | 2*24 | 768 | 41.667 | 41 | 2/3 | 40 | 25 | 125 | 125 | | 125 | 24 |
| 16 | 2*32 | 1024 | 31.25 | 31 | 1/4 | 25 | 40 | 93,75 | 93 | 3/4 | 75 | 40 |
| 32 | 1*8 | 256 | 125 | 125 | | 125 | 8 | 375 | 375 | | 375 | 8 |
| 32 | 1*16 | 512 | 62.5 | 62 | 1/2 | 50 | 20 | 187,5 | 187 | 1/2 | 150 | 20 |
| 32 | 1*24 | 768 | 41.667 | 41 | 2/3 | 40 | 25 | 125 | 125 | | 125 | 24 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| | | | XTAL (KHz) | | | | | PLL (KHz) | | | | |
|----|------|------|------------|----|-----|-----|-----|-----------|-----|-----|-----|----|
| 32 | 1*32 | 1024 | 31.25 | 31 | 1/4 | 20 | 50 | 93.75 | 93 | 3/4 | 75 | 40 |
| 32 | 2*8 | 512 | 62.5 | 62 | 1/2 | 50 | 10 | 187.5 | 187 | 1/2 | 150 | 10 |
| 32 | 2*16 | 1024 | 31.25 | 31 | 1/4 | 20 | 25 | 93.75 | 93 | 3/4 | 75 | 20 |
| 32 | 2*24 | 1536 | 20.833 | 20 | 5/6 | 20 | 25 | 62.5 | 62 | 1/2 | 50 | 30 |
| 32 | 2*32 | 2048 | 15.625 | 15 | 5/8 | 10 | 50 | 46.875 | 46 | 7/8 | 25 | 60 |
| 48 | 1*8 | 384 | 83.333 | 83 | 1/3 | N/A | N/A | 250 | 250 | | 250 | 8 |
| 48 | 1*16 | 768 | 41.667 | 41 | 2/3 | N/A | N/A | 125 | 125 | | 125 | 16 |
| 48 | 1*24 | 1152 | 27.778 | 27 | 7/9 | N/A | N/A | 83.3333 | 83 | 1/3 | 80 | 25 |
| 48 | 1*32 | 1536 | 20.833 | 20 | 5/6 | N/A | N/A | 62.5 | 62 | 1/2 | 50 | 40 |
| 48 | 2*8 | 768 | 41.667 | 41 | 2/3 | N/A | N/A | 125 | 125 | | 125 | 8 |
| 48 | 2*16 | 1536 | 20.833 | 20 | 5/6 | N/A | N/A | 62.5 | 62 | 1/2 | 50 | 20 |
| 48 | 2*24 | 2304 | 13.889 | 13 | 8/9 | N/A | N/A | 41.6666 | 41 | 2/3 | 40 | 25 |
| 48 | 2*32 | 3072 | 10.417 | 10 | 2/5 | N/A | N/A | 31.25 | 31 | 1/4 | 25 | 40 |

The yellow marked fields designate that the actual word size achieved in the Integer Only option, is larger than the required bits. The last clock pulses will be ignored in this case. For example, to get 24 bits at 8 KHz sampling rate, a clock of 192 KHz is required. In the Integer Only option, this is not possible. A higher clock will be generated (200 KHz), which results in a word of 25 bits. In this case, the last bit will be ignored.

29.2.3.3 External Synchronization

With the PCM interface in slave mode, the PCM interface supports direct routing through the sample rate converter (SRC). Any drift in PCM_FSC or other frame sync frequencies like 44.1 kHz, can be directly resampled to for example, 48 kHz internal sample rate.

29.2.3.4 Data Formats

PCM Master Mode

Master mode is selected if PCM1_CTRL_REG[PCM_MASTER] = 1.

In master mode, PCM_FSC is output and falls always over Channel 0. The duration of PCM_FSC is programmable with PCM1_CTRL_REG[PCM_FSCLEN]= 1 or 8,16, 24, 32 clock pulses high. The start position is programmable with PCM1_CTRL_REG[PCM_FSCDEL] and can be placed before or on the first bit of channel 0. The repetition frequency of PCM_FSC is programmable in PCM1_CTRL_REG[PCM_FSC_DIV] from 8 to 192kHz.

If master mode selected, PCM_CLK is output and provides one or two clocks per data bit programmable in PCM1_CTRL_REG[PCM_CLK_BIT]. Note that, for PCM_CLK_BIT=1, the length of the FSC (FSC_LEN) is doubled.

The polarity of the signal can be inverted with bit PCM1_CTRL_REG[PCM_CLKINV].

The PCM_CLK frequency selection is described in Section [29.2.3.2](#).

PCM Slave Mode

In slave mode, (bit MASTER = 0) PCM_FSC is input and determines the starting point of channel 0. The repetition rate of PCM_FSC must be equal to PCM_SYNC and must be high for at least one PCM_CLK cycle. Within one frame, PCM_FSC must be low for at least PCM_CLK cycle. Bit PCM_FSCDEL sets the start position of PCM_FSC before or on the first bit (MSB).

In slave mode, PCM_CLK is input. The minimum received frequency is 256 kHz, the maximum is 12.288 MHz.

In slave mode, the main counter can be stopped and resumed on a PCM1_FSC or PCM2_FSC rising edge.

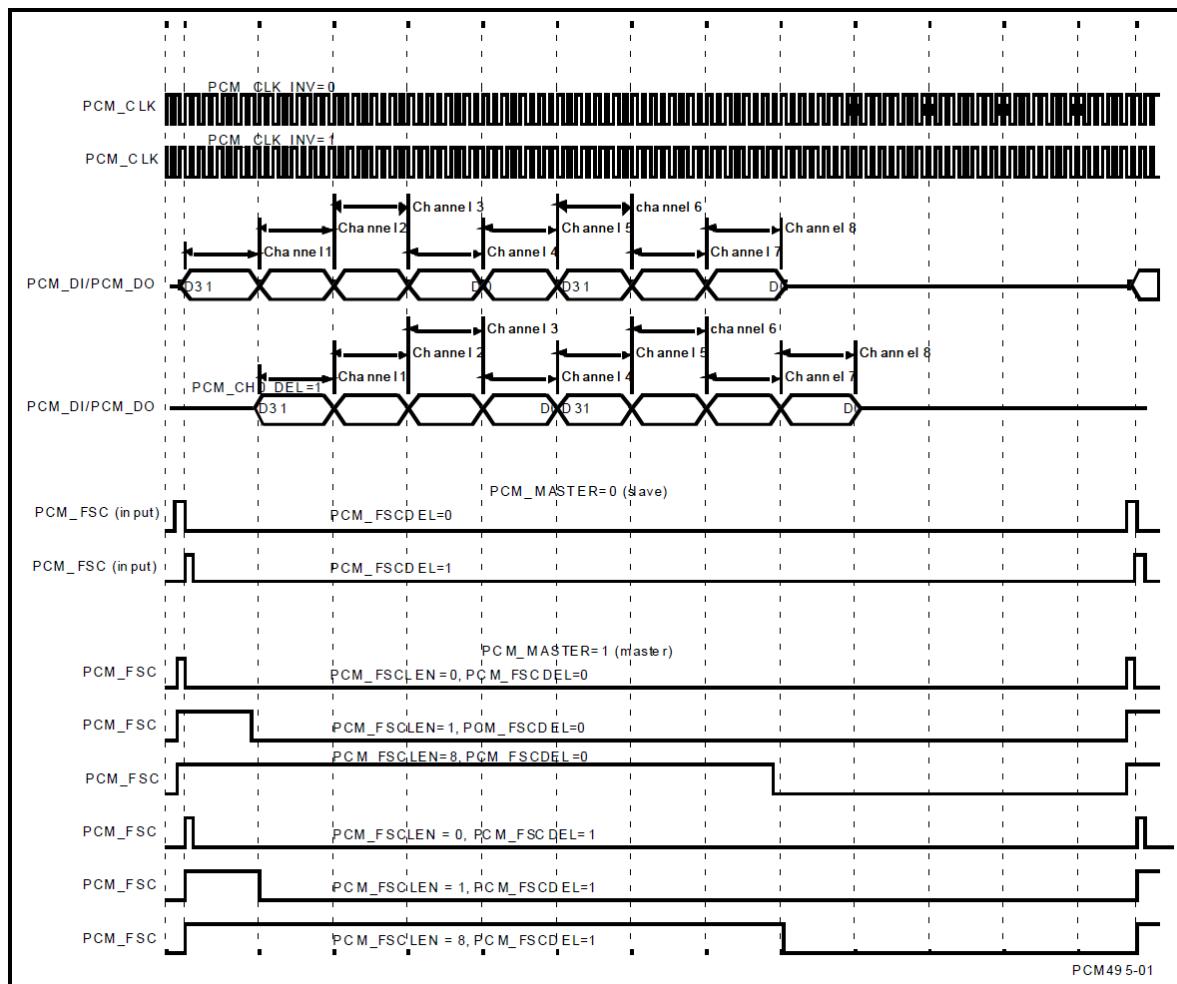


Figure 63: PCM Interface Formats

I2S Formats

The digital audio interface supports I2S mode, Left Justified mode, Right Justified mode and TDM mode.

I2S mode

To support I2S mode, the MSB of the right channel is valid on the second rising edge of the bit clock after the rising edge of the PCM_FSC, and the MSB of the left channel is valid on the second rising edge of the bit clock after the falling edge of the PCM_FSC.

Settings for I2S mode:

- PCM_FSC_EDGE: 1 (all after PCM_FSC)
- PCM_FSCLEN: 4 (4x8 High, 4x8 Low)
- PCM_FSC_DEL: 0 (one bit delayed)
- PCM_CLK_INV: 1 (output on falling edge)
- PCM_CH_DEL: 0 (no channel delay)

Multi-core Bluetooth 5.2 SoC family with system PMU

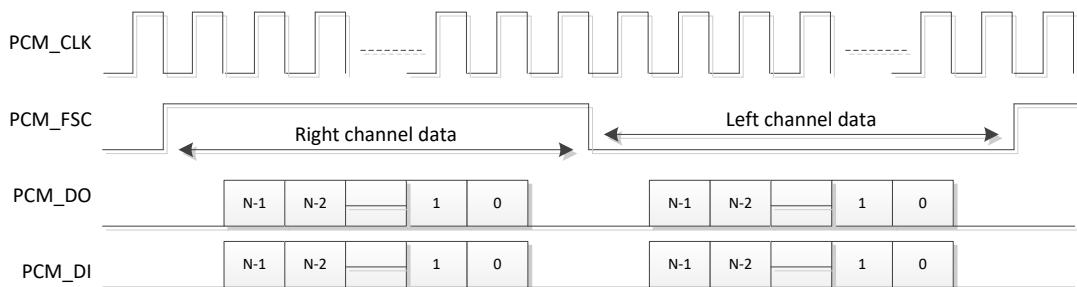


Figure 64: I2S Mode

TDM mode

A time is specified from the normal ‘start of frame’ condition using register bits PCM_CH_DEL. In the left-justified TDM example illustrated in [Figure 65](#), the left channel data is valid PCM_CH_DEL clock cycles, after the rising edge of the PCM_FSC, and the right channel data is valid the same PCM_CH_DEL number of clock cycles after the falling edge of the PCM_FSC.

By delaying the channels, left and right alignment can also be achieved.

Settings for TDM mode:

- PCM_FSC_EDGE: 1 (rising and falling PCM_FSC)
- PCM_FSCLEN: Master 1 to 4
Slave waiting for edge.
- PCM_FSC_DEL: 1 (no bit delay)
- PCM_CLK_INV: 1 (output on falling edge)
- PCM_CH0_DEL: Slave 0-31 (channel delay)
Master 1-3

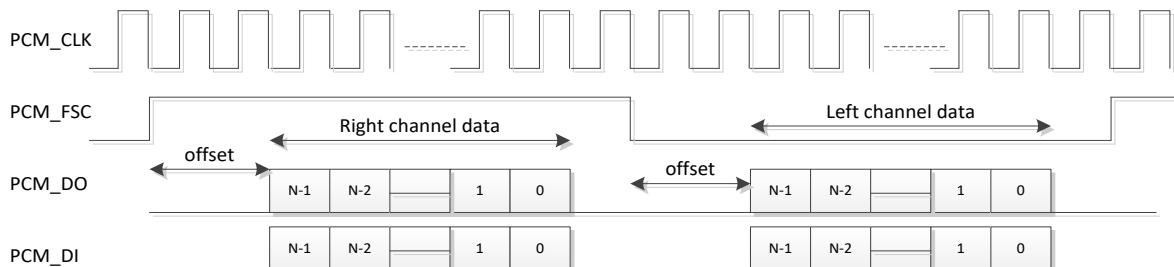


Figure 65: TDM mode (left justified mode)

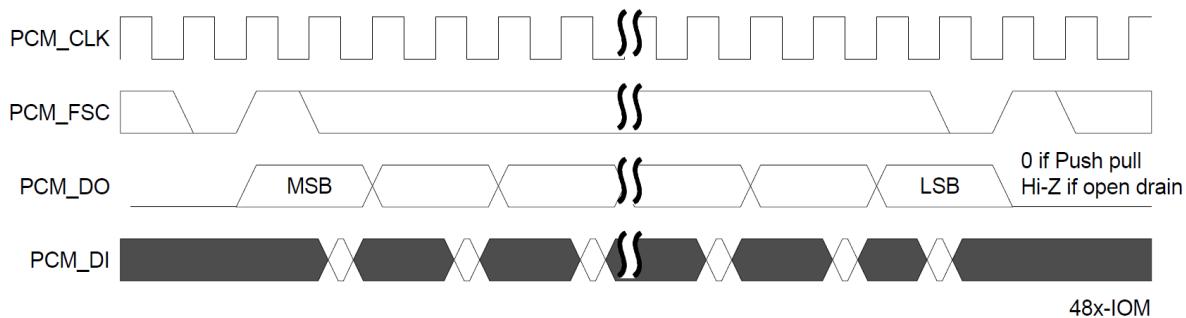
Note that, offset is always in multiples of 8.

IOM Mode

In the IOM format, the PCM_CLK frequency is twice the data bit cell duration. In slave mode, synchronization is on the first rising edge of PCM_FSC while data is clocked in on the second falling edge.

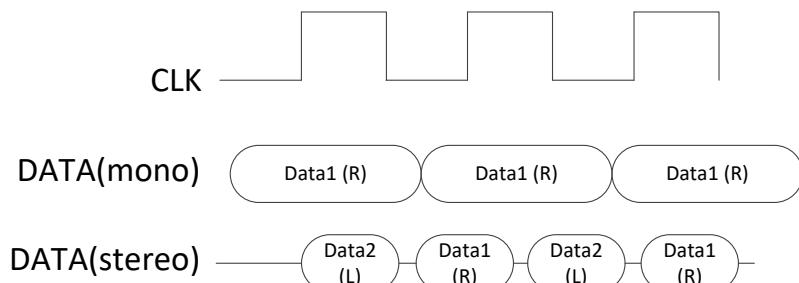
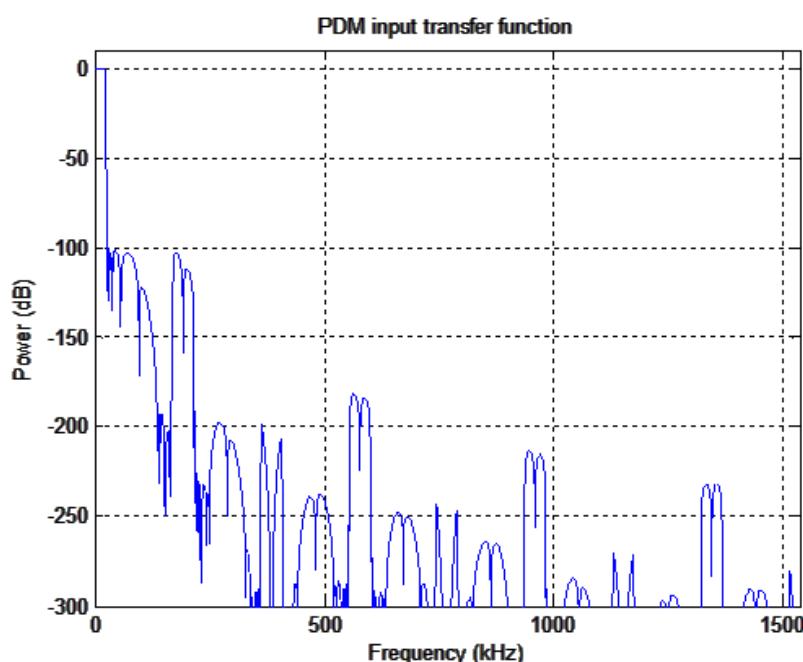
Settings for IOM mode:

- PCM_FSC_EDGE: 0 (rising edge PCM_FSC)
- PCM_FSCLEN: 0 (one cycle)
- PCM_FSC_DEL: 1 (no bit delay)
- PCM_CLK_INV: 0 (output on rising edge)
- PCM_CH0_DEL: 0 (no delay)
- PCM_CLK_BIT: 1

Multi-core Bluetooth 5.2 SoC family with system PMU

Figure 66: IOM format
29.2.4 PDM Interface

The PDM comprises two signals, namely the DATA and the CLK, and supports stereo streams. PDM_DATA is encoded so that the left channel is clocked in on the falling edge of CLK and the right channel is clocked on the rising edge of PDM_CLK as shown in Figure 67.

The interface supports MEMS microphone sleep mode by disabling the PDM_CLK. The PDM interface signals can be mapped on any GPIO by programming PID=32 and PID=33 for DATA and CLK respectively in the Px_yz_MODE_REG.


Figure 67: PDM Mono/Stereo formats

Figure 68: SRC PDM Input Transfer Function

It should be noted that the audio quality degrades when the oversampling ratio is less than 64. For an 8 kHz sample rate the minimum recommended PDM clock rate is $64 \times 8 \text{ kHz} = 512 \text{ kHz}$.

29.2.5 DMA Support

If more than one sample needs to be transferred to or from the CPU, or the sample rate is so high that it interrupts the CPU too often, the DMA controller must be engaged to perform the transactions. Three channels are reserved in the DMA to support the PCM, the SRC (IN) and the SRC (OUT) directions.

29.2.6 Interrupts

After a Sample Rate Conversion, the input up-sampler and output down-sampler, generate edge triggered interrupts on SRC_IN_SYNC and SRC_OUT_SYNC to the CPU which do not have to be cleared. Note that only one sample shall be read from or written to a single register at a time (i.e. there are no FIFOs included).

29.3 Programming

29.3.1 PDM Input to PCM Output

There is a simple sequence of steps that needs to be followed to configure the Audio Unit:

1. Configure the GPIOs functionality used for the PDM and PCM I/F by writing the appropriate Px_yy_MODE_REG[PID] = 32,33 (PDM) and 28-31 (PCM).
2. Configure the GPIOs direction (Px_yy_MODE_REG[PUPD]).
3. Configure PDM I/F:
 - a. Configure as Master by setting the PDM_DIV_REG[PDM_MASTER_MODE] bit.
 - b. Set PDM clock divider (PDM_DIV_REG[PDM_DIV]).
 - c. Enable PDM (internal) block clock (PDM_DIV_REG[CLK_PDM_EN]).
4. Configure PCM I/F (Actual word size should be taken into consideration to calculate the values of PCM_FSC_DIV_REG and PCM_FSCLEN):
 - a. Select the clock source (PCM_DIV_REG[PCM_SRC_SEL]).
 - b. Setup PCM clock division (PCM_DIV_REG[PCM_DIV], PCM_FDIV_REG).
 - c. Enable the clock (master mode) by setting the PCM_DIV_REG[CLK_PCM_EN] bit.
 - d. Disable PCM (PCM1_CTRL_REG[PCM_EN] = 0).
 - e. Set PCM Framesync divider (PCM1_CTRL_REG[PCM_FSC_DIV]).
 - f. (PCM1_CTRL_REG[PCM_FSC_EDGE]).
 - g. Set channel delay in multiples of 8 bits (PCM1_CTRL_REG[PCM_CH_DEL]).
 - h. Set the number of clock cycles per data bit (PCM1_CTRL_REG[PCM_CLK_BIT]).
 - i. Set polarity of PCM FSC (PCM1_CTRL_REG[PCM_FSCINV]).
 - j. Set polarity of PCM CLK(PCM1_CTRL_REG[PCM_CLKINV]).
 - k. Set PCM DO output mode (PCM1_CTRL_REG[PCM_PPOD]).
 - l. Set PCM FSC start time (PCM1_CTRL_REG[PCM_FSCDEL]).
 - m. Set PCM FSC data length (PCM1_CTRL_REG[PCM_FSCLEN]).
 - n. Set PCM in Master mode (PCM1_CTRL_REG[PCM_MASTER] = 1).
5. Configure the Sample Rate Converter:
 - a. Set the SRC clock divider (SRC_DIV_REG[SRC_DIV]).
 - b. Enable the SRC block clock by setting the SRC_DIV_REG[CLK_SRC_EN] bit.
 - c. Select the SRC input Up Sampling IIR filters setting according to the sample rate (SRC1_CTRL_REG[SRC_IN_DS]).

Multi-core Bluetooth 5.2 SoC family with system PMU

- d. Configure the SRC input sample rate (SRC1_IN_FS_REG).
- e. Select the SRC output Up Sampling IIR filters setting according to the sample rate (SRC1_CTRL_REG[SRC_OUT_US]).
- f. Configure the SRC output sample rate (SRC1_OUT_FS_REG).
- g. Select the PDM as input to SRC (APU_MUX_REG[PDM1_MUX_IN] = 1).
- h. Enable the SRC FIFO (SRC1_CTRL_REG[SRC_FIFO_ENABLE]) and set direction (SRC1_CTRL_REG[SRC_FIFO_DIRECTION] = 1). Note that in stereo mode, FIFO cannot be used!
 - i. Select the output to PCM (APU_MUX_REG[PCM1_MUX_IN] = 1).
 - j. Set APU_MUX_REG[SRC1_MUX_IN] = 0.
 - k. Set SRC input to Automatic conversion mode (SRC1_CTRL_REG[SRC_IN_AMODE] = 1).
6. Enable SRC (SRC1_CTRL_REG[SRC_EN] = 1).
7. Enable PCM (PCM1_CTRL_REG[PCM_EN] = 1).

30 I2C Interface

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

30.1 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 two-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, A/D and D/A converters. It comprises 32 levels deep in both directions.

Features

- Two-wire I2C serial interface consists of a serial data line (SDA) and a serial clock (SCL)
- Three speeds are supported:
 - Standard mode (0 to 100 kbit/s)
 - Fast mode (<= 400 kbit/s)
 - High Speed mode (<= 3.4 Mbit/s)
- Clock synchronization
- 32 locations deep transmit/receive FIFOs (32 x 8-bit Rx, 32 x 10-bit Tx)
- Master transmit, Master 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

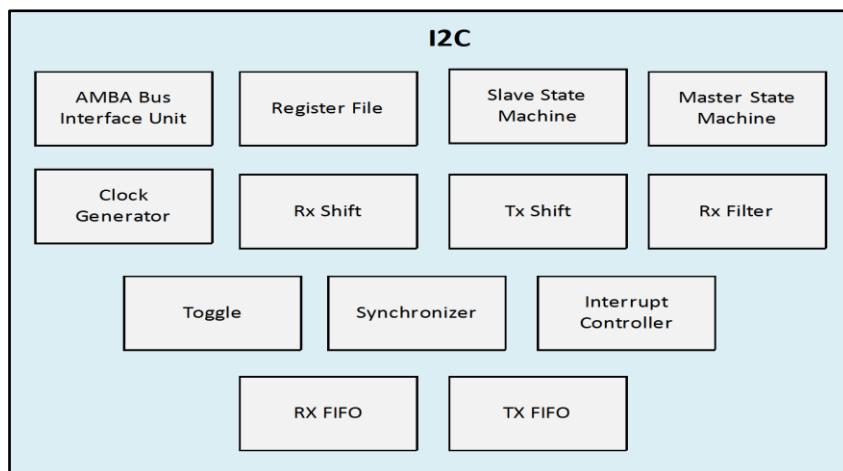


Figure 69: I2C Controller Block Diagram

The I2C Controller block diagram is shown in [Figure 69](#). It contains the following sub-blocks:

- AMBA Bus Interface Unit. Interfacing via the APB interface to access the register file
- Register File. Contains configuration registers and is the interface with software
- Master State Machine. Generates the I2C protocol for the master transfers

Multi-core Bluetooth 5.2 SoC family with system PMU

- Clock Generator. Calculates the required timing to do the following:
 - Generate the SCL clock when configured as a master
 - Check for bus idle
 - Generate a START and a STOP
 - Setup the data and hold the data
- Rx Shift. Takes data into the design and extracts it in byte format
- Tx Shift. Presents data supplied by CPU for transfer on the I2C bus
- Rx Filter. Detects the events in the bus; for example, start, stop and arbitration lost
- Toggle. Generates pulses on both sides and toggles to transfer signals across clock domains
- Synchronizer. Transfers signals from one clock domain to another
- Interrupt Controller. Generates the raw interrupt and interrupt flags, allowing them to be set and cleared
- RX FIFO/TX. Holds the RX FIFO and TX FIFO register banks and controllers, along with their status levels

30.2 Architecture

30.2.1 I2C Behavior

The I2C can be controlled (through software) to be a I2C master only, communicating with other I2C slaves

The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for transmitting or receiving data to and from the master. Data acknowledgement is sent by the device that receives data, which can be master or slave. The I2C protocol allows multiple masters to reside on the I2C bus. It uses an arbitration procedure to determine bus ownership.

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 if the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge pulse (ACK) after the address.

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver gets one byte of data. This transaction continues until the master terminates the transmission with a STOP condition. If the master is reading from a slave (master-receiver), the slave transmits (slave-transmitter) a byte of data to the master, and the master then acknowledges the transaction with the 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 illustrated in [Figure 70](#).

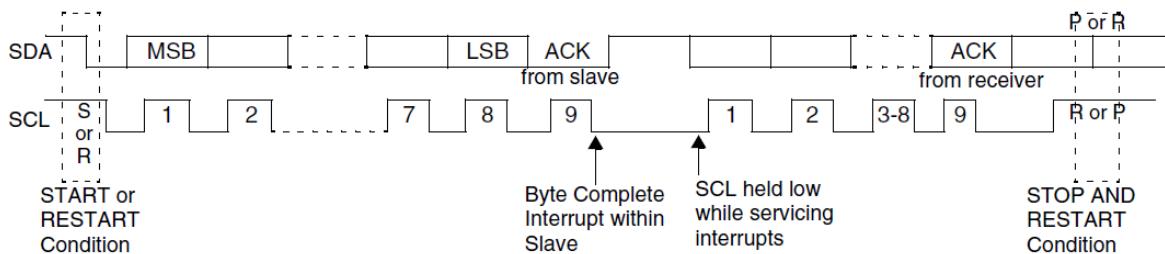


Figure 70: Data Transfer on the I2C Bus

The I2C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only while 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

Multi-core Bluetooth 5.2 SoC family with system PMU

devices on the bus is limited by only the maximum capacitance specification of 400 pF. Data is transmitted in byte packages.

30.2.1.1 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. Writing a 1 to I2C_DATA_CMD_REG[9] causes the i2c to generate a STOP condition on the I2C bus; a STOP condition is not issued if this bit is not set, even if the transmit FIFO is empty.

When operating as a slave, the I2C Controller does not generate START and 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.

30.2.1.2 Combined Formats

The I2C Controller supports mixed read and write combined format transactions 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 - combined format transactions.

To initiate combined format transfers, I2C_CON.I2C_RESTART_EN should be set to 1. With this value set and operating as a master, when the I2C Controller completes an I2C transfer, it checks the transmit FIFO and executes the next transfer. If the direction of this transfer differs from the previous transfer, 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 following a START condition.

30.2.2 I2C Protocols

The I2C Controller has the following protocols:

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

30.2.2.1 START and STOP Conditions

When the bus is idle, both the SCL and SDA signals are pulled high through external pull-up resistors on the bus. When the master wants to start a transmission on the bus, the master 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, the master issues a STOP condition. This is defined to be a low-to-high transition of the SDA line while SCL is 1. [Figure 71](#) 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.

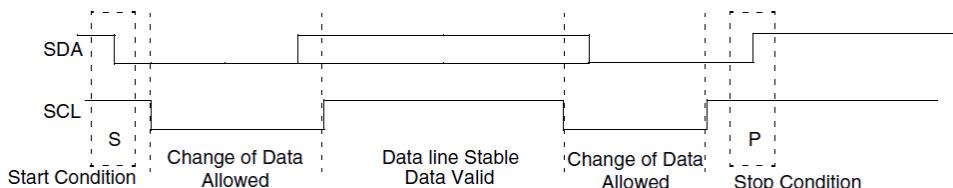


Figure 71: START and STOP Conditions

Multi-core Bluetooth 5.2 SoC family with system PMU

Note: The signal transitions for the START/STOP conditions, as depicted in [Figure 71](#), reflect those observed at the output signals of the Master driving the I2C bus. Care should be taken when 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.

30.2.2.2 Addressing Slave Protocol

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

7-bit Address Format

During 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 as shown in [Figure 72](#). 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.

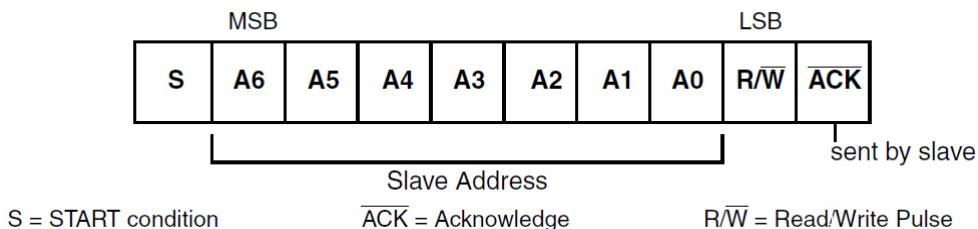


Figure 72: 7-bit Address Format

10-bit Address Format

During 10-bit addressing, 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 followed by the next two bits (bits 2:1), which 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 73](#) shows the 10-bit address format, and [Table 139](#) defines the special purpose and reserved first byte addresses.

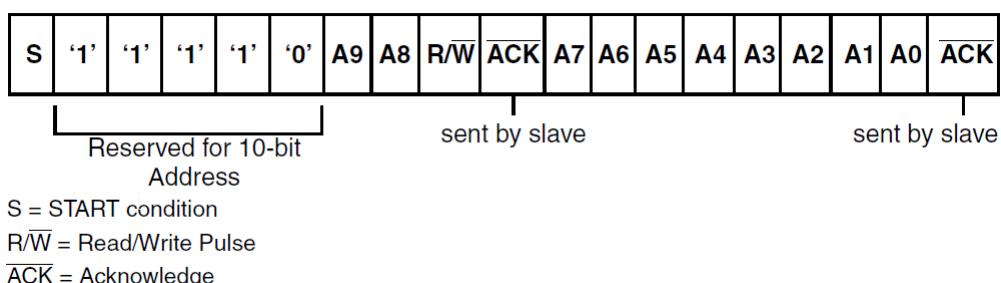


Figure 73: 10-bit Address Format

Table 139: I2C Definition of Bits in First Byte

| 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, refer to "START BYTE Transfer Protocol" |
| 0000 001 | X | CBUS address. I2C Controller ignores these accesses |
| 0000 010 | X | Reserved |
| 0000 011 | X | Reserved |

| Slave Address | R/W Bits | Description |
|---------------|----------|---|
| 0000 1XX | X | High-speed master code (for more information, refer to “Multiple Master Arbitration”) |
| 1111 1XX | X | Reserved |
| 1111 0XX | X | 10-bit slave addressing |

The I2C Controller does not restrict you from using these reserved addresses. However, if you use these reserved addresses, you may run into incompatibilities with other I2C components.

30.2.2.3 Transmitting and Receiving Protocols

The 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 the master to either transmit data or receive data to/from the bus, acting as either a slave-transmitter or slave-receiver, respectively.

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 the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, 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 the master-transmitter is transmitting data as shown in [Figure 74](#), then the slave-receiver responds to the master-transmitter with an acknowledge pulse after every byte of data is received.

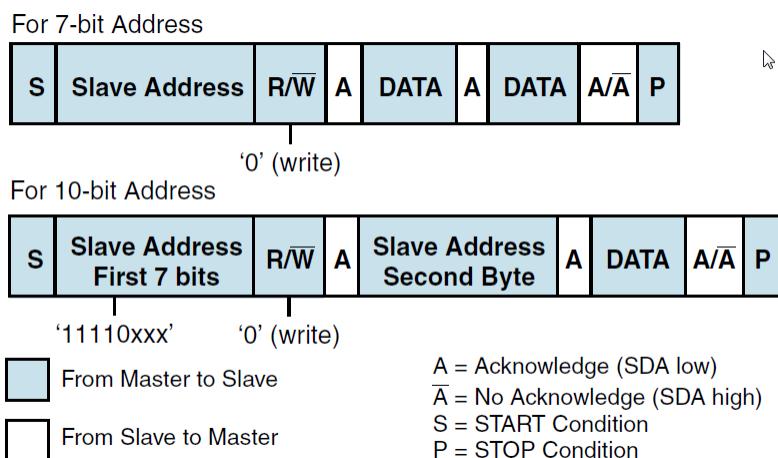


Figure 74: Master-Transmitter Protocol

Master-Receiver and Slave-Transmitter

If the master is receiving data as shown in [Figure 75](#) then the master responds to the slave-transmitter with an acknowledge pulse 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 pulse. The master can then communicate with the same slave or a different slave.

Multi-core Bluetooth 5.2 SoC family with system PMU

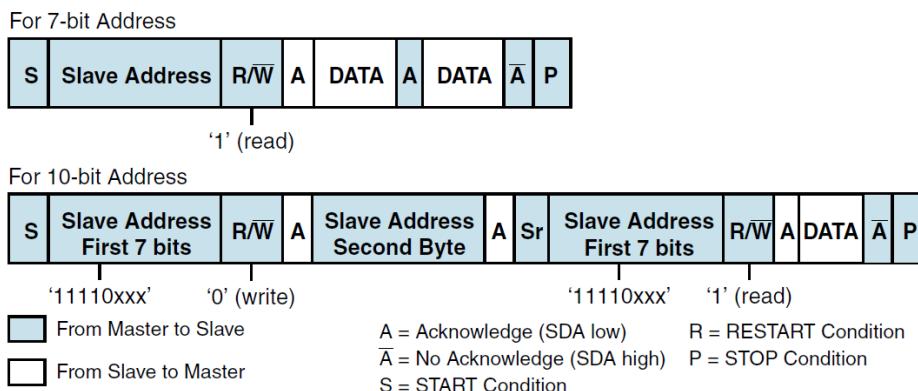


Figure 75: Master-Receiver Protocol

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 being transmitted followed by a 1, as illustrated in Figure 76. This allows the processor that is polling the bus to undersample the address phase until 0 is detected. Once the microcontroller detects a 0, it switches from the undersampling rate to the correct rate of the master.

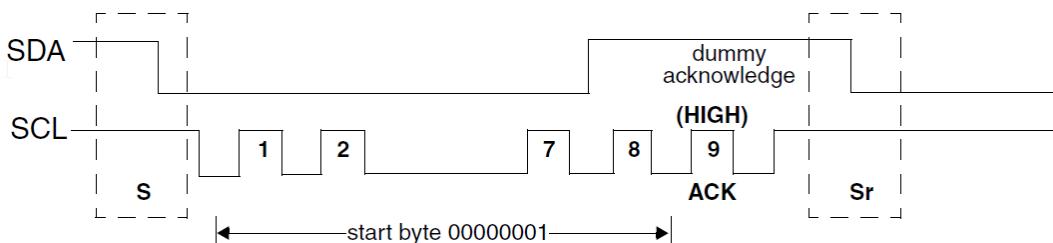


Figure 76: START BYTE Transfer

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 resets after the RESTART condition is generated.

30.2.3 Multiple Master Arbitration

The I2C Controller bus protocol allows multiple masters to reside on the same bus. If there are two masters on the same I2C-bus, there is an arbitration procedure if both try to take control of the bus at the same time by generating a START condition at the same time. Once 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 0, loses arbitration and turns off its data output stage. The master that lost arbitration can continue to generate clocks until the end of the byte transfer. If both masters

Multi-core Bluetooth 5.2 SoC family with system PMU

are addressing the same slave device, the arbitration could go into the data phase. [Figure 77](#) illustrates the timing of when two masters are arbitrating on the bus.

For high-speed mode, the arbitration cannot go into the data phase because each master is programmed with a unique high-speed master code. This 8-bit code is defined by the system designer and is set by writing to the High-Speed Master Mode Code Address Register, I2C_HS_MADDR. Because the codes are unique, only one master can win arbitration, which occurs by the end of the transmission of the high-speed master code.

Control of the bus is determined by address or master code and data sent by 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.

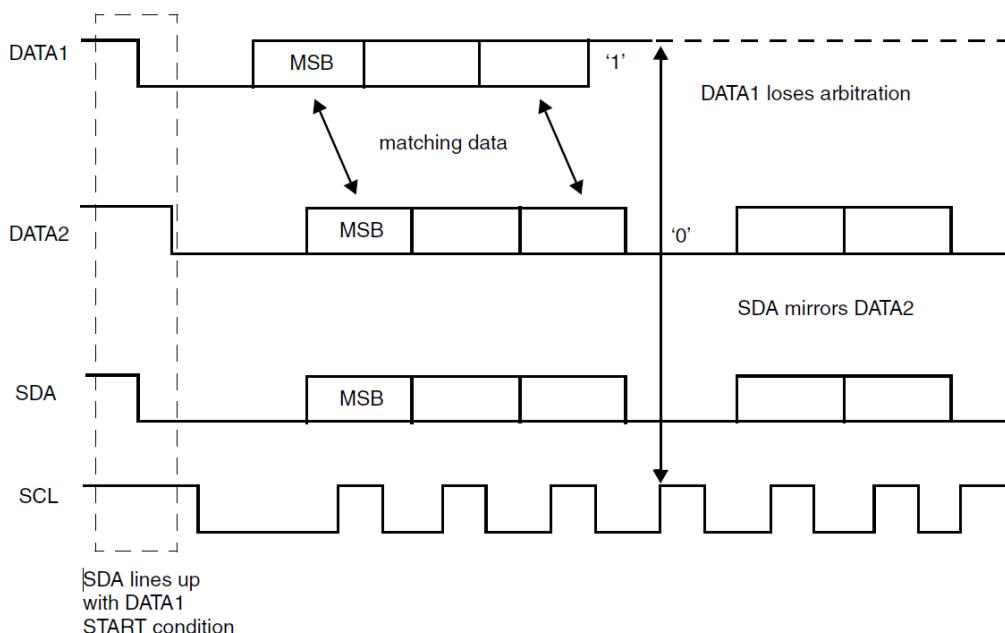


Figure 77: Multiple Master Arbitration

30.2.4 Clock Synchronization

When two or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters generate their own clock to transfer messages. Data is valid only during the high period of 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 time 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, then the master goes into a HIGH wait state until the SCL clock line transitions to 1.

All masters then count off 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. 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 78](#). Optionally, slaves may hold the SCL line low, to slow down the timing on the I2C bus.

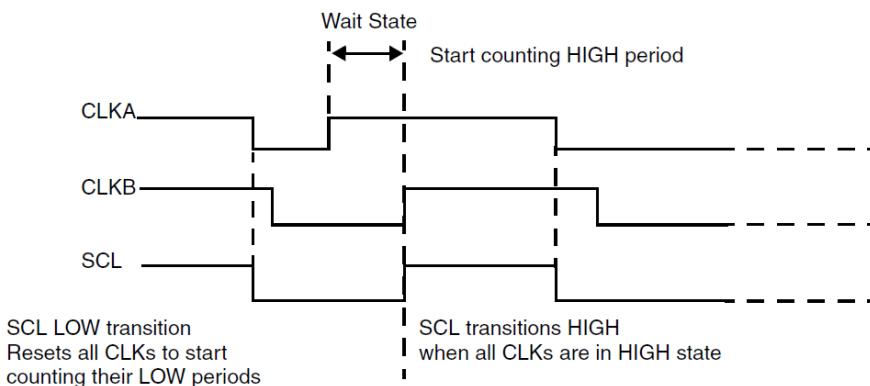


Figure 78: Multiple Master Clock Synchronization.

30.3 Programming

There is a simple sequence of steps that needs to be followed to configure and use the I2C Controllers:

1. Setup the GPIOs to be used for the I2C interface ($Px_yy_MODE_REG[PID] = 21$ to 24).
2. Configure I2C clock frequency.
For $CLK_COM_REG[I2Cx_CLK_SEL] = 0$ (DivN clock):
 - a. Standard mode (100 kbit/s) : $I2Cx_CON_REG[I2C_SPEED] = 1$.
 - b. Full speed mode (400 kbit/s) : $I2Cx_CON_REG[I2C_SPEED] = 2$.
 - c. High speed mode (1 Mbit/s) : $I2Cx_CON_REG[I2C_SPEED] = 3$.
3. Setup the Controller as:
 - a. Master : $I2Cx_CON_REG[I2C_MASTER_MODE] = 1$ and $I2Cx_CON_REG[I2C_SLAVE_DISABLE] = 1$.
 - b. Slave : $I2Cx_CON_REG[I2C_MASTER_MODE] = 0$ and $I2Cx_CON_REG[I2C_SLAVE_DISABLE] = 0$.
4. Choose whether the controller starts its transfers in 7 or 10-bit addressing mode when acting as a master ($I2Cx_CON_REG[I2C_10BITADDR_MASTER]$) or when acting as a slave, whether the controller responds to 7- or 10-bit addresses ($I2Cx_CON_REG[I2C_10BITADDR_SLAVE]$).
5. Set target slave address in:
 - a. Master mode ($I2Cx_TAR_REG[IC_TAR] = 0x55$ (default)).
 - b. Slave mode ($I2Cx_SAR_REG[IC_SAR] = 0x55$ (default)).
6. Set threshold level on RX and TX FIFO ($I2Cx_RX_TL_REG$, $I2Cx_TX_TL_REG$).
7. Enable the required interrupts ($I2Cx_INTR_MASK_REG$).
8. Enable the I2C Controller by setting the $CLK_COM_REG[I2Cx_ENABLE]$ bit.
9. Read a byte:
 - a. Prepare to transmit the read command byte ($I2Cx_DATA_CMD_REG[I2C_CMD] = 1$).
 - b. Wait until TX FIFO is empty ($I2Cx_STATUS_REG[TFE] = 1$).
 - c. Wait until master has finished reading the byte from slave device ($I2Cx_STATUS_REG[MST_ACTIVITY] = 0$).
10. Write a byte:
 - a. Prepare to transmit the write command byte ($I2Cx_DATA_CMD_REG[I2C_CMD] = 0$ and $I2Cx_DATA_CMD_REG[I2C_DAT] = command\ byte$).
 - b. Wait until TX FIFO is empty ($I2Cx_STATUS_REG[TFE] = 1$).
 - c. Wait until master has finished reading the response byte from slave device ($I2Cx_STATUS_REG[MST_ACTIVITY] = 0$).

31 UART

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

31.1 Introduction

The DA1469x contains three instances of this block. These are: UART, UART2, and UART3.

The UART is compliant to 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. It is converted to 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, so the internal FIFOs can be used. UART2 and UART3 support hardware flow control signals (RTS, CTS).

Features

- Dedicated 16 bytes Transmit and 16 bytes Receive FIFO for each UART
- Hardware flow control and 9-bit mode support (CTS/RTS, UART2 and UART3)
- Shadow registers reduces software overhead and include a software programmable reset
- Transmitter Holding Register Empty (THRE) interrupt mode
- Functionality based on 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 as calculated by the following: baud rate = (serial clock frequency)/(16 x divisor) and the fractional part UART_DLF/16.
- ISO7816 support (UART3)

Multi-core Bluetooth 5.2 SoC family with system PMU

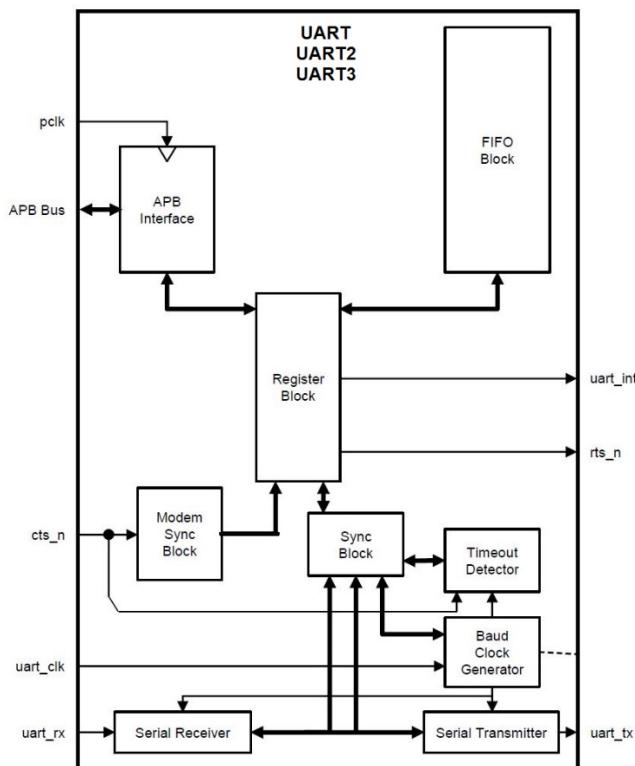


Figure 79: UART Block Diagram

31.2 Architecture

31.2.1 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 start and stop bits is referred to as a character, as shown in Figure 80.

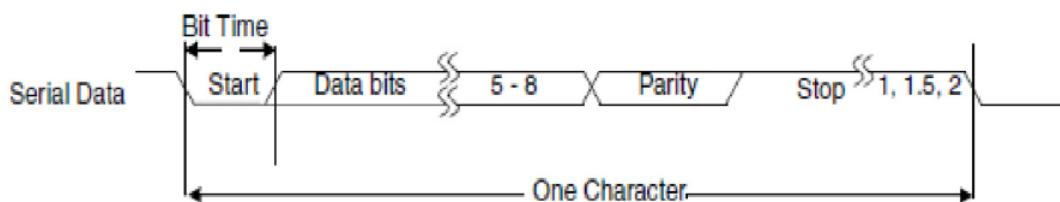


Figure 80: Serial Data Format

An additional parity bit may be added to the serial character. This bit appears after the last data bit, but before the stop bit(s) in the character structure. It provides 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, followed by the stop bit(s), which can be 1, 1.5 or 2.

All the bits in the transmission (with exception of the half-stop bit when 1.5 stop bits are used) are transmitted for exactly the same time duration. This is referred to as a Bit Period or Bit Time. One Bit Time equals 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, once the start bit has been detected. As the exact number of baud clocks that each bit was transmitted for is known, calculating the mid-point for

Multi-core Bluetooth 5.2 SoC family with system PMU

sampling is not difficult, that is every 16 baud clocks after the mid-point sample of the start bit. [Figure 81](#) shows the sampling points of the first couple of bits in a serial character.

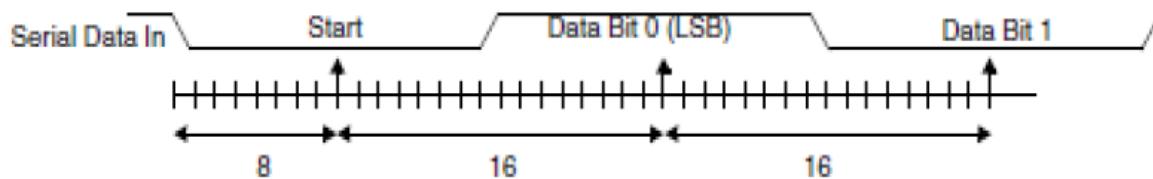


Figure 81: Receiver Serial Data Sampling Points

As part of the 16550 standard, an optional baud clock reference output signal (`baudout_n`) is supplied to provide timing information to 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). The registers settings for common baud rate values are presented in the following tables:

Table 140: UART Baud Rate Generation

| Baud Rate (Note 1) | Divider | Divisor Latch | DLH Reg | DLL Reg | DLF Reg | Actual BR | Error % |
|---------------------------------------|----------|---------------|---------|---------|---------|-----------|---------|
| 1200 | 1666.667 | 1666.6875 | 6 | 130 | 11 | 1199.99 | 0.00 |
| 2400 | 833.333 | 833.3125 | 3 | 65 | 5 | 2400.06 | 0.00 |
| 4800 | 416.667 | 416.6875 | 1 | 160 | 11 | 4799.76 | 0.00 |
| 9600 | 208.333 | 208.3125 | 0 | 208 | 5 | 9600.96 | 0.01 |
| 14400 | 138.889 | 138.875 | 0 | 138 | 14 | 14401.44 | 0.01 |
| 19200 | 104.167 | 104.1875 | 0 | 104 | 3 | 19196.16 | 0.02 |
| 28800 | 69.444 | 69.4375 | 0 | 69 | 7 | 28802.88 | 0.01 |
| 38400 | 52.083 | 52.0625 | 0 | 52 | 1 | 38415.37 | 0.04 |
| 57600 | 34.722 | 34.75 | 0 | 34 | 12 | 57553.96 | 0.08 |
| 115200 | 17.361 | 17.375 | 0 | 17 | 6 | 115107.91 | 0.08 |
| 230400 | 8.681 | 8.6875 | 0 | 8 | 11 | 230215.83 | 0.08 |
| 460800 | 4.340 | 4.3125 | 0 | 4 | 5 | 463768.12 | 0.64 |
| 921600 | 2.170 | 2.1875 | 0 | 2 | 3 | 914285.71 | 0.79 |
| 1000000 | 2 | 2 | 0 | 2 | 0 | 1000000 | 0.00 |

Note 1 Values are valid for UART CLK = 32 MHz (`divN_clk`).

Table 141: UART2/3 Baud Rate Generation

| Baud Rate (Note 2) | Divider | Divisor Latch | DLH Reg | DLL Reg | DLF Reg | Actual BR | Error % |
|---------------------------------------|----------|---------------|---------|---------|---------|-----------|---------|
| 1200 | 5000.000 | 5000 | 19 | 136 | 0 | 1200.00 | 0.00 |
| 2400 | 2500.000 | 2500 | 9 | 196 | 0 | 2400.00 | 0.00 |
| 4800 | 1250.000 | 1250 | 4 | 226 | 0 | 4800.00 | 0.00 |
| 9600 | 625.000 | 625 | 2 | 113 | 0 | 9600.00 | 0.00 |
| 14400 | 416.667 | 416.6875 | 1 | 160 | 11 | 14399.28 | 0.00 |
| 19200 | 312.500 | 312.5 | 1 | 56 | 8 | 19200.00 | 0.00 |
| 28800 | 208.333 | 208.3125 | 0 | 208 | 5 | 28802.88 | 0.01 |

| Baud Rate (Note 2) | Divider | Divisor Latch | DLH Reg | DLL Reg | DLF Reg | Actual BR | Error % |
|-----------------------|---------|---------------|---------|---------|---------|-----------|---------|
| 38400 | 156.250 | 156.25 | 0 | 156 | 4 | 38400.00 | 0.00 |
| 57600 | 104.167 | 104.1875 | 0 | 104 | 3 | 57588.48 | 0.02 |
| 115200 | 52.083 | 52.0625 | 0 | 52 | 1 | 115246.10 | 0.04 |
| 230400 | 26.042 | 26.0625 | 0 | 26 | 1 | 230215.83 | 0.08 |
| 460800 | 13.021 | 13 | 0 | 13 | 0 | 461538.46 | 0.16 |
| 921600 | 6.510 | 6.5 | 0 | 6 | 8 | 923076.92 | 0.16 |
| 1000000 | 6 | 6 | 0 | 6 | 0 | 1000000 | 0.00 |
| 3000000 | 2 | 2 | 0 | 2 | 0 | 3000000 | 0.00 |

Note 2 Values are valid for CLK_COM_REG[UART2/3_CLK_SEL] = 1 and sys_clk = 96MHz. For CLK_COM_REG[UART2/3_CLK_SEL] = 0, see [Table 140](#).

31.2.2 Clock Support

The UART has two system clocks (*pclk* and *sclk*). Having the second asynchronous serial clock (*sclk*) implemented, accommodates accurate serial baud rate settings, as well as APB bus interface requirements.

With the two-clock design, a synchronization module is implemented for synchronization of all control and data across the two system clock boundaries.

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. However, 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 initial configuration.

31.2.3 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 142](#).

Table 142: 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 |

| Interrupt ID | Interrupt Set and Reset Functions | | | |
|--------------|-----------------------------------|------------------------------------|---|--|
| 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) |
| 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 (if source of interrupt); 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 | Busy detect | Line Control Register was written while the UART is busy (RX or TX line is low) | Reading the UART status register |

31.2.4 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 via 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 opposed to empty, as shown in the flowchart in [Figure 82](#).

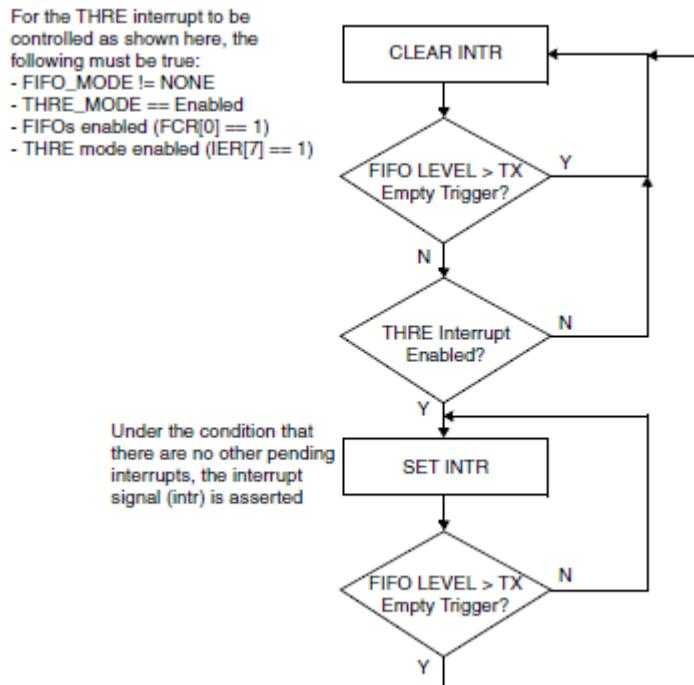


Figure 82: Flowchart of Interrupt Generation for Programmable THRE Interrupt Mode

This threshold level is programmed into FCR[5:4]. The available empty thresholds are: empty, 2, $\frac{1}{4}$ and $\frac{1}{2}$. 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 function from indicating transmitter FIFO empty, to FIFO full. This allows software to fill the FIFO each transmit sequence, by polling LSR[5] before writing another character. The flow then becomes, "fill transmitter FIFO whenever an interrupt occurs and there is data to transmit", instead of waiting until the FIFO is completely empty. 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 via 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 83](#).

Multi-core Bluetooth 5.2 SoC family with system PMU

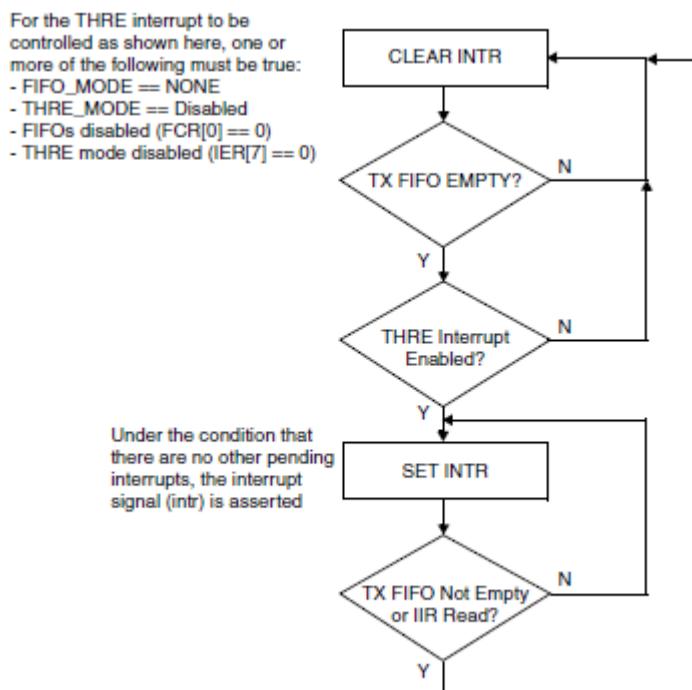


Figure 83: Flowchart of Interrupt Generation When Not in Programmable THRE Interrupt Mode

31.2.5 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 support a host burst mode where the host increments its address, but still accesses the same Receive buffer register.
- UART_STHR support 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.

31.2.6 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 using a 2-wire UART directly as described in the *Bluetooth Low Energy Specification (Volume 6, Part F)*.

31.3 Programming

There is a simple sequence of steps that need to be followed to configure and use the UART controllers:

1. Setup the GPIOs to be used for the UART interface (Px_yy_MODE_REG[PID] = 1 to 10).
2. Select the UART clock (CLK_COM_REG[UART2/3_CLK_SEL]). This step is not needed when UART is used.
3. Enable the selected UART by setting the CLK_COM_REG[UARTx_ENABLE] bit.

Multi-core Bluetooth 5.2 SoC family with system PMU

4. Enable access to Divisor Latch Registers (DLL and DLH) by setting the `UARTx_LCR_REG[UART_DLAB]` bit.
5. Set the desired baud rate. To calculate the registers values for the desired baud rate, use the formula: $\text{Divisor} = \text{UART CLK} / (16 \times \text{Baud rate})$.
 - 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.
6. Configure the brake control bit, parity, number of stop bits and data length (`UARTx_LCR_REG`).
7. Enable and configure the FIFO (`UARTx_IIR_FCR_REG`).
8. Configure the generated interrupts, if needed (`UARTx_IER_DLH_REG`).
9. 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 was transmitted (`UARTx_LSR_REG[UART_TEMT]`).
10. 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`).

32 Smart Card Interface

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

32.1 Introduction

The DA1469x features a Smart Card interface implemented using the UART3 block. The UART3 block supports asynchronous protocol smartcards as defined in the ISO 7816-3 (Class B and C) standard.

Features

- ISO/IEC 7816-3 (Class B and C) compliant
- UART Line with Flow Control Signals
- Interrupt line
- DMA support
- An Error signal support with indication for character repetition
- Inverse and direct convention
- Guard time

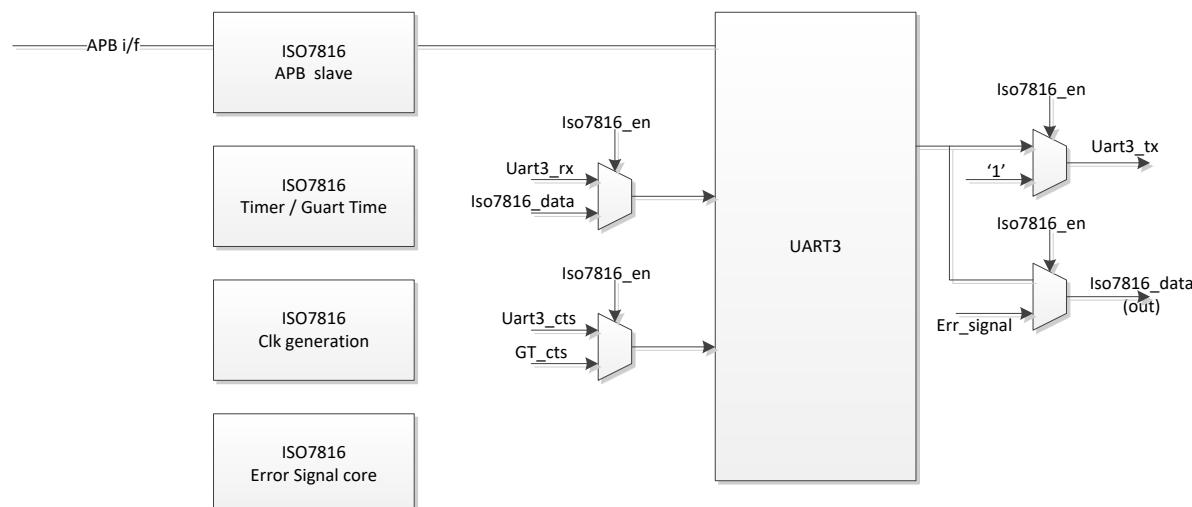


Figure 84: Smart Card (ISO7816-3) Block Diagram

32.2 Architecture

32.2.1 ISO7816-3 Clock Generation

The ISO7816-3 block operates on a system based clock with a 50% duty cycle. The clock frequency is calculated using the formula:

$$F_{SCLK} / [2 * (UART3_CTRL_REG[ISO7816_CLK_DIV] + 1)]$$

The `UART3_CTRL_REG[ISO7816_CLK_DIV]` value can be updated at any time, whether the ISO7816-3 clock is enabled or not. The operation of the clock can be checked at any given time by polling the `UART3_CTRL_REG[ISO7816_CLK_STATUS]` bit, and disabled by clearing the `ISO7816_CLK_EN[ISO7816_CLK_EN]` bit. The `UART3_CTRL_REG[ISO7816_CLK_LEVEL]` reflects the logical level of the clock while it has been disabled.

Multi-core Bluetooth 5.2 SoC family with system PMU

32.2.2 ISO7816-3 Timer – Guard Time

The ISO7816-3 block is equipped with an ISO7816-3 Timer/Guard Timer and can operate in 2 modes:

- UART3_TIMER_REG[ISO7816_TIM_MODE] = 0.
The timer is clocked with the clock of the ISO7816 module and will start when UART3_TIMER_REG[ISO7816_TIM_EN] = 1. It will count from 0 to UART3_TIMER_REG[ISO7816_TIM_MAX] value. The value of the timer can be read from UART3_TIMER_REG[ISO_TIM_MAX] register field.
At the point where the top value is reached, the timer will stop and the UART3_IRQ_STATUS_REG[ISO7816_TIM_EXPIRED IRQ] will be set. If the UART3_CTRL_REG[ISO7816_TIM_EXPIRED_IRQMASK] is 1, an interrupt will be generated. The interrupt will be cleared when UART3_TIMER_REG[ISO7816_TIM_EN] is set.
- UART3_TIMER_REG[ISO7816_TIM_MODE] = 1.
The timer is clocked with the 1/16 of the UART bit clock (1/16 etu). If the UART fractional divider has been set, the 1/16 of the UART bit clock will not have a fixed value. If the UART3_TIMER_REG [ISO7816_TIM_EN] bit is set, the timer will start counting from 0 to UART3_TIMER_REG[ISO_TIM_MAX] value, each time a start bit is transmitted by the UART. If the UART3_TIMER_REG[ISO_TIM_MAX] value is set equal to 16*GuartTime-1, the timer will count the minimum delay between the leading edges of two consecutive characters (Guard time). In case of no FIFO mode and GT>12 etu, the expiration of the timer indicates that the module is allowed to send the next character (Guard Time elapsed). In case of FIFO mode and GT>12 etu, the UART3_CTRL_REG[ISO7816_AUTO_GT] bit can be set so the module will be able to send the next character automatically each time the Guard Time is reached.

32.2.3 ISO7816-3 Error Detection

The ISO7816-3 block is designed to support the functionality described in the section 7.3 of ISO7816-3 specification document.

Transmit Phase

The transmitter checks the ISO7816 data level for 11etu sampling time after a character's leading edge is detected. The module samples the data line at 11etu time and creates two types of interrupts:

1. The ISO7816_ERR_TX_TIME_IRQ interrupt is created each time a character is sent.
2. The ISO7816_ERR_TX_VALUE_IRQ interrupt is created each time the receiver sends an error.

The software has to check if an ISO7816_ERR_TX_VALUE_IRQ interrupt is generated in order to retransmit the character. The IRQs are generated at the same time.

Receive Phase

The receiver will hold the data line level Low between 1 etu (minimum) and 2 etu (maximum) at 10.5 etu time. The error detection circuit will use the UART parity check and create an error signal to the transmitter at the proper time. The error signal pulse width and offset can be configured using the UART3_ERR_CTRL_REG register. Furthermore, the ISO7816-3 block can hold the UART_Rx signal High during the error signal transmitting in order to prevent UART errors from happening.

32.3 Programming

There is a simple sequence of steps that needs to be followed to configure the Smart Card Controller:

3. Setup the GPIOs to be used for the ISO7816-3 interface (Px_yy_MODE_REG[PID] = 11 to 12). Reset (output) and card insert (input) signals shall be configured as GPIOs (Px_yy_MODE_REG[PID] = 0).
4. Enable UART3 by setting the CLK_COM_REG[UART3_ENABLE] bit.
5. Setup UART3 clock (CLK_COM_REG[UART3_CLK_SEL]).
6. Enable the ISO7816-3 module by setting the UART3_CONFIG_REG[ISO7816_ENABLE] bit.

Multi-core Bluetooth 5.2 SoC family with system PMU

7. Setup the ISO7816-3 clock (UART3_CTRL_REG[ISO7816_CLK_DIV]).
8. Initialize UART3 as follows:
 - a. Configure FIFO, if needed (UART3_IIR_FCR_REG, UART3_SRT_REG, UART3_STET_REG, UART3_SFE_REG). Note that if Error detection is enabled, the TX FIFO shall remain disabled.
 - b. Configure ISO7816 convention:
9. Select data length (UART3_LCR_REG[UART_DLS]):
 - i. Select the number of stop bits (UART3_LCR_REG[UART_STOP]).
 - ii. Enable/disable parity (UART3_LCR_REG[UART_PEN]).
 - iii. Select even or odd parity (UART3_LCR_REG[UART_EPS]).
 - iv. If needed, enable Error detection (UART3_CONFIG_REG[ISO7816_ERR_SIG_EN]) and configure the error pulse width and offset (UART3_ERR_CTRL_REG).
 - v. Select direct/inverse convention (UART3_CONFIG_REG[ISO7816_CONVENTION]).
10. Configure the baud rate. To calculate the registers values for the desired baud rate, use the formula: Divisor = $F_i * (ISO7816_CLK_DIV + 1) / (8 * D_i)$. For F_i and D_i values, refer to ISO/IEC 7816-3 Standard Specification, table 7 and table 8:
 - i. Enable access to the Divisor Latch register: UART3_LCR_REG[UART_DLAB] = 1.
 - ii. Set the High byte of the Divisor integer part (UART3_IER_DLH_REG).
 - iii. Set the Low byte of the Divisor integer part (UART3_RBR_THR_DLL_REG).
 - iv. Set the fractional part of the Divisor (UART3_DLF_REG).
11. Configure the generated interrupts, if needed (UART3_CTRL_REG).
10. Send a byte:
 - a. Setup the Guard Timer (UART3_TIMER_REG).
 - b. In case of TX FIFO is enabled, check if FIFO is full (UART3_USR_REG[UART_TFNF]) or check if Transmit Hold Register (THR) is empty (UART3_LSR_REG[UART_THRE]).
 - c. Load the byte to THR (UART3_RBR_THR_DLL_REG).
 - d. Check if the byte was transmitted (UART3_LSR_REG[UART_TEMT]).
11. Receive a byte:
 - a. Setup the Wait Time (UART3_TIMER_REG).
 - b. Wait until serial data is ready (UART3_LSR_REG[UART_DR]) or timer expiration.
 - c. Read the received byte from the THR (UART3_RBR_THR_DLL_REG).

33 SPI+ Interface

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

33.1 Introduction

The DA1469x contains two instances of this block, namely, SPI and SPI2.

The SPI+ interface supports a subset of the Serial Peripheral Interface (SPI™). The serial interface can transmit and receive 8, 16 or 32 bits in master/slave mode and transmit 9 bits in master mode. The SPI+ interface has enhanced functionality with 8-byte RX and 4-byte TX FIFOs.

Features

- Slave and Master mode
- 8 bit, 9 bit, 16 bit or 32 bit operation
- Clock speeds up to 32 MHz for the SPI controller. Programmable output frequencies of SPI source clock divided by 2, 4, 8, 14
- SPI clock line speed up to 16 MHz
- SPI mode 0, 1, 2, 3 support (clock edge and phase)
- Programmable SPI_DO idle level
- Maskable Interrupt generation
- Bus load reduction by unidirectional writes-only and reads-only modes.
- Dedicated 8-byte RX and 4-byte TX FIFOs for each SPI block.
- DMA support

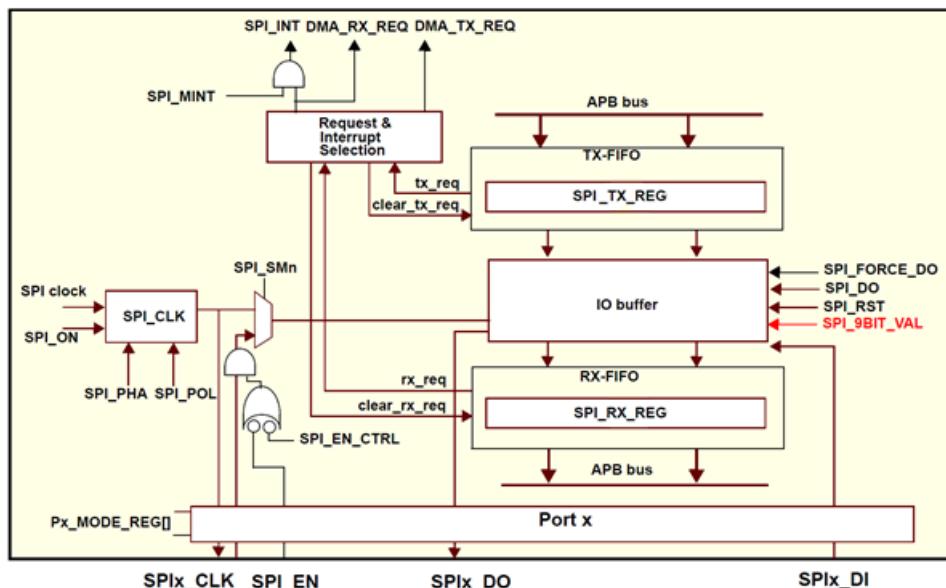


Figure 85: SPI Block Diagram

33.2 Architecture

33.2.1 Master Mode

This mode is the default mode. To enable SPITM operation, the individual port signal must first be enabled. Next, the SPI must be configured in SPI_CTRL_REG, for the desired mode. Finally, bit SPI_ON must be set to 1.

A SPI transfer cycle starts after writing to the SPI_RX_TX_REG. Writing to SPI_RX_TX_REG also sets the SPI_TXH. As soon as the holding register is copied to the IO buffer, the SPI_TXH is reset. Then a serial transfer cycle of 8/9/16/32 clock-cycles is started leading to 8/9/16/32 bits being transmitted on SPI_DO. Simultaneously, data is received on SPI_DI and shifted into the IO buffer. The transfer cycle finishes after the 8th/9th/16th/32nd clock cycle and SPI_INT_BIT bit is set in the SPI_CTRL_REG, and SPI_INT_PEND bit in (RE)SET_INT_PENDING_REG is set. The received bits in the IO buffer are copied to the SPI_RX_TX_REG where they can be read by the CPU.

Interrupts to the CPU can be disabled using the SPI_MINT bit. To clear the SPI interrupt source, any value to SPI_CLEAR_INT_REG must be written. Take note that SPI_INT will be set if the RX-FIFO contains unread data.

33.2.2 Slave Mode

The slave mode is selected with SPI_SMn set to 1 and the Px_MODE_REG must also select SPI_CLK as input. The functionality of the IO buffer in slave and master mode is identical. The SPI module clocks data in from SPI_DI and out on SPI_DO on every active edge of SPI_CLK, as shown in [Figure 86](#). The SPI has an active low clock enable SPI_EN, which can be enabled with bit SPI_EN_CTRL=1.

In slave mode, the internal SPI clock must be four times greater than the SPI_CLK coming from the master. The SPI_EN serves as a clock enable and bit synchronization, if enabled with bit SPI_EN_CTRL. As soon as SPI_EN is deactivated between the MSB and LSB bits, the I/O buffer is reset.

33.2.3 SPI_POL and SPI_PHA

The phase and polarity of the serial clock can be changed with bits SPI_POL and SPI_PHA in the SPI_CTRL_REG.

33.2.4 SPI_DO Idle Levels

The idle level of signal SPI_DO depends on the master or slave mode, polarity and phase mode of the clock.

In master mode, pin SPI_DO gets the value of bit SPI_DO if the SPI is idle in all modes (mode 0 to 3). Also if slave is in SPI mode 0 or 2, then SPI_DO bit is the initial and final idle level.

In SPI modes 1 and 3, there is no clock edge after the sampled lsb and pin SPI_DO gets the lsb value of the IO buffer. If required, the SPI_DO can be forced to the SPI_DO bit level by resetting the SPI to the idle state. This is done by shortly setting bit SPI_RST to 1. (Optionally, SPI_FORCE_DO can be set, but this does not reset the IO buffer). The following diagrams show the timing of the SPITM interface.

33.2.5 Write Only Mode

In "write only" mode (SPI_FIFO_MODE = "10") only the TX-FIFO is used. Received data will be copied to the SPI_RX_TX_REG, but if a new SPI transfer is finished before the old data is read from the memory, this register will be overwritten.

SPI_INT acts as a tx_request signal, indicating that there is still place in the FIFO. It is '0' when the FIFO is full or '1' when it's not full. This is shown in the SPI_CTRL_REG[SPI_TXH], which is '1' if the TX-FIFO is full. Writing to the FIFO if this bit is still 1, will result in transmission of undefined data. If all data has been transferred, SPI_CTRL_REG1 [SPI_BUSY] will become '0'.

Multi-core Bluetooth 5.2 SoC family with system PMU

33.2.6 Read Only Mode

In “read only” mode (`SPI_FIFO_MODE` = “01”), only the RX-FIFO is used. Transfers will start immediately when the SPI is turned on in this mode. In transmit direction, the `SPI_DO` pin will transmit the IO buffer contents being the actual value of the `SPI_TX_REGx` (all 0’s after reset). This means that no dummy writes are needed for read only transfers.

In Slave mode, transfers only take place if the external master initiates them, but in master mode this means that transfers will continue until the RX-FIFO is full. If this happens, `SPI_CTRL_REG1[SPI_BUSY]` will become ‘0’. If exactly N words need to be read from SPI device, first read ($N - \text{fifosize} + 1$) words. Then it waits until the `SPI_BUSY` becomes ‘0’, set `SPI_FIFO_MODE` to “00” and finally reads the remaining (`fifosize + 1`) words. Here, `fifosize` is 4/2/1 words for 8/16/32 bits mode respectively.

If this is not done, more data will be read from the SPI device until the FIFO is completely filled, or the SPI is turned off.

33.2.7 Bidirectional Transfers with FIFO

If `SPI_FIFO_MODE` is “00”, both registers are used as a FIFO. `SPI_TXH` indicates that TX-FIFO is full, `SPI_INT` indicates that there is data in the RX-FIFO.

33.2.8 TXreq Mode

In case of the no FIFO mode, a `Tx_buffer available` event is generated when Tx buffer is empty. This event triggers the DMA with a `dma_tx_request` when `SPI_DMA_TXREQ_MODE` is set. Additionally, the same event generates an interrupt to the CPU when `SPI_TX_FIFO_NOFULL_MASK` is set.

33.2.9 DMA Operation Requirements

If the DMA operation is needed when the TX-FIFO is disabled (`SPI_CTRL_REG[SPI_FIFO_MODE]` = 0x3) and `SPI_CTRL_REG[SPI_DMA_TXREQ_MODE]` = 0, the CPU needs to write one byte to the `SPI_RX_TX_REG` register before the DMA takes control of the data transfer.

Additionally, the edge-sensitive requests have to be enabled in the Tx DMA channel (`DMAx_CTRL_REG[REQ_SENSE]` = 0x1). Otherwise, the DMA priorities should be programmed in favor of the Rx DMA channel, when both channels (Rx/Tx) are used.

If `SPI_CTRL_REG[SPI_PRIORITY]` is set, edge-sensitive DMA requests cannot be used, since the SPI Rx/Tx DMA requests do not de-assert until the RX-FIFO is empty and the TX-FIFO is full. So, in this mode (and `SPI_CTRL_REG[SPI_FIFO_MODE]` = 0x3), the only way to use the DMA to serve SPI in both directions, is by programming the DMA channel priorities in favor of the Rx DMA channel.

It should be noted that Tx DMA requests are not supported when:

1. `SPI_CTRL_REG[SPI_FIFO_MODE]`=0x0 or 0x2 (both FIFOs enabled or Tx FIFO only) and `SPI_CTRL_REG[SPI_WORD]`=0x2 (32-bit SPI data enabled).The Tx DMA requests can only be supported for 8-bit and 16-bit SPI data.
2. `SPI_CTRL_REG[SPI_FIFO_MODE]`=0x1 (Rx FIFO only mode).

33.2.10 The 9-Bit Mode

The 9-bit mode can be used to support 9-bit displays. This is selected with `SPI_CTRL_REG[SPI_WORD]` set to ‘11’. The value of the 9th bit is set in the `SPI_CTRL_REG1[SPI_9BIT_VAL]` and is used to determine if the next 8 bits form a command word or data word. Because the 9th bit is not part of the data, the FIFOs are still used in the 8-bit mode. The 9th bit is received but not saved because it is shifted out of the 8-bit shift register upon reception.

33.2.11 Timing Diagrams

Multi-core Bluetooth 5.2 SoC family with system PMU

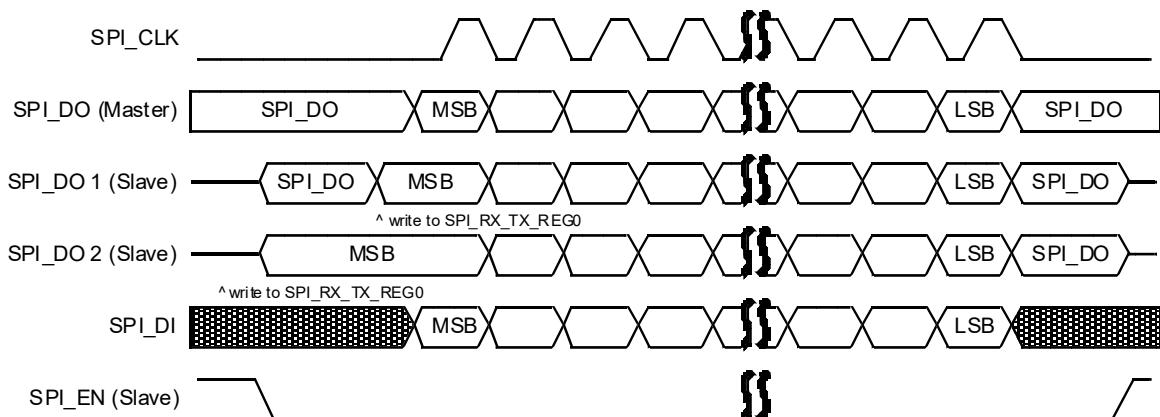


Figure 86: SPI Master/Slave, Mode 0: SPI_POL=0 and SPI_PHA=0

Note 1 If 9-bit SPI mode, the MSB bit in transmit direction is determined by bit SPI_CTRL_REG[SPI_9BIT_VAL]. In receive direction, the MSB is received but not stored.

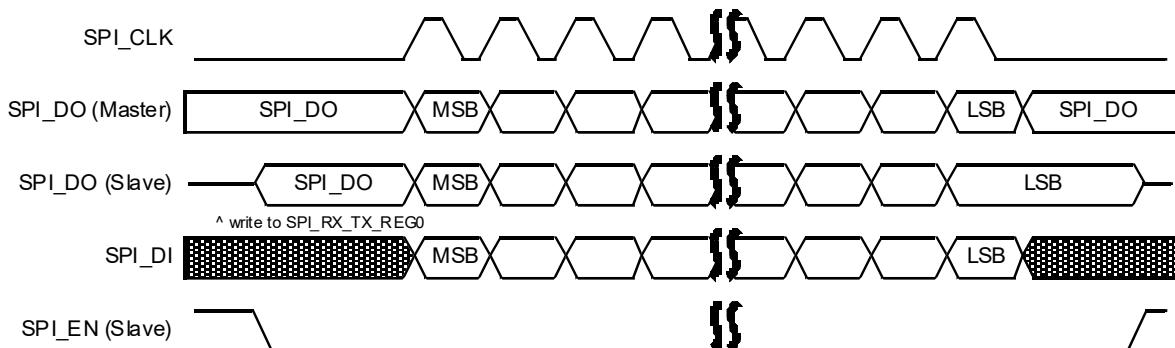


Figure 87: SPI Master/Slave, Mode 1: SPI_POL=0 and SPI_PHA=1

For the MSB bit refer to [Note 1](#).

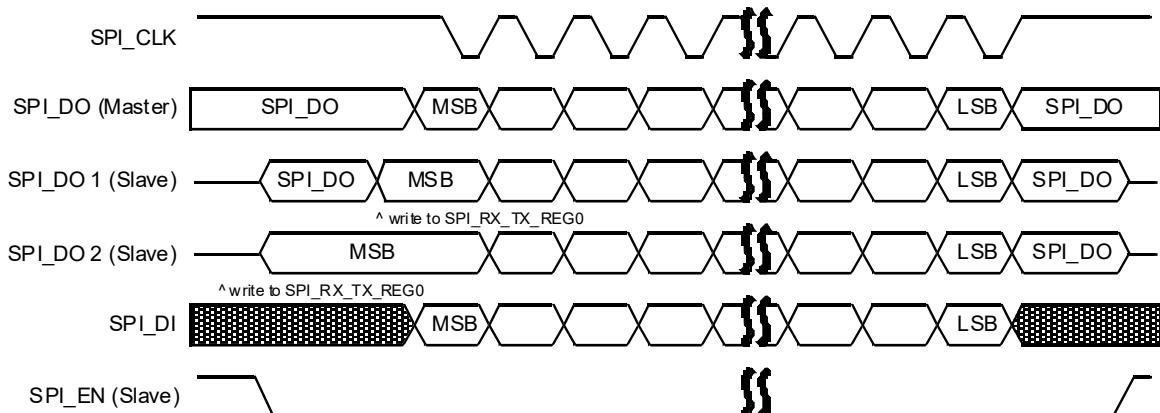


Figure 88: SPI Master/Slave, Mode 2: SPI_POL=1 and SPI_PHA=0

For the MSB bit refer to [Note 1](#).

Multi-core Bluetooth 5.2 SoC family with system PMU

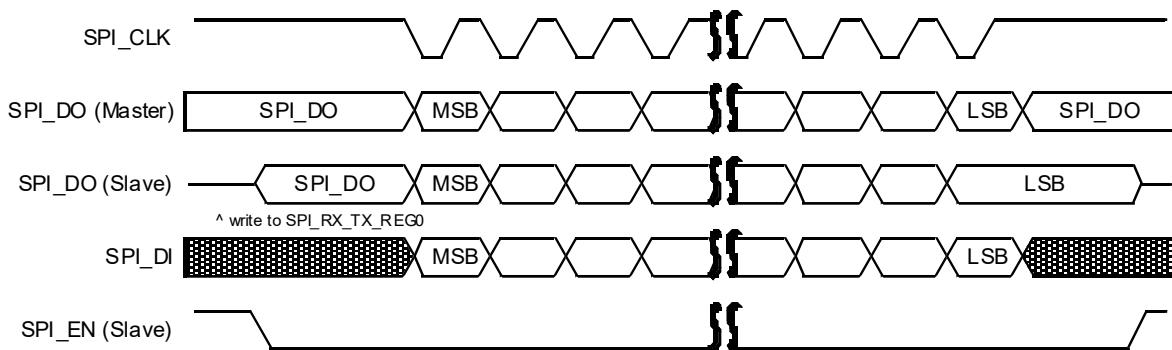


Figure 89: SPI Master/slave, Mode 3: SPI_POL=1 and SPI_PHA=1

For the MSB bit refer to [Note 1](#).

33.3 Programming

There is a simple sequence of steps that can be followed to configure and use the SPI controllers:

1. Setup the GPIOs to be used for the SPI interface ($Px_{yy_MODE_REG}[PID] = 13$ to 20).
2. Set SPI mode as master by clearing the $SPIx_CTRL_REG[SPI_SMN]$ bit or slave by setting it. In SPI slave mode, enable the SPI_EN pin ($SPIx_CTRL_REG [SPI_EN_CTRL]$).
3. Setup SPI clock frequency ($SPIx_CTRL_REG[SPI_CLK]$).
4. Select SPI clock phase ($SPIx_CTRL_REG[SPI_PHA]$).
5. Select SPI clock polarity ($SPIx_CTRL_REG[SPI_POL]$).
6. Select SPI word size ($SPIx_CTRL_REG[SPI_WORD]$).
7. Setup SPI FIFO mode if needed ($SPIx_CTRL_REG[SPI_FIFO_MODE]$).
8. Enable SPI by setting the $SPIx_CTRL_REG[SPI_ON]$ bit.
9. Assert the CS pin.
10. Write the data to $SPIx_RX_TX_REG$.
11. Poll the $SPIx_CTRL_REG[SPI_INT_BIT]$ until it becomes 1.
12. Read the data from $SPIx_RX_TX_REG$.
13. Clear SPI_INT_BIT by writing the $SPIx_CLEAR_INT_REG$ register.
14. De-assert the CS pin.

To reset/update the SPI controller configuration, a dedicated sequence should be followed:

1. Disable SPI controller by clearing the $SPIx_CTRL_REG[SPI_ON]$ bit.
2. Reset the SPI controller by setting the $SPIx_CTRL_REG[SPI_RST]$ bit.
3. Update the SPI configuration
4. Enable the SPI controller ($SPIx_CTRL_REG[SPI_ON] = 1$).
5. Release SPI reset ($SPIx_CTRL_REG[SPI_RST] = 0$).
6. Setup SPI_INT_BIT ($SPIx_CTRL_REG[SPI_MINT]$).

34 Real Time Clock

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

34.1 Introduction

The DA1469x is equipped with a Real Time Clock (RTC) which provides 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
- 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 resolution
- Event interrupt on any calendar or time unit
- Available during sleep
- Granularity of 10 ms

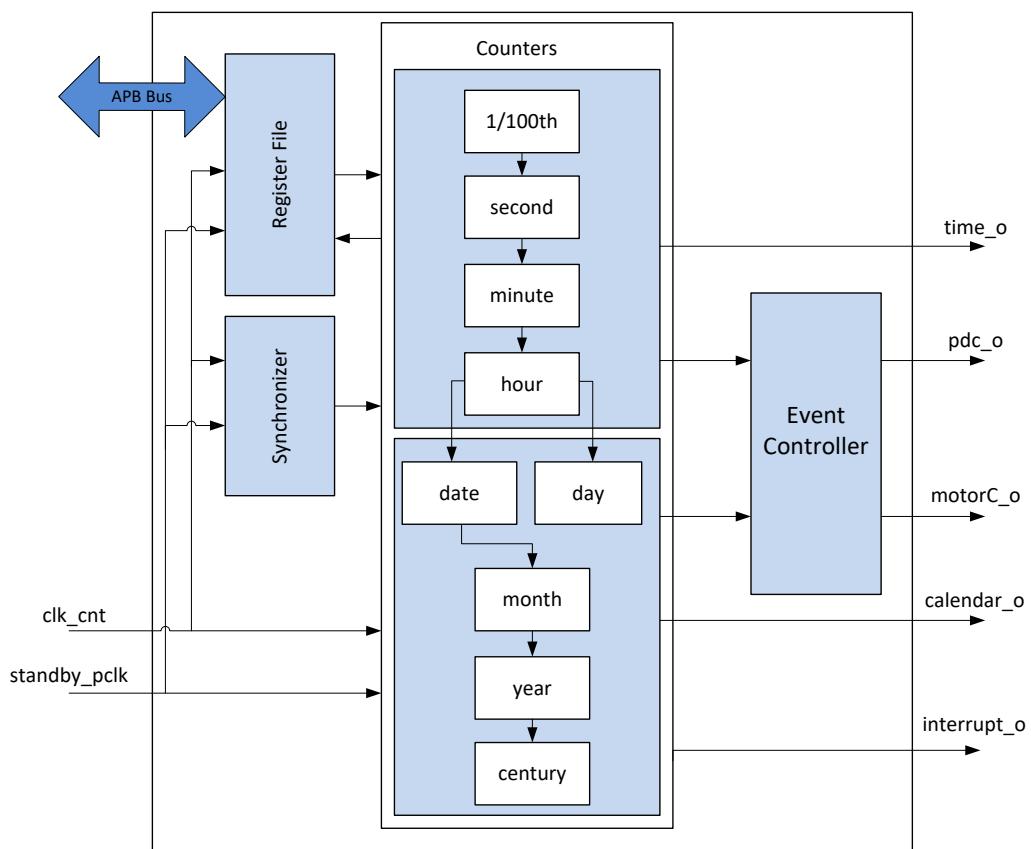


Figure 90: Real Time Clock Block Diagram

34.2 Architecture

The architecture of the Real Time Clock is depicted in Figure 90.

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).

Multi-core Bluetooth 5.2 SoC family with system PMU

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 RTC block has been enhanced with the RTC_EVENT_CONTROLLER.

This controller is triggered every 10 ms or when there is a roll-over. It comprises a 12-bit counter which counts down starting from a configurable value (programmed in RTC_MOTOR_EVENT_PERIOD). This operation can be enabled/disabled by RTC_MOTOR_EVENT_EN.

Every time the counter reaches 0, an event signal is asserted, generating a pulse to the Motor Controller. This signal is used as a trigger for the motor controller operation.

The RTC_EVENT_CONTROLLER also comprises a 13-bit counter which counts down starting from a configurable value (programmed in RTC_PDC_EVENT_PERIOD). This operation can be enabled/disabled by RTC_PDC_EVENT_EN.

Every time this counter reaches 0, a signal towards the PDC is asserted. The same signal is also driven to the Cortex-M33 RTC_EVENT interrupt line. The signal is automatically de-asserted after the RTC_PDC_EVENT_CLEAR_REG is read.

Both counters are accessible by the Cortex-M33 CPU (RTC_PDC_EVENT_CNT_REG and RTC_MOTOR_EVENT_CNT_REG).

34.3 Programming

There is a simple sequence of steps that needs to be followed 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).
This value 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).
This value should be equal to the fractional divisor part of the formula $F_{LP_CLK}/100 = 327.680$.
 - c. To achieve 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 or 24 hours based mode (RTC_HOUR_MODE_REG[RTC_HMS]).
5. Configure the time (RTC_TIME_REG).
6. Configure the date (RTC_CALENDAR_REG).
7. Setup a time alarm if needed (RTC_ALARM_ENABLE_REG).
8. Setup 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. Choose if SW reset resets the RTC (RTC_KEEP_RTC_REG[RTC_KEEP]).
13. Setup the Event Controller, if needed:

Multi-core Bluetooth 5.2 SoC family with system PMU

- a. If trigger to PDC is enable:
 - i. Setup the event period (RTC_PDC_EVENT_PERIOD_REG[RTC_PDC_EVENT_PERIOD]).
 - ii. Enable the PDC to trigger (RTC_EVENT_CTRL_REG[RTC_PDC_EVENT_EN]).
- b. If trigger to Motor Controller is enable:
 - i. Setup the event period (RTC_MOTOR_EVENT_PERIOD_REG[RTC_MOTOR_EVENT_PERIOD]).
 - ii. Enable the Motor Controller to trigger (RTC_EVENT_CTRL_REG[RTC_MOTOR_EVENT_EN]).

35 General Purpose Timers

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

35.1 Introduction

The Timers block contains four identical 24-bit wide timers (namely, Timer, Timer2, Timer3, and Timer4) that are software controlled, programmable, and can be used for various tasks. Timer and Timer2 are in the timers' power domain (PD_TMR), while Timer3 and Timer4 are in the system power domain (PD_SYS).

Features

- Four 24-bit general purpose timers
- Pulse Width Modulated signal (PWM)
- Two channels for capture input triggered by GPIOs (Timer has four channels)
- One shot pulse with programmable pulse width (only valid for Timer and Timer2)
- 5-bit clock pre-scaler
- Selectable system or sleep clock source
- Up/down counting capability with free running mode
- Active while system is in sleep mode
- Dedicated interrupt line per timer (Timer supports one extra interrupt line for complex capture events)

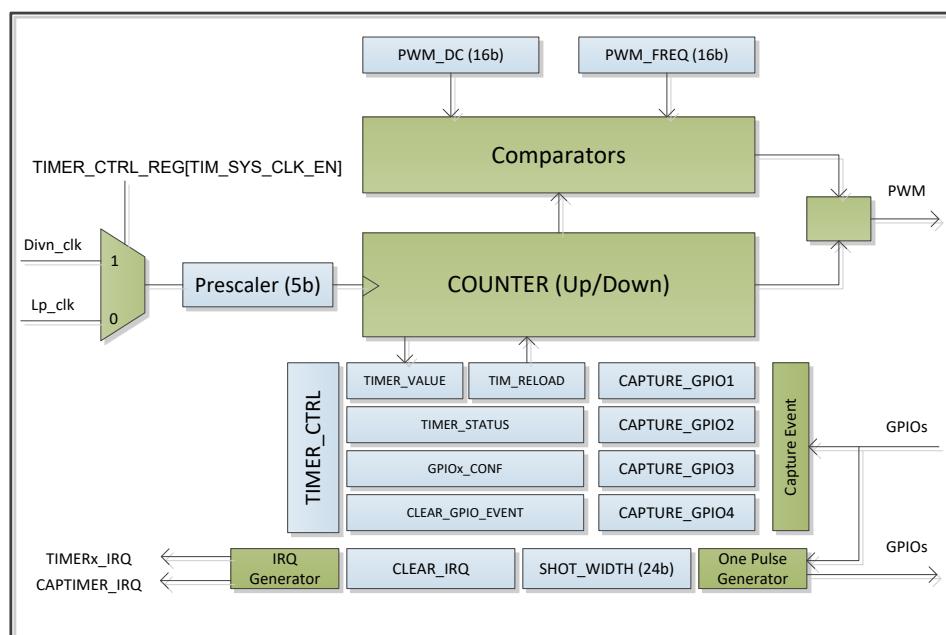


Figure 91: General Purpose Timer's Block Diagram

35.2 Architecture

There are four instances of the same Timer block (Timer, Timer2, Timer3, and Timer4) in the system with minor modifications between the instances. The Timer's block diagram is shown in [Figure 91](#). The only differences between the actual instances are the following:

- Timer and Timer2 are part of the PD_TMR domain, while Timer3 and Timer 4 are part of the PD_SYS power domain

Multi-core Bluetooth 5.2 SoC family with system PMU

- Timer3 and Timer4 do not support the one-shot feature
- Timer supports four channels (GPIOs) for capturing events, while the rest supports two channels. Timer also has a separate interrupt line dedicated to these channel activities
- Timer and Timer2 can drive specific GPIOs, even when the system is in sleep mode

In general, each block has a configurable free-running up/down counter that operates either on the system clock (sys_clk) or the low power clock (lp_clk) with a pre-scaler of 5 bits. This results in a minimum frequency of one kHz and a maximum of 32 MHz.

Each block has a PWM generator which does not affect the actual timer running. The PWM generator can define the frequency and the duty cycle of the generated PWM signal by further dividing the clock by maximum ($2^{16}+1$), resulting in a minimum speed of 0.015 kHz. A 16-bit PWM duty cycle enables 65535 steps for defining the pulse width. Note that for the minimum PWM frequency only 50 % duty cycle is possible. Also, note that the values in the respective registers for the duty cycle and the frequency represent clock counts. Duty cycle value needs to be always less than frequency to the PWM generator to work correctly.

Capturing an edge of a GPIO is another feature of the Timer block. As a matter of fact, any edge on any (configurable) GPIO can be sensed to trigger a Timer's snapshot stored into a separate register. Another (or the same) GPIO can be configured for capturing a second edge, hence storing a second snapshot into a second register. The difference of these two registers indicates the time distance of the two triggering events, hence it can be used for measuring the frequency of a signal or the timing interval between two interrupts coming from external devices.

One-shot is also supported. Upon a trigger configured as an input on a GPIO, another GPIO serves as an output, delivering a pulse of configurable width. This is basically a PWM reply in hardware.

There are specific GPIOs that can be configured to be input events to the Timer blocks. The actual configuration is presented in [Table 143](#):

Table 143: Timer's Input GPIOs

| Timer Input | Value | GPIO |
|-----------------------|-------|---|
| TIMER_GPIOx_CONF_REG | 0-32 | 0: Disabled, 1-32: P0_0 to P0_31 |
| TIMER2_GPIOx_CONF_REG | 0-32 | 0: Disabled, 1-23: P1_0 to P1_22, 24-32: P0_23 to P0_31 |
| TIMER3_GPIOx_CONF_REG | 0-32 | 0: Disabled, 1-32: P0_0 to P0_31 |
| TIMER4_GPIOx_CONF_REG | 0-32 | 0: Disabled, 1-23: P1_0 to P1_22, 24-32: P0_23 to P0_31 |

Note that Timer and Timer2 are able to output PWM signals available during sleep (CLK_TMR_REG[TMR_PWM2_AON_MODE]) on P1_01 and P1_06 accordingly.

35.3 Programming

There is a simple sequence of steps that needs to be followed to program the GP Timers:

1. Disable the Timer by clearing the TIMERx_CTRL_REG[TIM_EN] bit and the TIMERx_CTRL_REG[TIM_CLK_EN] bit.
2. Select Timer's clock (TIMER_CTRL_REG[TIM_SYS_CLK_EN]) and pre-scaler (TIMER_PRESCALER_REG).
3. Enable Timer's clock (TIMERx_CTRL_REG[TIM_CLK_EN]).
4. Select Timer's mode (TIMER_CTRL_REG[TIM_ONESHOT_MODE_EN]).
 - a. One shot mode (TIMER/TIMER2):
 - i. Set delay phase (TIMER/2_RELOAD_REG).
 - ii. Set shot phase duration (TIMER/2_SHOTWIDTH_REG).
 - iii. Select one shot/primary capture event input (TIMER/2_GPIO1_CONF_REG) and trigger polarity (TIMER/2_CTRL_REG[TIM_IN1_EVENT_FALL_EN]).
 - iv. Select the secondary capture event input (TIMER/2_GPIO2_CONF_REG) and trigger polarity (TIMER/2_CTRL_REG[TIM_IN2_EVENT_FALL_EN]).

Multi-core Bluetooth 5.2 SoC family with system PMU

- b. Counter mode (all Timers):
 - i. Set timer's up/down direction (TIMER_CTRL_REG[TIM_COUNT_DOWN_EN]).
 - ii. Set timer's reload or max value (TIMERx_RELOAD_REG).
 - iii. Enable free run mode if timer counts upwards by setting the TIMERx_CTRL_REG[TIM_FREE_RUN_MODE_EN] bit.
 - iv. Set up capture events input (TIMERx_GPIOy_CONF_REG) and trigger polarity (TIMERx_CTRL_REG[TIM_INx_EVENT_FALL_EN]). Note that TIMER supports up to four capture events inputs.
5. Configure PWM:
 - a. Set up frequency (TIMERx_PWM_FREQ_REG).
 - b. Configure duty cycle (TIMERx_PWM_DC_REG).
6. Select if event on the selected input GPIOs creates a CAPTIM interrupt (TIMERx_CTRL_REG[TIM_CAP_GPIOy_IRQ_EN]).
7. Set up GPIO as Timer output (Px_yy_MODE_REG[PID] = 3, Px_yy_MODE_REG[PID] = 49 to 54, see P0_00_MODE_REG description).
Note that TIMER and TIMER2 PWM outputs can be activated at specific pins (P1_01 and P1_06) during sleep by setting the CLK_TMR_REG[TMRx_PWM_AON_MODE] bits.
8. Enable Timer interrupt by setting the TIMERx_CTRL_REG[TIM IRQ_EN] bit.
9. Enable Timer by setting the TIMERx_CTRL_REG[TIM_EN] bit.

36 Watchdog Timers

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

36.1 Introduction

The DA1469x contains two watchdog timers:

- The System watchdog that is in the always ON power domain
- The CMAC watchdog that is in the radio power domain

They protect the system from getting stuck because of SW problems both in the application processor (Cortex-M33) and in the CMAC area (Cortex-M0+). They are clocked by internally generated clocks, RC32K or RCX, and the low power clock, XTAL32K or RCX, respectively.

CMAC watchdog is accessible by the Cortex-M33 if the radio power domain is powered up.

Features

- System Watchdog
 - A 13 bits down counter running on either RC32K or RCX further divides these clocks by 320 and can operate for 84 seconds or three minutes, depending on the clock
 - Internal programmable clock sources are RC32K or RCX, of which RC32K is the default. If RCX is used as a sleep clock, RC32K might be turned off
 - Generates:
 - NMI to Cortex-M33 if counter reaches 0
 - HW reset if counter reaches -16
 - Protected by a lock bit to avoid freeze by mistake
 - Automatically frozen when Cortex-M33 is in debug mode
- CMAC Watchdog
 - A 13 bits down counter running on the lp_clk further divides this clock by 32
 - Generates:
 - Early notification interrupt to Cortex-M0+ if counter reaches 16
 - CMAC2SYS_IRQ to Cortex-M33 if counter reaches 0
 - HW reset if counter reaches -16
 - Protected by a lock bit to avoid freeze by mistake
 - Automatically frozen when Cortex-M0+ is in debug mode

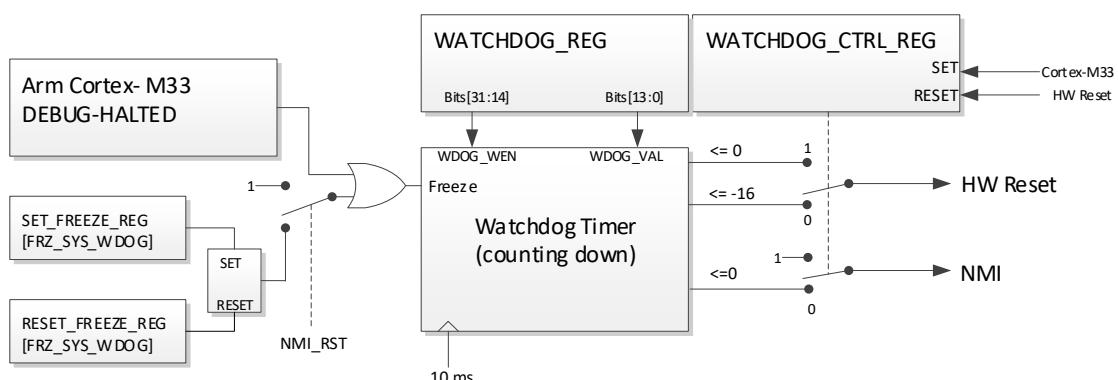


Figure 92: System Watchdog Block Diagram

36.2 Architecture

The System Watchdog is supplied by the always on domain (PD_AON) and is automatically enabled as soon as the system powers up. It is decremented by one every 10 ms, assuming the default source clock is the RC32K. The timer value can be accessed through the WATCHDOG_REG which is set to max value at reset. This results in a maximum watchdog time-out of 87 seconds. If the RCX is used as a source clock, the time-out time is even longer, depending on the RCX frequency.

During write access WATCHDOG_REG bits [31-14] must be 0. This provides extra filtering for a software run-away writing all ones to the WATCHDOG_REG. If the watchdog reaches 0, the counter value will get a negative value setting bit 8. The counter sequence is 1, 0, 1FFF16 (-1), 1FFE16(-2),...,1FF016 (-16).

If WATCHDOG_CTRL_REG[NMI_RST] = 0, the watchdog timer will generate an NMI to the Cortex-M33 when the watchdog timer reaches 0 and a HW reset when the counter becomes less or equal to -16. The NMI handler must write any value > -16 to the WATCHDOG_REG to prevent the generation of a WDOG reset within $16 \times 10 = 160$ ms.

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

The system watchdog can be frozen by Cortex-M33. It is always frozen automatically when the debugger is attached and the Cortex-M33 CPU is halted during debugging. However, even if the watchdog is frozen by Cortex-M33, when Cortex-M33 gets into any sleep mode (PD_SYS is turned off), the system watchdog will be automatically resumed to operate during sleep.

The CMAC Watchdog is basically the same circuit with the following amendments:

- It resides in the radio power domain (PD_RAD). It will be active as soon as the radio power domain is enabled
- Its clock source is the low power clock (lp_clk). It is further divided by 32 to provide a clock period of 1 ms (assuming lp_clk is XTAL32K). Hence, this watchdog can reach maximum 16 ms before reaching 0
- It has one extra notification compared to the system watchdog:
 - When 16 is reached, an interrupt will be issued to the Cortex-M0+ (CMAC)
 - When 0 is reached, an interrupt will be issued to the Cortex-M33
 - When -16 is reached, a HW reset will be triggered

This watchdog is also automatically halted when the debugger is attached and the Cortex-M0+ is halted during debugging.

36.3 Programming

36.3.1 System Watchdog

There is a simple sequence of steps that needs to be followed to program the System Watchdog Timer:

1. Switch to RCX clock as source if needed (CLK_RCX_REG[RCX_ENABLE]).
2. Freeze watchdog by setting the SET_FREEZE_REG[FRZ_SYS_WDOG] bit (optionally).
3. Select NMI and reset events (WATCHDOG_CTRL_REG[NMI_RST]).
4. Wait until WATCHDOG_CTRL_REG[WRITE_BUSY] = 0.
5. Enable writing of the watchdog timer (WATCHDOG_REG[WDOG_WEN] = 0).
6. Write watchdog timer reload value (WATCHDOG_REG[WDOG_VAL], see register description).
7. Resume watchdog (RESET_FREEZE_REG[FRZ_SYS_WDOG] = 1), if frozen.

36.3.2 CMAC Watchdog

There is no specific sequence of steps for the CMAC Watchdog since it gets automatically enabled. Some general points are referred below:

- The CMAC watchdog is automatically enabled when the radio power domain is activated.
- CMAC takes care of reloading the watchdog timer frequently. M33 intervention is not necessary.
- M33 can freeze/unfreeze the CMAC watchdog (SET_FREEZE_REG[FRZ_SYS_WDOG]) and reload the watchdog timer (CM_WDOG_REG[CM_WDOG_CNT]), if needed.
- CMAC watchdog automatically freezes when Cortex-M0+ is halted via the SWD.
- When the system power domain is powered down, it's not possible to freeze CMAC watchdog via SW.

37 USB Controller

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

37.1 Introduction

The USB interface is an integrated USB Node controller compatible with the full speed (FS) and low speed (LS) *USB specification, version 1.1* (V1.1).

It integrates a Serial Interface Engine (SIE) and USB endpoint (EP) FIFOs. Seven endpoint pipes are supported: one for the mandatory control endpoint and the other six for interrupt endpoints. Each endpoint pipe has a dedicated FIFO, 8 bytes for the control endpoint and 64 bytes for the other endpoints.

The USB transceiver module is accessed through USB_Dp and USB_Dm pins.

Features

- Full Speed/Low Speed USB node
- Interfaces to USB V1.1 transceiver with programmable rise and fall times and integrated D+/D- pull-up resistors
- Serial Interface Engine (SIE) consisting of a Media Access Controller (MAC), *USB Specification 1.0 and 1.1* compliant
- USB Function Controller with seven FIFO-based Endpoints:
 - One bidirectional Control Endpoint 0 (8 bytes)
 - Three Transmit Endpoints (64 bytes each)
 - Three Receive Endpoints (64 bytes each)
- Automatic Data PID toggling/checking and NAK packet recovery (maximum 256x32 bytes of data = 8 kB)

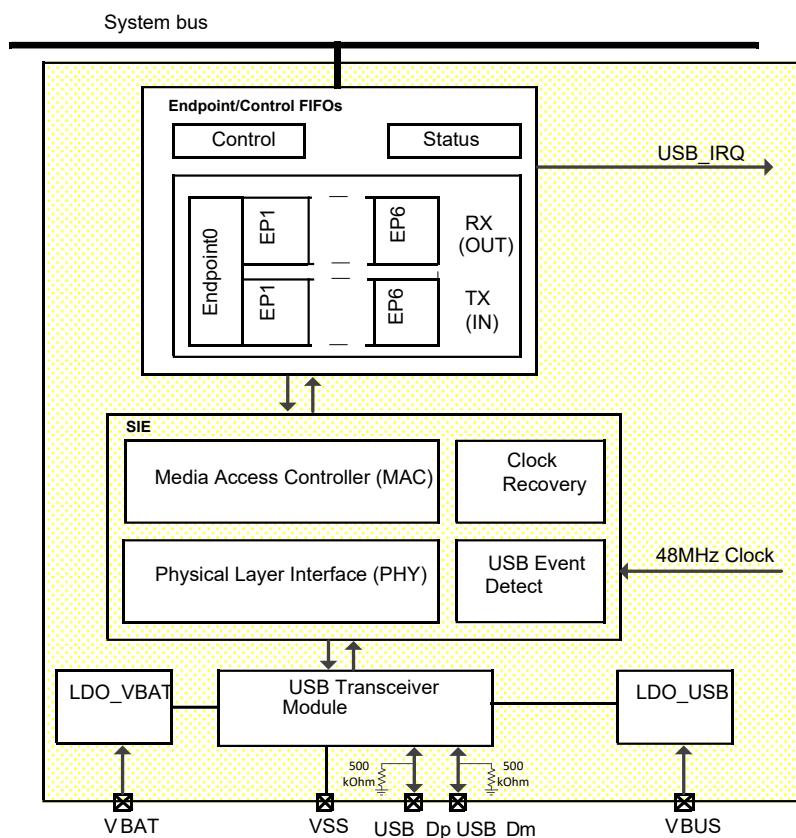


Figure 93: USB Node Block Diagram

37.2 Architecture

37.2.1 Serial Interface Engine

The SIE is comprised of physical (PHY) and Media Access Controller (MAC) modules. The PHY module includes the digital-clock recovery circuit, a digital glitch filter, End Of Packet (EOP) detection circuitry, and bit stuffing and unstuffing logic. The MAC module includes packet formatting, CRC generation and checking, and endpoint address detection. It provides the necessary control to give the NAK, ACK, and STALL responses determined by the Endpoint Pipe Controller (EPC) for the specified endpoint pipe. The SIE is also responsible for detecting and reporting USB-specific events, such as NodeReset, NodeSuspend, and NodeResume. The module output signals to the transceiver are well matched (under 1 ns) to minimize skew on the USB signals.

The USB specifications use bit stuffing and unstuffing as the method to ensure adequate electrical transitions on the line to enable clock recovery at the receiving end. The bit stuffing block ensures that whenever a string of consecutive 1's is encountered, a 0 is inserted after every sixth 1 in the data stream. The bit unstuffing logic reverses this process.

The clock recovery block uses the incoming NRZI data to extract a data clock (12 MHz for FS, 1.5 MHz for LS) from a 48 MHz (FS)/6 MHz (LS) input clock. The extracted data clock is used in the data recovery circuit. The output of this block is binary data which is decoded from the NRZI stream and can be appropriately sampled using the extracted 12(1.5) MHz clock. The jitter performance and timing characteristics meet the requirements in *Chapter 7* of the *USB Specification*.

37.2.2 Endpoint Pipe Controller (EPC)

The EPC provides the interface for USB function endpoints. An endpoint is the ultimate source or sink of data. An endpoint pipe facilitates the movement of data between USB and memory and completes the path between the USB host and the function endpoint. According to the *USB specification*, up to 31 such endpoints are supported at any given time. USB allows a total of 16

Multi-core Bluetooth 5.2 SoC family with system PMU

unidirectional endpoints for receive and 16 for transmit. As the control endpoint 0 is always bidirectional, the total number is 31. The FS/LS USB node supports a maximum of seven endpoint pipes with the same function address. See [Figure 94](#) for a schematic diagram of the EPC operation.

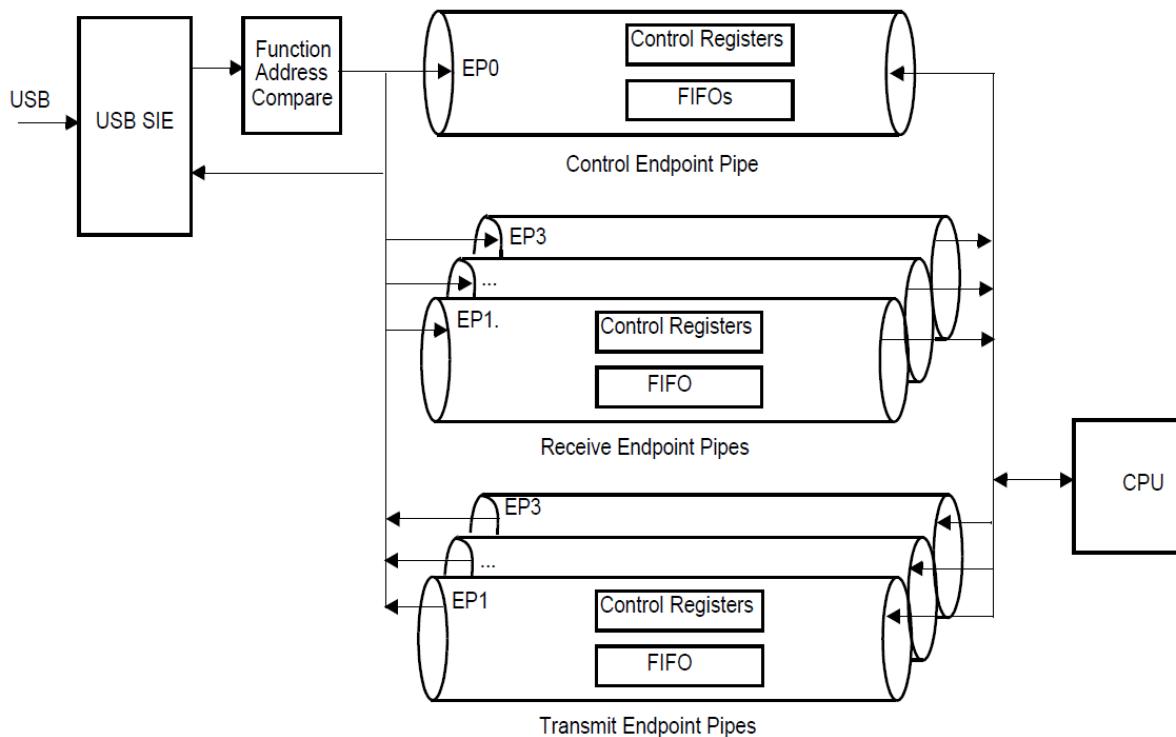


Figure 94: Endpoint Operation

A USB function is a USB device that can transmit and receive information on the bus. A function may have one or more configurations, each of which defines the interfaces that make up the device. Each interface, in turn, is composed of one or more endpoints.

Each endpoint is an addressable entity on USB and is required to respond to IN and OUT tokens from the USB host (typically a PC). IN tokens indicate that the host has requested to receive information from an endpoint, and OUT tokens indicate that it is about to send information to an endpoint.

On detection of an IN token addressed to an endpoint, the endpoint pipe should respond with a data packet. If the endpoint pipe is currently stalled, a STALL handshake packet is sent under software control. If the endpoint pipe is enabled but no data is present, a Negative Acknowledgment (NAK) handshake packet is sent automatically. If the endpoint pipe is isochronous and enabled but no data is present, a bit stuff error followed by an “end of packet” is sent on the bus.

Similarly, on detection of an OUT token addressed to an endpoint, the endpoint pipe should receive a data packet sent by the host and load it into the appropriate FIFO. If the endpoint pipe is stalled, a STALL handshake packet is sent. If the endpoint pipe is enabled but no buffer is present for data storage, a NAK handshake packet is sent.

A disabled endpoint does not respond to IN, OUT, or SETUP tokens.

The EPC maintains separate status and control information for each endpoint pipe.

For IN tokens, the EPC transfers data from the associated FIFO to the host. For OUT tokens, the EPC transfers data in the opposite direction.

37.2.3 Functional States

37.2.3.1 Line Condition Detection

At any given time, the USB node is in one of the following states:

Multi-core Bluetooth 5.2 SoC family with system PMU

- NodeOperational: this is the normal operating state of the node. In this state, the node is configured for operation on the USB
- NodeSuspend: Device operation suspended due to USB inactivity
 - A USB node is expected to enter NodeSuspend state when 3 ms have elapsed and no bus activities have been detected
 - The USB node looks for this event and signals it by setting the SD3 bit in the USB_ALTEV register, which causes an USB_INT, if enabled, to be generated. The firmware should respond by putting the USB node in NodeSuspend state
- NodeResume: Device wake-up from the suspended state
 - In NodeResume state, a constant “K” is signaled on the USB and should last at least 1 ms and no more than 15 ms. After that, the USB host should continue sending the NodeResume signal for at least another 20 ms, and then completes the NodeResume operation by issuing the End Of Packet (EOP) sequence
- NodeReset: Device reset
 - When detecting a NodeResume or NodeReset signal while in NodeSuspend state, the USB node can signal it to the CPU by generating an interrupt

The NodeSuspend, NodeResume, or NodeReset line condition causes a transition from one operating state to another. These conditions are detected by specialized hardware and reported via the Alternate Event (ALTEV) register. If interrupts are enabled, an interrupt is generated upon the occurrence of any of the specified conditions.

Refer to section [37.2.11](#) on how to obtain the lowest power consumption in the NodeSuspend state.

The USB node can resume the normal operation in two ways:

- Host initiated. When a resume signal followed by LS EOP on the USB is detected, an ALTEV[RESUME] interrupt is generated. The firmware responds by setting the NodeOperational in the USB_NFRS_REG (see also [37.2.11](#)).
- Device initiated. When a local event is detected, for example, a GPIO key is pressed, a KEYB_INT is generated. The firmware releases the USB node from the NodeSuspend state by initiating a NodeResume signal on the USB using the NFSR register. The node firmware must ensure at least 5 ms of Idle on the USB by checking the SD5 in the USB_ALTEV before going to the NodeResume state.

To successfully detect the EOP, the firmware must respond by setting NodeOperational in the USB_NFRS_REG. Once an EOP is detected, the USB_ALTEV_REG[EOP] is set. If no EOP is received from the host within 100 ms, the software must re-initiate NodeResume.

USB specifications require that a device must be ready to respond to USB tokens within 10 ms after wake-up or reset.

37.2.4 Functional State Diagram

[Figure 95](#) shows the device states and transitions as well as the conditions that trigger each transition. All FS/LS USB node state transitions are initiated by the firmware.

Multi-core Bluetooth 5.2 SoC family with system PMU

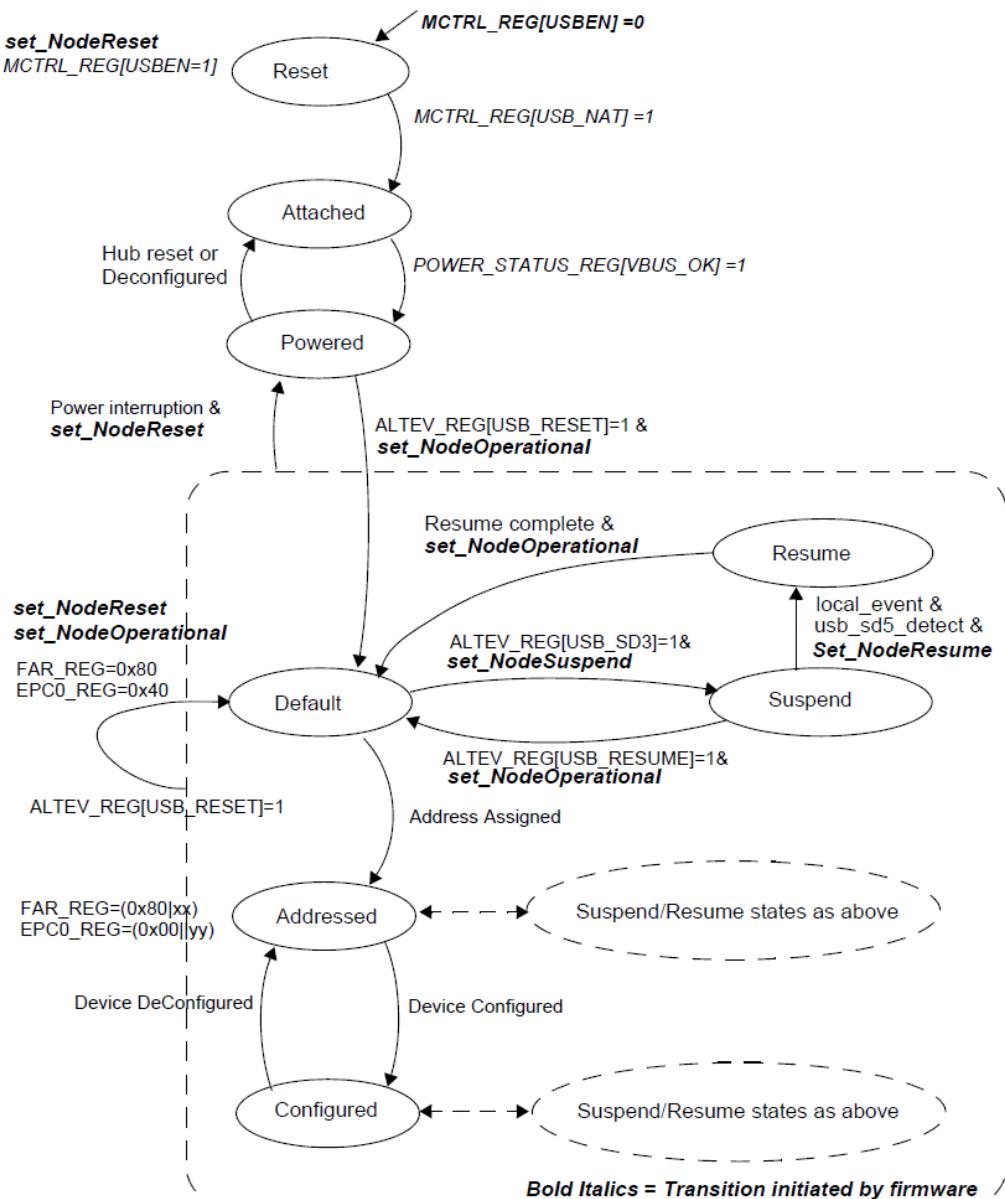


Figure 95: Node Functional State Diagram

- Note 1** When the node is not in NodeOperational state, all registers are frozen except for the endpoint controller state machines and the TX_EN, LAST, and RX_EN bits which are reset.
- Note 2** In NodeResume state, resume signaling is propagated upstream.
- Note 3** In NodeSuspend state, the node may enter a low power state and is able to detect resume signaling.

Table 144: Functional States

| State Transition | Condition Asserted |
|-------------------|---|
| set_NodeReset | Node Functional State register NFS[1:0] bits are written with 00 _b The firmware should only initiate set_NodeReset if RESET in the ALTEV register is set. |
| set_NodeSuspend | Node Functional State register NFS[1:0] bits are written with 11 _b The firmware should only initiate set_suspend if SD3 in the ALTEV register is set. |
| set_NodeOperation | Node Functional State register NFS[1:0] bits are written with 10 _b |
| set_NodeResume | Node Functional State register NFS[1:0] bits are written with 01 _b The firmware should only initiate clear_suspend if SD5 in the ALTEV register is set. |

| State Transition | Condition Asserted |
|--------------------|---|
| usb_reset_detect | USB_RESET in the ALTEV register is set to 1. |
| local_event | A local event that should wake up the USB. |
| usb_sd5_detect | USB_SD5 in the ALTEV register is set to 1. |
| usb_suspend_detect | USB_SD3 in the ALTEV register is set to 1. |
| usb_resume_detect | RESUME in the ALTEV register is set to 1. |
| resume_complete | The node should stay in NodeResume state for at least 10 ms and then must enter NodeOperational state to detect the EOP from the host, which terminates this Remote Resume operation. EOP is signaled when EOP in the ALTEV register is set to 1. |

37.2.5 Address Detection

Packets are broadcasted from the host controller to all the nodes on the USB network. Address detection is implemented in hardware to allow selective reception of packets and to permit optimal use of microcontroller bandwidth. One function address with seven different endpoint combinations is decoded in parallel. If a match is found, that particular packet is received into the FIFO; otherwise, it is ignored.

Figure 96 shows the block diagram of the function address and endpoint decoding. The incoming USB Token, Packet Address field, and four bits Endpoint field are extracted from the incoming bit stream. The address field is compared to the Function Address register (FADR) and if a match is detected, the USB Endpoint field is compared to all EP bit fields in the Endpoint Control registers (EPCx). The transmit Endpoint Control registers are compared with IN tokens and the receive Endpoint Control registers are compared with OUT tokens. A match then enables the respective endpoint FIFO and transfers the payload data to/from the FIFO. Note that EPC0 is bidirectional and is enabled for IN, OUT, and SETUP tokens.

Multi-core Bluetooth 5.2 SoC family with system PMU

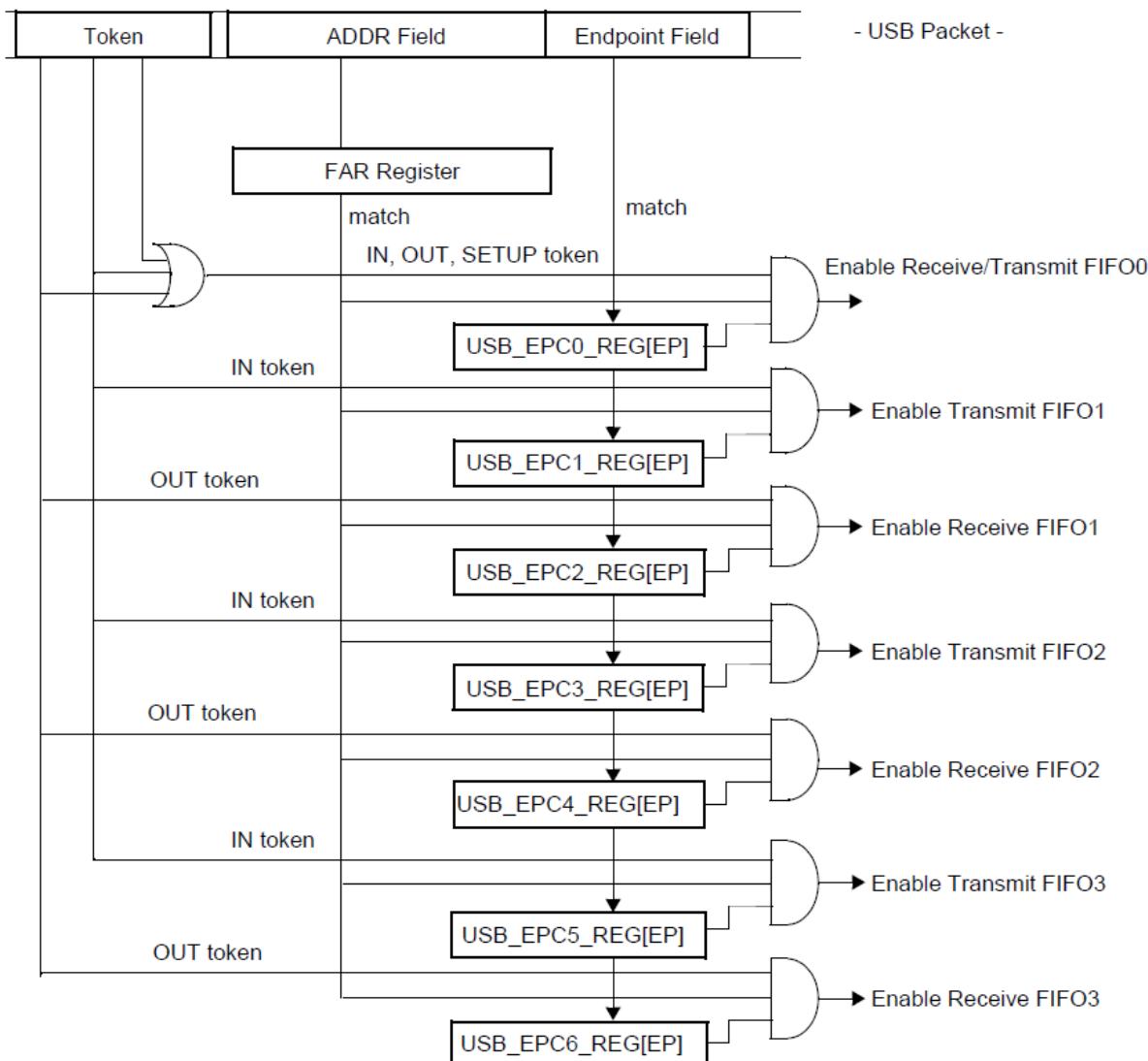


Figure 96: USB Function Address/Endpoint Decoding

37.2.6 Transmit and Receive Endpoint FIFOs

The FS/LS USB node uses a total of seven transmit and receive FIFOs: one bidirectional transmit and receive FIFO for the mandatory control endpoint, three transmit FIFOs, and three receive FIFOs. As shown in [Table 145](#), the bidirectional FIFO for the control endpoint is 8 bytes deep. The additional unidirectional FIFOs are of 64 bytes each for both transmit and receive. Each FIFO can be programmed for one exclusive USB endpoint, used together with one globally decoded USB function address. The firmware must not enable both transmit and receive FIFOs for endpoint zero at any given time.

Table 145: USB Node Endpoint Sizes

| Endpoint No. | TX FIFO | | RX FIFO | |
|--------------|--------------|--------------|--------------|---------------|
| | Size (Bytes) | Name | Size (Bytes) | Name |
| 0 | | | 8 FIFO0 | |
| 1 | 64 | TXFIFO1 (IN) | | |
| 2 | | | 64 | RXFIFO1 (OUT) |
| 3 | 64 | TXFIFO2 (IN) | | |

| Endpoint No. | TX FIFO | | RX FIFO | |
|--------------|---------|--------------|---------|---------------|
| 4 | | | 64 | RXFIFO2 (OUT) |
| 5 | 64 | TXFIFO3 (IN) | | |
| 6 | | | 64 | RXFIFO3 (OUT) |

If two endpoints in the same direction are programmed with the same endpoint number [EP field] and both are enabled, data is received or transmitted to/from the endpoint with the lower number until that endpoint is disabled for bulk or interrupt transfers, or becomes full or empty for ISO transfers. For example, if receive EP1 and receive EP2 both use endpoint 3 and are both isochronous, the first OUT packet is received into EP1 and the second OUT packet into EP2, assuming no firmware interaction in between. For ISO endpoints, this allows implementing a ping-pong buffer scheme together with the frame number match logic. Endpoints in different directions programmed with the same endpoint number operate independently.

37.2.7 Bidirectional Control Endpoint FIFO0

FIFO0 should be used for the bidirectional control endpoint zero. It can be configured to receive data sent to the default address with the DEF bit in the EPC0 register.

The Endpoint 0 FIFO can hold a single receive or transmit packet with up to 8 bytes of data.

| NOTE |
|--|
| A packet written to the FIFO is transmitted if an IN token for the respective endpoint is received. If an error condition is detected, the packet data remains in the FIFO and the transmission is retried with the next IN token. |

The FIFO contents can be flushed to allow a response to an OUT token or to write new data into the FIFO for the next IN token.

Figure 97 shows the Endpoint 0 state machine. In state TXWAIT, if `USB_RXC0_REG[SETUP_FIX]` = 0, no state change will take place if a SETUP is received. With `SETUP_FIX` = 1, the state machine goes to IDLE, flushes to EP0, and receives the token in the RXWAIT state. If a SETUP is received in states TXFILL or RXDRAIN and `SETUP_FIX` = 1, the SETUP will be ignored and no ACK is sent. This allows undisturbed FIFO filling/emptying. This state is usually present for a very short time and forces the host to retransmit the SETUP once. If an OUT token is received for the FIFO, the firmware is informed that the FIFO has received data, only if there was no error condition (CRC or STUFF error). Erroneous receptions are automatically discarded.

Multi-core Bluetooth 5.2 SoC family with system PMU

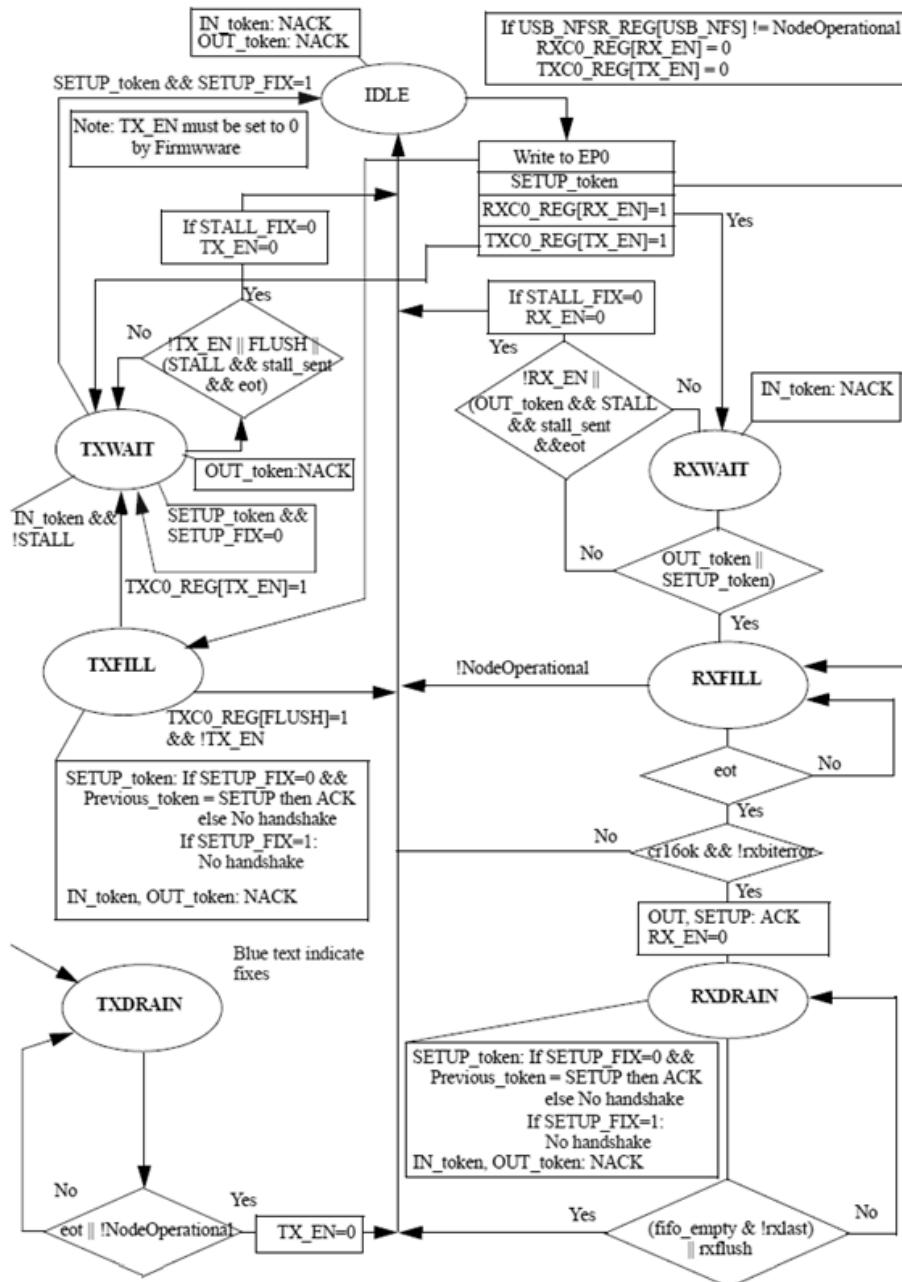


Figure 97: Endpoint 0 Operation

37.2.8 Transmit Endpoint FIFO

The Transmit FIFOs for Endpoints 1, 3, and 5 support bulk and interrupt USB packet transfers larger than the actual FIFO size. Therefore, the firmware must update the FIFO contents while the USB packet is transmitted on the bus.

Figure 98 illustrates the operation of the transmit FIFOs.

Multi-core Bluetooth 5.2 SoC family with system PMU

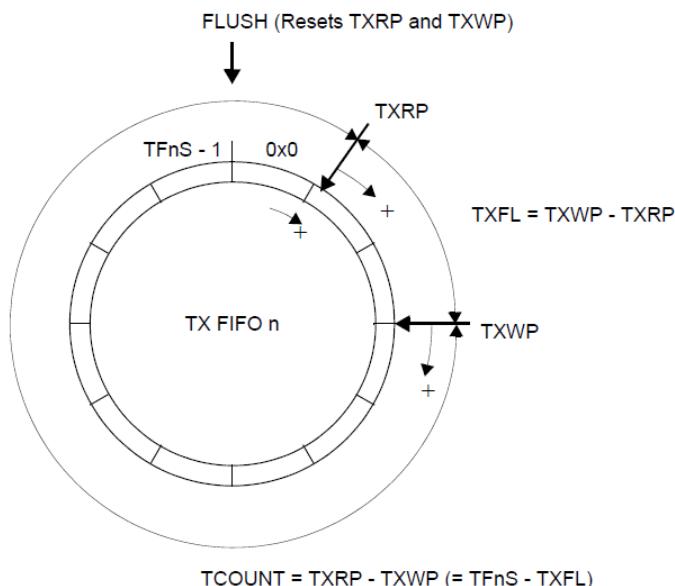


Figure 98: USB Tx FIFO Operation

- **TFnS: Transmit FIFO n Size**

This is the total number of bytes available within the FIFO.

- **TXRP: Transmit Read Pointer**

This pointer is incremented every time the Endpoint Controller reads from the transmit FIFO. This pointer wraps around to zero if TFnS is reached. TXRP is never incremented beyond the value of the write pointer TXWP.

An underrun condition occurs if TXRP equals TXWP and an attempt is made to transmit more bytes when the LAST bit in the TXCMDx register is not set.

- **TXWP: Transmit Write Pointer**

This pointer is incremented every time the firmware writes to the transmit FIFO. This pointer wraps around to zero if TFnS is reached.

If an attempt is made to write more bytes to the FIFO than actual space available (FIFO overrun), the write to the FIFO is ignored and TCOUNT is checked for an indication of the number of empty bytes remaining.

- **TXFL: Transmit FIFO Level**

This value indicates how many bytes are currently in the FIFO.

A FIFO warning is issued if TXFL decreases to a specific value. The respective WARFn bit in the FWR register is set if TXFL is equal to or less than the number specified by the TFWL bit in the TXCn register.

- **TCOUNT: Transmit FIFO Count**

This value indicates how many empty bytes can be filled within the transmit FIFO. This value is accessible by firmware via the TXSn register.

37.2.9 Receive Endpoint FIFO

The Receive FIFOs for the Endpoints 2, 4, and 6 support bulk and interrupt USB packet transfers larger than the actual FIFO size. If the packet length exceeds the FIFO size, the firmware must read the FIFO contents while the USB packet is being received on the bus.

Figure 99 illustrates the operation of the receive FIFOs.

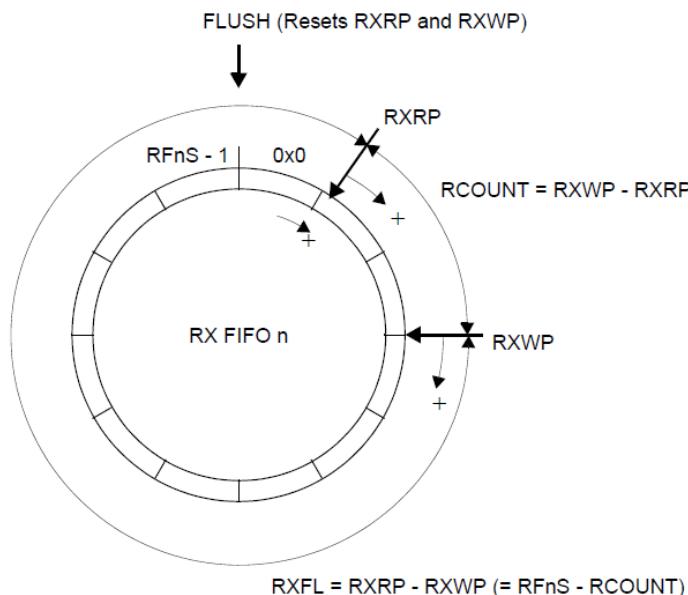


Figure 99: USB Rx FIFO Operation

- **RFnS: Receive FIFO n Size**

This is the total number of bytes available within the FIFO.

- **RXRP: Receive Read Pointer**

This pointer is incremented with every read of the firmware from the receive FIFO. This pointer wraps around to zero if RFnS is reached. RXRP is never incremented beyond the value of RXWP.

If an attempt is made to read more bytes than the actually available bytes (FIFO underrun), the last byte is read repeatedly.

- **RXWP: Receive Write Pointer**

This pointer is incremented every time the Endpoint Controller writes to the receive FIFO. This pointer wraps around to zero if RFnS is reached.

An overrun condition occurs if RXRP equals RXWP and an attempt is made to write an additional byte.

- **RXFL: Receive FIFO Level**

This value indicates how many more bytes can be received until an overrun condition occurs with the next write to the FIFO.

A FIFO warning is issued if RXFL decreases to a specific value. The respective WARFn bit in the FWR register is set if RXFL is equal to or less than the number specified by the RFWL bit in the RXCn register.

- **RCOUNT: Receive FIFO Count**

This value indicates how many bytes can be read from the receive FIFO. This value is accessible by firmware via the RXSn register.

37.2.10 Interrupt Hierarchy

Figure 100 shows the register hierarchy for generating USB interrupt events. Each bit in the event register can be masked by setting the corresponding bit in the xxxMSK_REG. A USBFS_IRQ to the CPU is generated if one or more bits in the MAEV_REG are set and the corresponding bits in the MAMSK_REG are set to '1'. Bit 7 in the MAMSK_REG is a global interrupt enabled for the USBFS_IRQ.

Multi-core Bluetooth 5.2 SoC family with system PMU

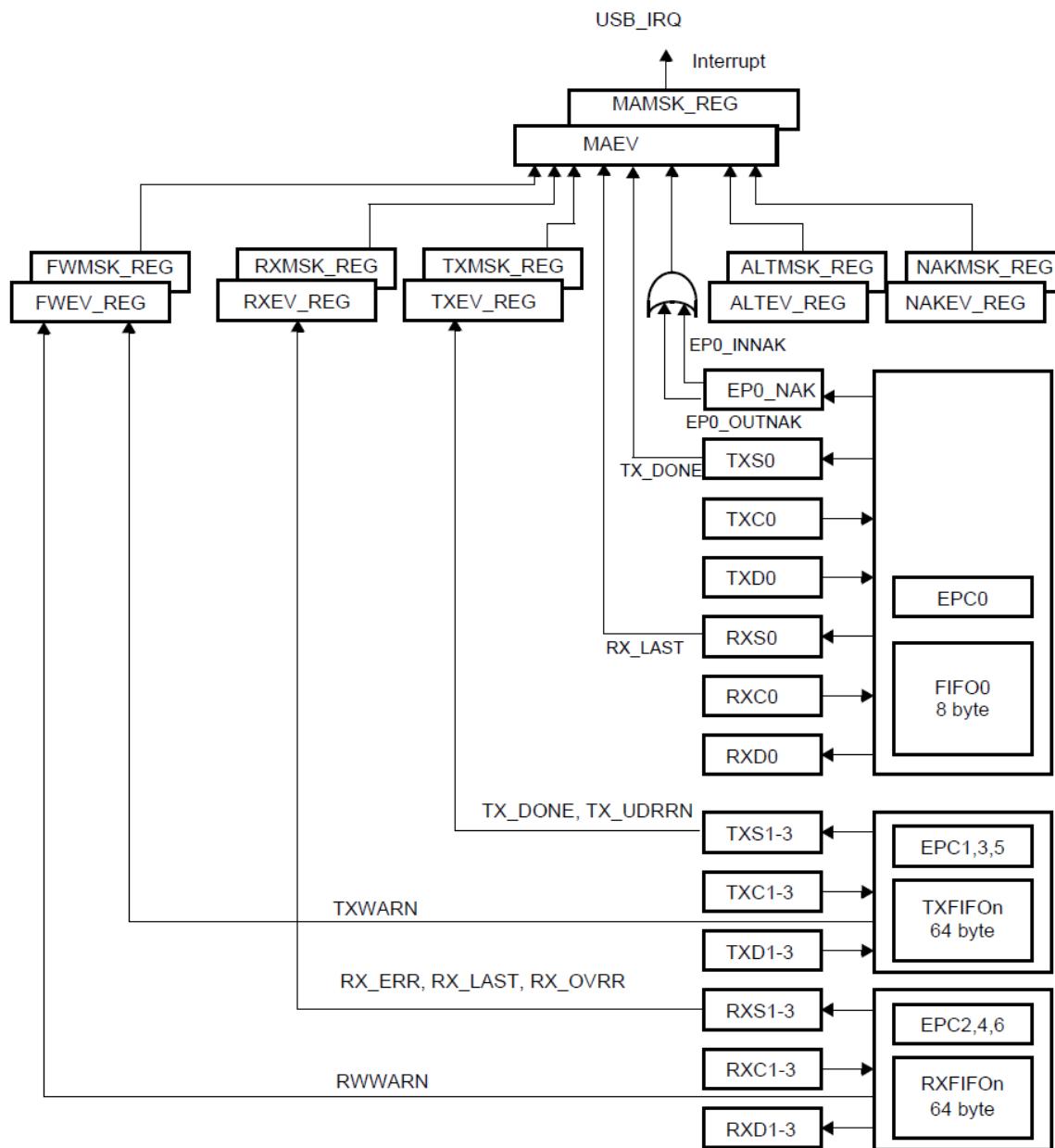


Figure 100: USB Interrupt Register Hierarchy

37.2.11 USB Power Saving Modes

In NodeSuspend state the USB transceiver is automatically switched to a low power mode. Optimal power saving rest of the chip level can be obtained by:

- Executing from on-chip RAM or ROM. On-chip access is more power efficient than external memory access. With external FLASH, appropriate gating of the chip selects may be applied
- Switching the PLL to bypass mode and then switching it off (CLK_CTRL_REG=0)
- Switching XTAL supply to AVD1 (AVD_SUPPLY=1, default state)
- Switching off AVD_XTAL (LDO_XTAL_ON = 0) to eliminate current through external potentiometers
- Switching off DCDC converter, LED current, GenDSP, and others
- Setting bandgap at the lowest voltage and internal current (BANDGAP_REG = 0x8)
- Enable LDO_SUSPEND and switch off CHARGE control circuit

Multi-core Bluetooth 5.2 SoC family with system PMU

Note that the XTAL cannot be switched off. So there is always a clock available for interrupt generation and keyboard scanning.

In suspend mode, any activity on the USB generates the USB_ALTEV[RESUME] interrupt.

The firmware must respond by:

- Starting the PLL and switching the system clock from XTAL to PLL mode to provide 48 MHz to the USB clock
- Re-enabling all functions and setting that have been turned off to reduce the currents in suspend mode

Note that the total time needed to bring the USB module back to operational mode is determined by the PLL startup time.

The CPU resumes normal operation after USB_NFSR register is set in NodeOperational state.

37.2.11.1 Freezing USB Node

The USB module provides support for an In-System-Emulator. If SET_FREEZE_REG[FRZ_USB] is set, the USB module will exhibit the following behaviors:

- The automatic Clear-on-Read function of status flags is disabled
- The FIFOs are not updated

To enable the module, the RESET_FREEZE_REG[FRZ_USB] must be set to 1 again.

37.2.11.2 Integrated resistors

The USB transceiver has integrated resistors as specified in *USB Specification 2.0 ECN “pull-up/pull-down resistors”*.

The resistors switching can be overruled as shown in [Figure 101](#) and [Figure 102](#), if `USB_UX20CDR_REG[RPU_TEST_EN]` = 1.

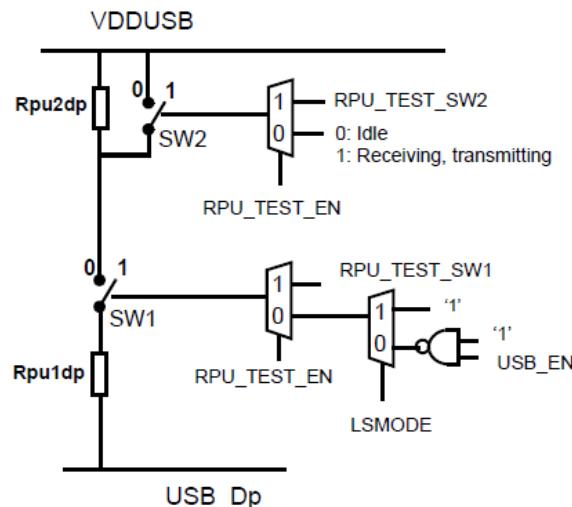


Figure 101: USB_Dp Resistor Switching

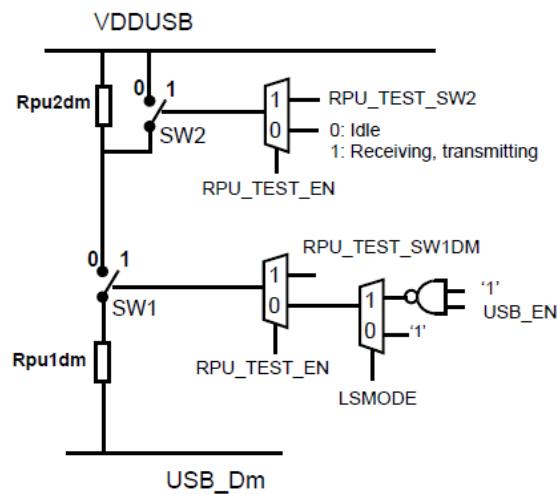


Figure 102: USB_Dm Resistor Switching

38 Haptic Driver

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | x | x | ✓ | ✓ |

38.1 Introduction

The haptic feedback driver and controller in DA1469x can be used in conjunction with external devices of type Linear Resonant Actuator (LRA) or Eccentric Rotating Mass (ERM). It automatically adapts to the resonant frequency of the haptic feedback actuator and has a configurable supply current to set the force of the haptic feedback. Furthermore, active acceleration and rapid stop functionality is implemented, enabling short bursts of haptic feedback to represent, for example, typing on a keyboard while using a touch screen.

Features

- Fully digitally controlled drive current
- Fully digitally and software controlled resonant frequency search
- Differential output driver capable of LRA and ERM
- Automatic LRA resonant frequency tracking within +/- 20% of programmed frequency
- “Active acceleration” and “rapid stop” technology for crisper haptic pattern generation, fully software controlled
- Built-in short circuit protection
- Edge Rate Control for minimized EMI
- Minimized power consumption in IDLE Mode
- 250 mA max output current

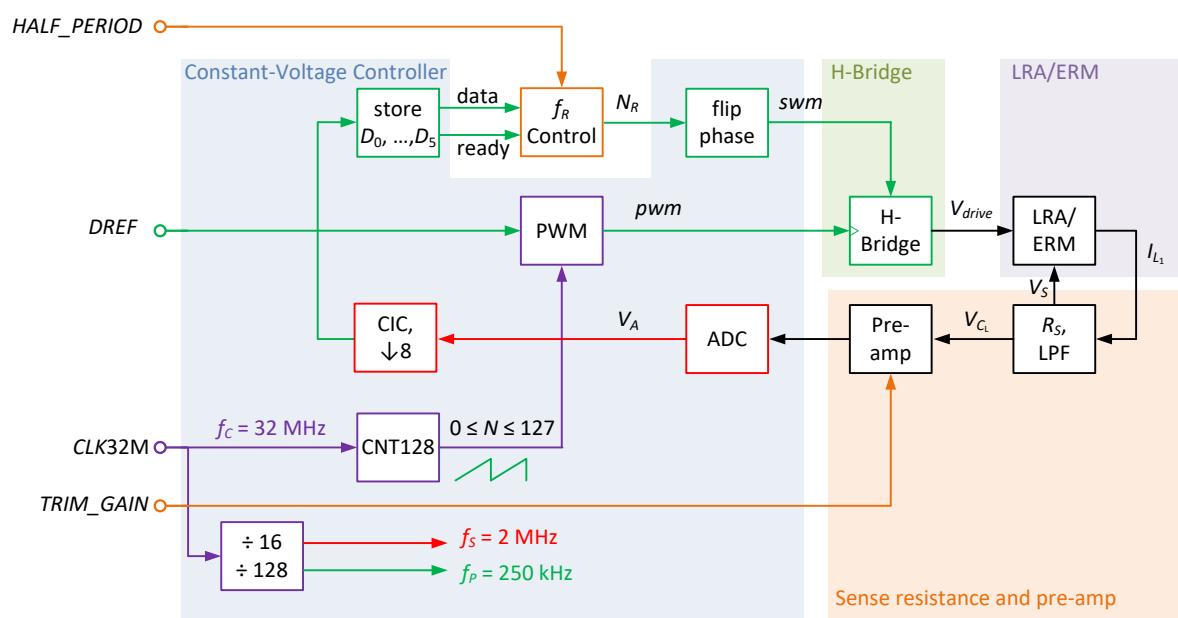


Figure 103: Block Diagram of the Haptic Driver

38.2 Architecture

The haptic feedback driver architecture is shown in Figure 103. It has the following sub blocks:

- Clock dividers
 - The system operates on the 32 MHz clock (system clock) and generates local frequencies of 2 MHz and 250 kHz, which are divided by 16 and 128, respectively

Multi-core Bluetooth 5.2 SoC family with system PMU

- Conceptually, a ramp generator (CNT128) provides a 250 kHz signal used for generating the pulse width modulation control for the H-bridge driver
- Pulse Width Modulator (PWM)
 - The 250 kHz ramp signal is compared with the value LRA_CTRL3_REG[DREF] to generate a square wave signal to drive the H-bridge with a variable duty cycle
 - The placement of the activation pulse can be controlled with LRA_CTRL2_REG[PWM_MODE] at the beginning/end of or in the middle of the period. The placement can also alternate between these settings to control the harmonic content of the supply current
- Flip Phase
 - A programmable activation frequency of the LRA/ERM can be configured by LRA_CTRL2_REG[HALF_PERIOD] in multiples of 4 μ s
 - For applications using an ERM, it is sufficient to set it to a fixed value
 - For applications using an LRA, its value must track the resonant frequency of the external component, which is notoriously sensitive to a range of environmental conditions. Refer to section [38.2.1](#) for a detailed explanation.
- H-Bridge driver
 - Fully configurable for edge rate control (ERC) and short circuit protection (SCP)
 - These configurations can be optimized using LRA_BRD_LS_REG and LRA_BRD_HS_REG for the low and high side of the bridge, respectively
- LRA/ERM, external components to close the controller loop
 - A wide range of haptic feedback devices is supported with a maximum load current up to 250 mA
- Sense resistance (R_s) and low-pass filter
 - The nominal value of R_s is 1 Ω but varies by up to $\pm 20\%$ due to process variations. These variations should be calibrated to fix the loop gain
 - The sense resistance is applied on both sides of the H-Bridge and the appropriate sense voltage is automatically connected to the pre-amp via a low-pass filter to minimize the switching transients
- Pre-Amplifier
 - As the maximum current is limited to 250 mA, the sense voltage is nominally limited to 20 mV and thus may be amplified to exploit the full range of the subsequent ADC
 - Depending on the desired load current, the pre-amplifier gain may be configured to six, eight (default), 10, or 12. The highest gain factor is meant to be used for the least desired current settings and the gain settings of six and 10 are meant to partially compensate the gain variations due to process tolerance of the sense resistance
- Analog to Digital Converter (ADC)
 - The 10-bit sense voltage is padded with zeros to a 16-bit signed representation with eight fractional bits
 - The sampling rate is 2 MHz
 - The offset voltage representing zero load current can be compensated by programming LRA_ADC_CTRL1_REG[LRA_ADC_OFFSET] to have an 8-bit signed value and three fractional bits
- Cascaded Integrator Comb (CIC) filtering and decimation by a factor of eight
 - This block computes a running mean and decimates the sense signal to the control rate of 250 kHz
- Store D_1, \dots, D_8
 - During each phase of the actuation (a half period), eight samples are stored at equidistant sample times. These samples can be used to aid resonant frequency tracking

Multi-core Bluetooth 5.2 SoC family with system PMU

- During the first half period, samples D_1 through D_8 are stored and during the second half period, samples D_9 through D_{16} are stored
- The samples are stored in LRA_FLT_SMP x _REG[LRA_SMP_(2 x - 1), LRA_SMP_2 x] with $1 \leq x \leq 8$
- After a specific sample given by LRA_CTRL1_REG[IRQ_IDX] is stored, an interrupt is generated to trigger calculation of the updated value of LRA_CTRL2_REG[HALF_PERIOD]. When the index is set to 0xF, the interrupt is generated after sample 16 is stored. When the index is set to 0x7 (default), the interrupt is generated both after sample 8 is stored and after sample 16 is stored. This functionality is useful when the load behaves in a symmetric fashion

38.2.1 Resonance Control

When an LRA is used as a load, the resonant frequency of the external device must be tracked across a wide range of environmental variations to maintain a constant haptic feedback force.

In constant voltage mode, the steady state current in the LRA is set by the series resistance R_1 as shown in [Figure 104](#), modulated by the back-electromotive-force (back-EMF) generated by the parallel resonant circuit L_2 , C_1 , R_2 in the same figure. When resonance is achieved, the current in the device represents a square wave with a dip in each half curve representing half a sinusoidal. The sense signal is a linear representation of this load current.

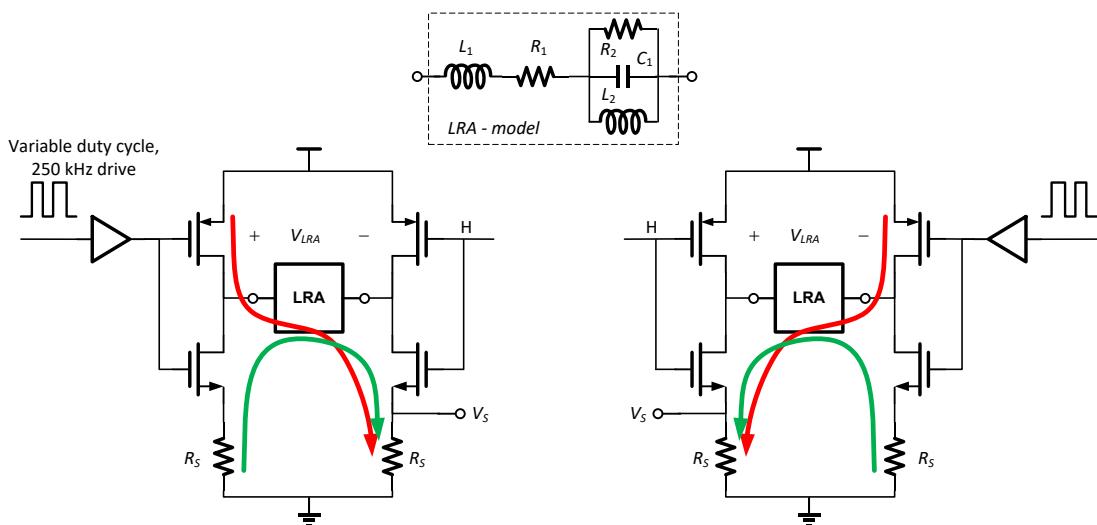


Figure 104: Circuit Diagram of the Haptic Driver and Equivalent Electrical Model of an LRA

The mathematical background is given here for reference. The LRA is driven by pulse width modulated signals, and the polarity of this PWM signal is changed every half period $T/2$. The amplitude of the LRA response is proportional to pulse width of the PWM signal. We can approximate the voltage drives as:

$$v(t) = P \cdot V \quad (1)$$

In the equation [above](#), the polarity $P = +1$ or -1 :

$$\text{when } \text{mod}(t, T) < \frac{T}{2}, P = 1, \\ \text{Otherwise, } P = -1$$

We can approximate the response of the LRA as:

$$x(t) = A \cos(\omega t) + B \sin(\omega t) \quad (2)$$

where $\omega = \frac{2\pi}{T}$, A is the ‘in phase’ amplitude, and B is the ‘quadrature’ amplitude.

Multi-core Bluetooth 5.2 SoC family with system PMU

A simplified equation describing the electrical behavior of the system is:

$$v(t) = R_1 i(t) + \varphi \frac{d}{dt} x(t) \quad (3)$$

where R_1 is the DC resistance and φ is a constant that translates currents into forces.

The LRA control system measures the current relative to the polarity of the voltage during a constant voltage segment:

$$i_M(t) = P.i(t) = \frac{V}{R_1} - \omega A \sin(\omega t) + \omega B \cos(\omega t) \quad (4)$$

Every half a period, the LRA control algorithm does a least squares fit a constant, sine and cosine term to the measured current $i_M(t)$, and adjust the phase and the frequency of the drive signal $v(t)$ such that $B = 0$ and A is positive (or negative for fast breaking). The amplitude of the LRA response is directly proportional to the sine term. The sine term can be used as input for a secondary control loop that varies the drive strength in such a way that the desired response amplitude A is achieved in minimal time (overdrive).

The software provided as part of the SDK approximates the sense signal with a combination of a constant value, a sine function, and a cosine function, elucidating to what degree the drive signal is tuned to the resonant frequency. The result is used to update the value of HALF_PERIOD in accordance to the resonant frequency.

38.3 Programming

The haptic feedback driver supports ERM and LRA devices. It has two basic means of operation:

- ERM can be setup for a desired force with a specific duty cycle pre-set using LRA_CTRL3_REG[DREF]. The enable signal LRA_CTRL1_REG[LRA_EN] switches it on and off.
- LRA can operate at varying strengths. For optimal usage, LRA needs to be actuated at its resonant frequency. This resonance depends on various factors (temperature, orientation, movement, humidity, and others), hence it needs to be controlled.

There is a simple sequence of steps that need to be followed:

1. Configuration:

Pre-set the desired initial conditions for the actuation frequency (LRA_CTRL2_REG[HALF_PERIOD]), the current amplitude (LRA_CTRL3_REG[DREF]), and the initial duty cycle of the constant current controller (LRA_CTRL3_REG[DREF]).

There are two operational modes:

- Constant current mode: it is enabled by default.
- Constant duty cycle mode: it can be selected by setting the state-space filter coefficients to 0 (LRA_FLT_COEFn_REG, with n = 1, 2, and 3) and selecting the ADC output samples (LRA_CTRL1_REG[SMP_SEL]) to be stored for the resonance control algorithm. Note that the values of the above registers should be chosen according to the external haptic device used.

2. Enable/Disable:

Setting the LRA_CTRL1_REG[LRA_EN] bit enables the haptic feedback, hence the current is controlled automatically. However, the resonant frequency remains at its initial setting.

3. Resonance Control:

In order to track the resonant frequency, an interrupt-controlled SW loop needs to be activated. Every half cycle of the actuation (around 2.5 ms intervals, when resonance frequency is set at 200 Hz) an interrupt is generated which should trigger the Cortex-M33 to run the code routine, reading up to eight automatically stored samples, calculating the updated resonant frequency, and writing it back to the register.

Multi-core Bluetooth 5.2 SoC family with system PMU

4. If an amplitude modulation is required, the patterns that adjust the current over time should run in software. A special case of amplitude modulation, like the fast start-up and rapid stop functionality, which requires abrupt pulses (that is, typing sensation on a touch screen), can also be implemented in SW.

38.4 Legal

NOTWITHSTANDING ANYTHING TO THE CONTRARY IN THIS DOCUMENT OR IN ANY OTHER AGREEMENT, CUSTOMER ACKNOWLEDGES THAT THE PRODUCTS ARE PROVIDED WITHOUT ANY WARRANTY OF NON-INFRINGEMENT, AND THAT IF ANY LICENSE OR RIGHT IS REQUIRED TO BE OBTAINED FROM ANY THIRD PARTY, THIS RESPONSIBILITY SHALL REST SOLELY WITH CUSTOMER.

39 LEDs Driver

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | x | x | ✓ | ✓ |

39.1 Introduction

The LED driver features two matched white LED outputs with an absolute accuracy of +/-5%. It supports two solutions for brightness dimming: a selectable output load current and PWM dimming.

The selectable output load is a dimming mode that changes the brightness by changing/reducing the maximum output current. It is controlled by three bits (eight possible settings). The minimum setting is 2.5 mA with an increment of 2.5 mA, so maximal 20 mA. The reduced current settings are especially interesting to be used as a coarse brightness control for short pulses, for example, for a heart rate monitor application.

PWM dimming changes the brightness by modulating the output current from 0% to 100% duty-adjustable pulse. The LED brightness is controlled by adjusting the relative ratios of the on and off times. Since the dimming frequency (timer operating on the low power clock) is higher than the human-eye sensitive range (>100 Hz), the effective brightness is perceived by averaging the on and off times. The PWM operation is possible for all eight load settings instead of just the maximum setting.

The block diagram of just one driver is presented in [Figure 105](#). Two of these are instantiated in the system.

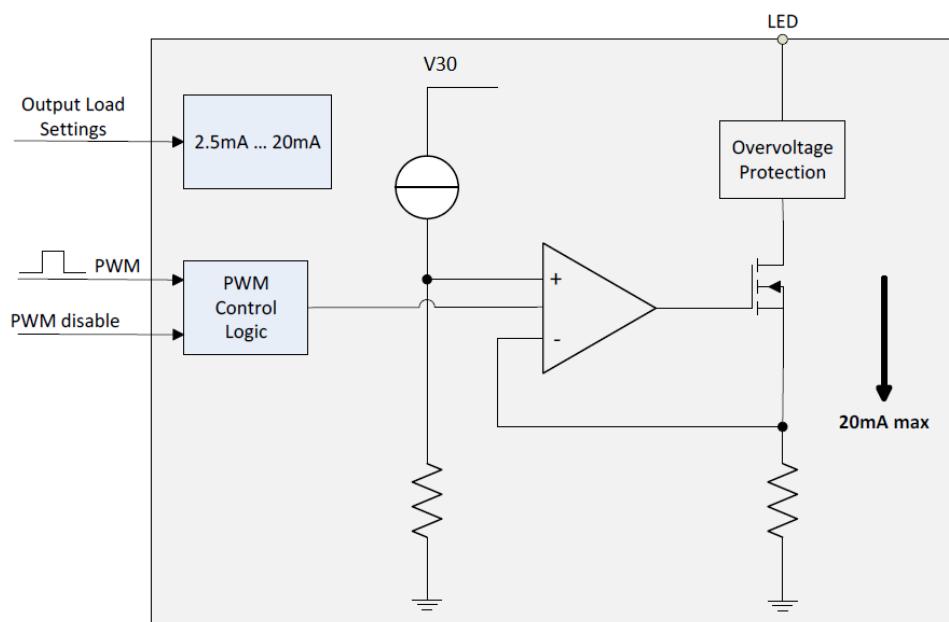


Figure 105: LED Driver Block Diagram

39.2 Architecture

The LED driver contains the following functional blocks:

- Output load setting. Input is an accurate reference current from a bandgap, and outputs are various (mirrored) currents
- High gain op-amp
- Matched resistor network
- PWM control logic

Multi-core Bluetooth 5.2 SoC family with system PMU

- Overvoltage protection

A reference current, coming from the bandgap, is mirrored into selectable output current(s). This current is routed through a resistor, resulting in a reference voltage on the positive input of the op-amp. Due to the negative feedback and the high gain, the op-amp in combination with the output transistor tries to force the same voltage on the negative input and over the output resistor, resulting in the selected output load current. The overvoltage protection shields the LED driver from excessive high voltages, which can be up to VBUS max, on the LED pins.

39.3 Programming

There is a simple sequence of steps that needs to be followed to program the LED drivers:

1. Define the PWM frequency and duty cycle for each LED driver by configuring PWMLED_DUTY_CYCLE_LED1/2_REG and PWMLED_FREQUENCY_REG. Note that it is possible to use different duty cycles for the two drivers, however they share the same frequency configuration register
2. Configure the sinking current by programming PWMLED_CTRL_REG[LEDx_LOAD_SEL]. This defines the sinking current according to the following formula:

$$I_{LED(sink)} = 2.5 \text{ mA} + (\text{LEDx_LOAD_SEL} \times 2.5 \text{ mA}) \quad (5)$$

3. Enable the LED drivers by asserting PWMLED_CTRL_REG[LED1_EN, LED2_EN]
4. Activate PWM by asserting PWMLED_CTRL_REG[PWM_ENABLE]

Note that the PWM can be stopped by writing to either SET_FREEZE_REG[FRZ_PWMLED] or PWMLED_CTRL_REG[SW_PAUSE_EN]. Furthermore, the control logic of the LEDs resides in PD_PER. PD_COM can be switched off during sleep with no impact on the LEDs operation.

40 Input/Output Ports

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------|---------|---------|---------|---------|
| P1_12 to P1_22: | x | x | ✓ | ✓ |

40.1 Introduction

The DA1469x has software-configurable input/output (I/O) pin assignment organized into ports Port 0 and Port 1.

Features

- Port 0 has 32 pins and Port 1 has 23 pins (including M33_SWCLK, M33_SWDIO, CMAC_SWCLK, and CMAC_SWDIO)
- Fully programmable pin assignment (PPA)
- Selectable pull-up and pull-down resistors of 25 kΩ per pin
- Programmable open-drain functionality
- Pull-up voltage at V30/V18P voltage configurable per pin
- Fixed assignment for analog pin, motor controller, and LCD controller pins
- Pins can retain their last state when system enters the Extended, Deep Sleep or Hibernation mode

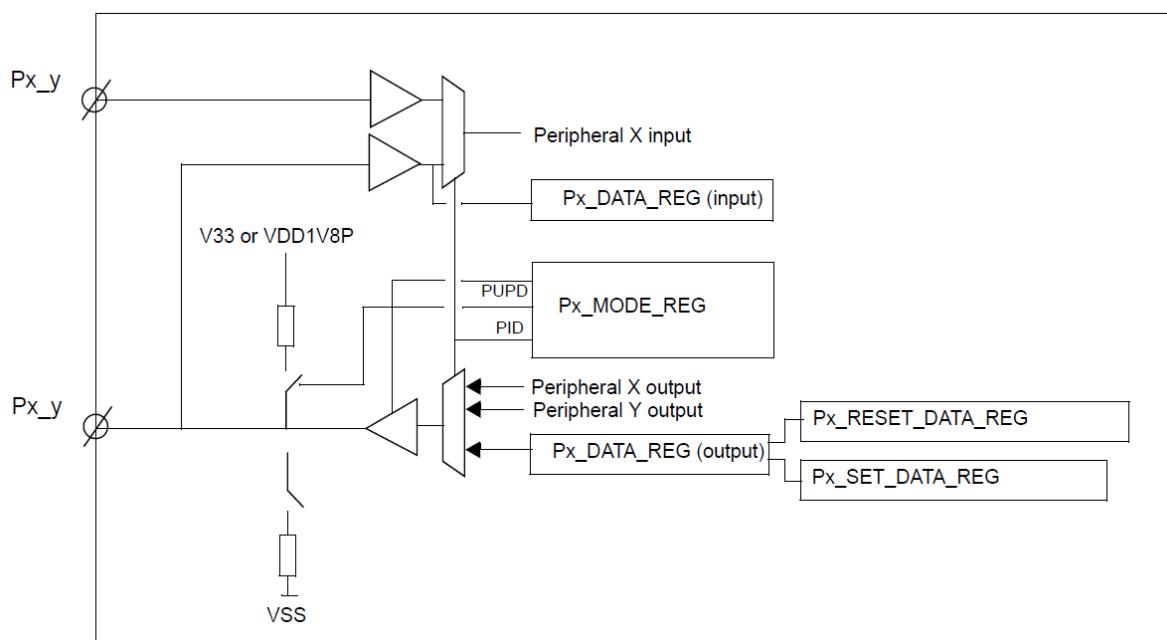


Figure 106: Port P0 and P1 with Programmable Pin Assignment

40.2 Architecture

40.2.1 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 signals can be freely mapped to any I/O port bit by setting Px_yy_MODE_REG[5:0]

0x00 to 0x3B: Peripheral IO ID (PID)

Refer to the Px_yy_MODE_REGS for an overview of the available PIDs. Analog, GPADC, and SDADC signals have fixed pin assignment in order to limit interference with the digital domain. The

Multi-core Bluetooth 5.2 SoC family with system PMU

same applies to Motor and LCD Controller signals. The M33_SWD interface is mapped on P0_10 and P0_11 and the CMAC_SWD interface on P0_12 and P0_13.

40.2.1.1 Priority

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

In case more than one input signals are assigned to a peripheral input, the left most pin in the lowest port pin number has priority, for example, P0_00_MODE_REG has priority over P0_01_MODE_REG.

40.2.1.2 Direction Control

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

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

40.2.2 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

40.2.2.1 Port Data Register

The registers input Px_DATA_REG and output Px_DATA_REG are mapped on 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-M33 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 the register changes the output value.

40.2.2.2 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.

40.2.2.3 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.

40.2.3 Fixed Assignment Functionality

There are certain signals that have a fixed mapping on specific general purpose IOs. This assignment is illustrated in Table 146:

Table 146: Fixed Assignment of Specific Signals

| GPIO | SWD (Note 1) | QSPI RAM | Analog (Note 2) | Clocks (Note 3) | PWM (Note 7) | Motor Controller | LCD Controller (Note 4) | Diagnostics |
|-------|-----------------|-----------|--------------------|--------------------|-----------------|---------------------|----------------------------|--|
| P0_00 | | QSPIR_D0 | | | | PG3_4 | | |
| P0_01 | | QSPIR_D1 | | | | PG4_2 | | |
| P0_02 | | QSPIR_D2 | | | | PG4_1 | | |
| P0_03 | | QSPIR_D3 | | | | PG3_2 | | |
| P0_04 | | QSPIR_CS | | | | PG3_1 | | |
| P0_05 | | QSPIR_CLK | | | | PG3_3 | | |
| P0_06 | | | SDADC_GND | | | | | |
| P0_07 | | | NTC Input | | | | | |
| P0_08 | | | GPADC_2 \ SDADC_2 | | | PG4_3 | | |
| P0_09 | | | GPADC_3 \ SDADC_3 | | | PG4_4 | | |
| P0_10 | M33_SWDIO | | | | | | | |
| P0_11 | M33_SWCLK | | | | | | | |
| P0_12 | CMAC_SWDIO | | | XTAL32M | | | | |
| P0_13 | CMAC_SWCLK | | | RC32M | | | | |
| P0_14 | | | | XTAL32k | | | | |
| P0_15 | | | | DIVN | | | | |
| P0_16 | | | SDADC_REF | RCX | | | | WOKENUP or CMAC_SLP_TIMER_EXPIRE (Note 6) |

| GPIO | SWD (Note 1) | QSPI RAM | Analog (Note 2) | Clocks (Note 3) | PWM (Note 7) | Motor Controller | LCD Controller (Note 4) | Diagnostics |
|-------|-----------------|------------|----------------------|--------------------|-----------------|---------------------|----------------------------|-------------------------|
| P0_17 | | | | RC32k | | PG2_1 | | |
| P0_18 | | | | | | PG2_2 | | CMAC_DIAG_7 (Note 5) |
| P0_19 | | | | | | PG2_3 | | CMAC_DIAG_6 (Note 5) |
| P0_20 | | | | | | PG2_4 | | CMAC_DIAG_5 (Note 5) |
| P0_21 | | | | | | PG1_3 | | CMAC_DIAG_4 (Note 5) |
| P0_22 | | | | | | | | |
| P0_23 | | | | | | | | |
| P0_24 | | | | | | PG1_4 | LCD_TE | CMAC_DIAG_3 (Note 5) |
| P0_25 | | | GPADC_1 \ SDADC_1 | | | | | BANDGAP_ENABLE (Note 6) |
| P0_26 | | | | | | PG0_1 | LCD_VCK | CMAC_DIAG_13 (Note 5) |
| P0_27 | | | | | | PG0_2 | LCD_ENB/PLCD_ENAB | CMAC_DIAG_12 (Note 5) |
| P0_28 | | | | | | PG0_3 | LCD_VST/PLCD_VSYNC | CMAC_DIAG_11 (Note 5) |
| P0_29 | | | | | | PG0_4 | LCD_HCK/PLCD_CLK | CMAC_DIAG_10 (Note 5) |
| P0_30 | | | | | | PG1_1 | LCD_HST/PLCD_HSYNC | CMAC_DIAG_9 (Note 5) |
| P0_31 | | | | | | PG1_2 | LCD_XRST | CMAC_DIAG_8 (Note 5) |
| P1_00 | | NTC Supply | | | | | | |
| P1_01 | | | | | Timer.PWM | | | |
| P1_02 | | | | | | | LCD_BLUE0 | |
| P1_03 | | | | | | | LCD_BLUE1 | |
| P1_04 | | | | | | | LCD_GREEN0 | |
| P1_05 | | | | | | | LCD_GREEN1 | |
| P1_06 | | | | | Timer.PWM2 | | | CMAC_DIAG_14 (Note 5) |
| P1_07 | | | | | | | LCD_RED0 | |

Multi-core Bluetooth 5.2 SoC family with system PMU

| GPIO | SWD (Note 1) | QSPI RAM | Analog (Note 2) | Clocks (Note 3) | PWM (Note 7) | Motor Controller | LCD Controller (Note 4) | Diagnostics |
|-------|-----------------|----------|----------------------|--------------------|-----------------|---------------------|--------------------------------------|-----------------------|
| P1_08 | | | | | | | LCD_RED1 | |
| P1_09 | | | GPADC_0 \ SDADC_0 | | | | | CMAC_DIAG_15 (Note 5) |
| P1_10 | | | | | | | LCD_VCOM / LCD_FRP / LCD_EXTCOMIN | |
| P1_11 | | | | | | | LCD_XFRP | |
| P1_12 | | | GPADC_5 | | | | | |
| P1_13 | | | GPADC_4 | | | | LCD_VCK | |
| P1_14 | | | SDADC_4 | | | | LCD_ENB / PLCD_ENAB | |
| P1_15 | | | | | | | LCD_VST / PLCD_VSYNC | |
| P1_16 | | | | | | | LCD_HCK / PLCD_CLK | |
| P1_17 | | | | | | | LCD_HST / PLCD_HSYNC | |
| P1_18 | | | GPADC_6 | | | | | |
| P1_19 | | | GPADC_7 | | | | | |
| P1_20 | | | SDADC_5 | | | | | |
| P1_21 | | | SDADC_6 | | | | LCD_XRST | |
| P1_22 | | | SDADC_7 | | | | LCD_TE | |

Note 1 The SWD signals mapping is defined by SYS_CTRL_REG[DEBUGGER_ENABLE] and SYS_CTRL_REG[CMAC_DEBUGGER_ENABLE]. However, these signals are mapped on the ports by default.

Note 2 The ADC case can be selected by the PID bit field on the respective Px port.

Note 3 Enable specific clock outputs through GPIO_CLK_SEL_REG register.

Note 4 Refer to LCD Controller chapter to choose the correct LCD pinning combination between the available LCD interfaces.

Note 5 CMAC_DIAG_[2:0] can be assigned to any GPIO through the Programmable Pin Assignment (PPA). For a complete functionality description of the CMAC diagnostics see section “Configurable MAC (CMAC)”.

Note 6 PMU_CTRL_REG[MAP_BANDGAP_EN].

Note 7 Timer.PWM and Timer.PWM2 signals are available during sleep (CLK_TMR_REG[TMR_PWM/2_AON_MODE]).

40.2.4 GPIO State Retention While Sleeping

Before setting the system to any sleep modes, the state of the pads needs to be retained in order to avoid external components being affected by the GPIOs changing states when the system goes to sleep and to avoid any floating driving signals from shut-off power domains leading to increasing power dissipation.

The state of the pads is automatically latched by always-on latches by setting the corresponding PAD_LATCH_EN bit in the Px_RESET_PAD_LATCH_REG. These bits latch the digital control signals going into the pad and latch the data output separately for each pin. Hence, if the pad has been set as an output driving high, it retains its precise state. Px_SET_PAD_LATCH_REG is used to unlatch the pins.

The signals in red in Figure 107 and Figure 108 are latched:

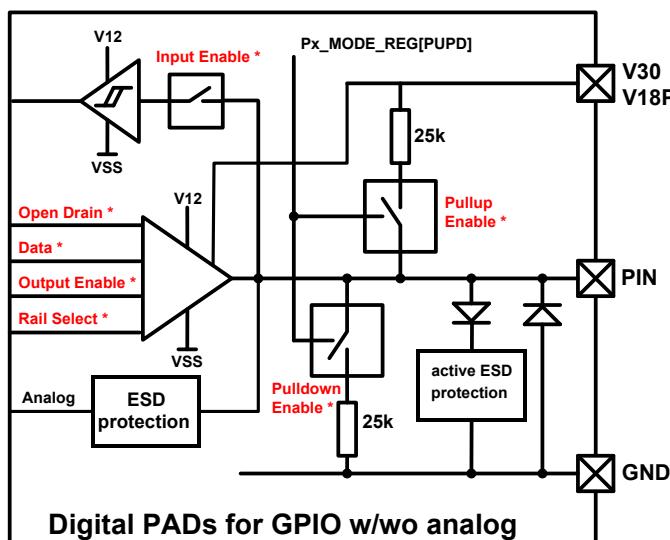


Figure 107: Latching of Digital Pad Signals

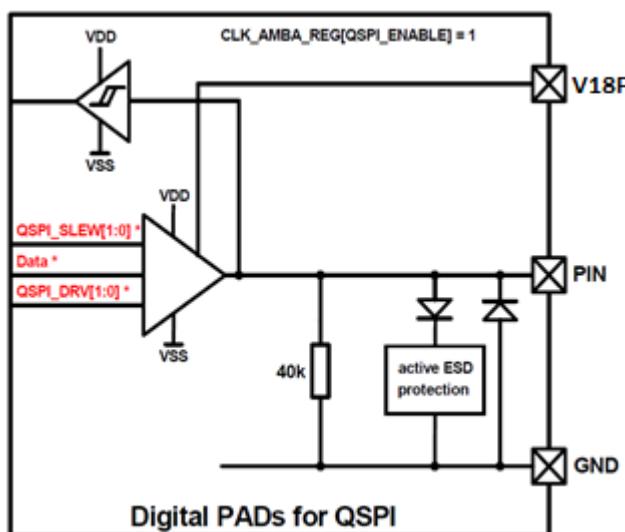


Figure 108: Latching of QSPI Pad Signals

After the system waking up, the software must disable the latching by setting the corresponding PAD_LATCH_EN bits of the Px_PAD_LATCH_REG so that all pads can be accessed and controlled again.

Multi-core Bluetooth 5.2 SoC family with system PMU

For QSPI pads, the pad latching is overwritten by the QSPI controller as soon as the clock of the controller is enabled.

40.2.5 Special I/O Considerations

There are certain considerations in using the GPIOs:

- To use P0_14 or P0_15 in GPIO mode, USBPAD_REG[USBPAD_EN] must be set. However, the levels allowed on these pins are 0 V and 3 V. The voltage comes from the V30 rail. If 1.8 V is selected as the pin supply, a current of 150 μ A is to be expected. Moreover, these pins should not be used in sleep modes, because the USBPAD_REG which belongs to the system power domain is powered off in sleep modes and those pins do not support state retention during power down
- P0_18, P0_16, P1_00, P0_07, P1_09, P0_27(VFBGA86) and P0_16, P1_00, P1_06, P0_18, P0_06, P0_07(VFBGA100) GPIOs might affect the performance of radio and XTAL32MHz crystal oscillator when being toggled while RF activity occurs or XTAL32MHz oscillates. It is recommended to use them at low speed and not to use them when radio is active. If used as outputs, the weak-drive capability should be selected (weak-drive capability does not apply to all the above GPIOs).
- P1_12, P1_18, and P1_19 might affect XTAL32K performance when being toggled at high frequencies while XTAL32K is in use. It is recommended to use them at low speed when set as outputs.
- P0_12 and P0_13 are by default connected to CMAC_SWDIO and CMAC_SWCLK. If GPIO function is needed they have to be programmed accordingly

41 Radio

| Device: | DA14691 | DA14695 | DA14697 | DA14699 |
|-----------------------|---------|---------|---------|---------|
| Feature Availability: | ✓ | ✓ | ✓ | ✓ |

41.1 Introduction

The Radio Transceiver provides a 103 dB RF link budget for reliable wireless communications. All RF blocks are supplied by on-chip low dropout regulators (LDOs). The bias scheme is programmable and optimized for minimum power consumption. The radio block diagram is given in [Figure 109](#). It comprises the Receiver, Transmitter, Synthesizer, Rx/Tx combiner block, and Biasing LDOs.

Features

- Single ended RFIO interface, $50\ \Omega$ matched
- Alignment free operation
- -97 dBm receiver sensitivity
- Configurable transmit output power from -18 dBm up to 6 dBm
- Ultra-low power consumption
- Fast frequency tuning minimizes overhead

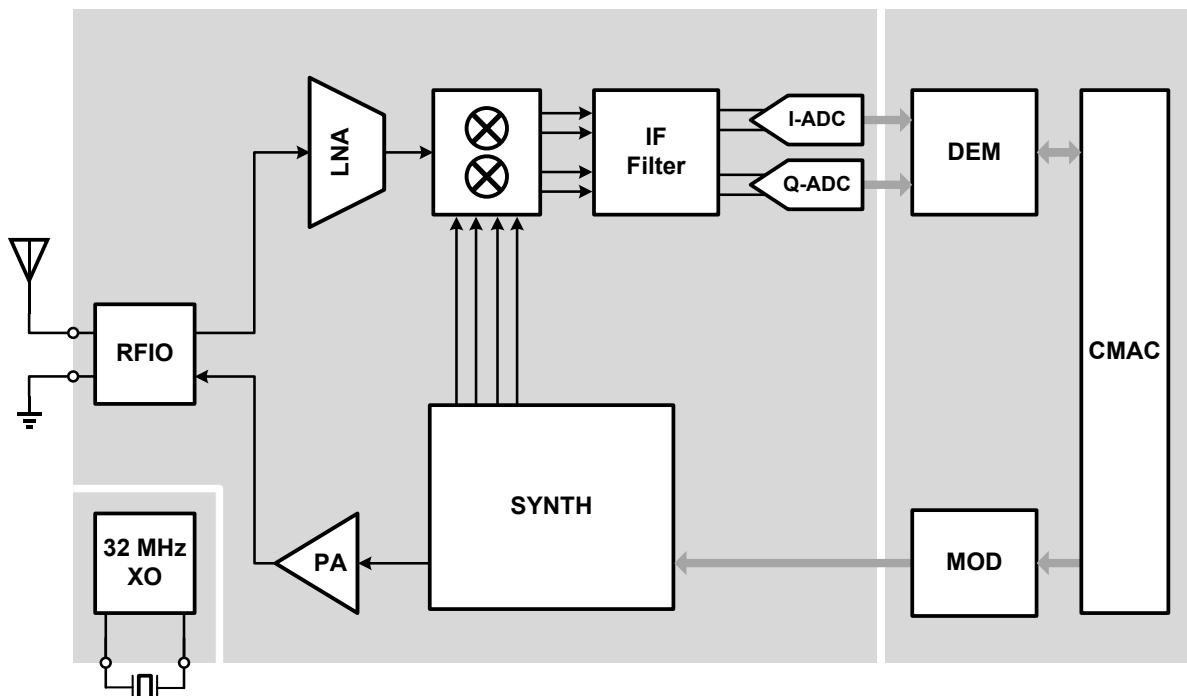


Figure 109: Radio Block Diagram

41.2 Architecture

41.2.1 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 programmable gain. The LNA and IF Filter gains are controlled by the AGC. This provides the necessary signal conditioning prior to digitalization. The digital demodulator block (DEM) provides a synchronous bit stream.

Multi-core Bluetooth 5.2 SoC family with system PMU

41.2.2 Synthesizer

The RF synthesizer generates the quadrature LO signal for the mixer. It also generates the modulated TX output signal. The DCO runs at twice the required frequency and a dedicated divide-by-two 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 2-point modulation.

41.2.3 Transmitter

The RF power amplifier (RFPA) is an extremely efficient Class-D structure, providing typically -18 dBm to +6 dBm to the antenna. It is fed by the VCO's divide-by-two circuit and delivers its TX power to the antenna pin through the combined RX/TX matching circuit.

41.2.4 RFIO

The RX/TX combiner block is a unique feature of the DA1469x. 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 $50\ \Omega$ in order to provide the simplest possible interfacing to the antenna on the printed circuit board.

41.2.5 Biasing

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

41.2.6 RF Monitoring

The radio is equipped with a monitoring block, of which the responsibility is to acquire the data provided by the functional RF Units and other various analog resources and the packing of the data in words of 32 bits (when necessary), and to store them in system's memory, in order to achieve the production test of the corresponding blocks. The data can be the output of the Demodulator (I and Q) and the data provided by the GPADC.

42 Registers

This section contains a detailed view of the DA1469x 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 Cortex-M33 and Cortex-M0+ can be found in the following documents, available on the Arm website:

Devices Generic User Guide:

[arm_cortex_m33_dgug_100235_0002_00_en.pdf](#)
[DUI0662B_cortex_m0p_r0p1_dgug.pdf](#)

Technical Reference Manual:

[Cortex_m33_trm_100230_0002_00_en.pdf](#)
[DDI0484C_cortex_m0p_r0p1_trm.pdf](#)

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

42.1 AMBA Bus Registers

Table 147: Register map DW

| Address | Register | Description |
|------------|---|--|
| 0x30020000 | AHB_DMA_PL1_REG | AHB-DMA layer priority level for RTP (AHB DMA layer only) |
| 0x30020004 | AHB_DMA_PL2_REG | AHB-DMA layer priority level for LCD (AHB DMA layer only) |
| 0x30020008 | AHB_DMA_PL3_REG | AHB-DMA layer Priority level for GEN-DMA (AHB DMA layer only) |
| 0x3002000C | AHB_DMA_PL4_REG | AHB-DMA layer Priority level for CRYPTO-DMA (AHB DMA layer only) |
| 0x30020048 | AHB_DMA_DFLT_MASTER_REG | Default master ID number (AHB DMA layer only) |
| 0x3002004C | AHB_DMA_WTEN_REG | Weighted-Token Arbitration Scheme Enable (AHB DMA layer only) |
| 0x30020050 | AHB_DMA_TCL_REG | Master clock refresh period (AHB DMA layer only) |
| 0x30020054 | AHB_DMA_CCLM1_REG | USB Master clock tokens (AHB DMA layer only) |
| 0x30020058 | AHB_DMA_CCLM2_REG | GenDMA Master clock tokens (AHB DMA layer only) |
| 0x3002005C | AHB_DMA_CCLM3_REG | CRYPTO Master clock tokens (AHB DMA layer only) |
| 0x30020060 | AHB_DMA_CCLM4_REG | CRYPTO Master clock tokens (AHB DMA layer only) |
| 0x30020090 | AHB_DMA_VERSION_REG | Version ID (AHB DMA layer only) |

Table 148: AHB_DMA_PL1_REG (0x30020000)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:4 | R | - | | 0x0 |
| 3:0 | R/W | AHB_DMA_PL1 | Arbitration priority for master RFPT. 0: lowest, 15: highest. | 0xF |

Table 149: AHB_DMA_PL2_REG (0x30020004)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:4 | R/W | - | | 0x0 |
| 3:0 | R/W | AHB_DMA_PL2 | Arbitration priority for master LCD. 0: lowest, 15: highest. | 0xE |

Table 150: AHB_DMA_PL3_REG (0x30020008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:4 | R | - | | 0x0 |
| 3:0 | R/W | AHB_DMA_PL3 | Arbitration priority for master GEN-DMA. 0: lowest, 15: highest. | 0xD |

Table 151: AHB_DMA_PL4_REG (0x3002000C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:4 | R | - | | 0x0 |
| 3:0 | R/W | AHB_DMA_PL4 | Arbitration priority for master CRYPTO-DMA. 0: lowest, 15: highest. | 0xC |

Table 152: AHB_DMA_DFLT_MASTER_REG (0x30020048)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:4 | R | - | | 0x0 |
| 3:0 | R/W | AHB_DMA_DFLT_MASTER | Default master ID number register. The default master is the master that is granted by the bus when no master has requested ownership. 0: Dummy master 1: RFPT 2: LCD 3: GEN-DMA 3: CRYPTO-DMA | 0x0 |

Table 153: AHB_DMA_WTEN_REG (0x3002004C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:1 | R | - | | 0x0 |
| 0 | R/W | AHB_DMA_WTEN | Weighted-token arbitration scheme enable. | 0x0 |

Table 154: AHB_DMA_TCL_REG (0x30020050)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|-------------|-------|
| 31:16 | R | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|--------|
| 15:0 | R/W | AHB_DMA_TCL | Master clock refresh period, counting clock cycles. An arbitration period is defined over this number of tokens. When a new arbitration period starts, the master counters are reloaded. Recommended value is the sum of the AHB_DMA_CCLMx_REG values plus 2 tokens for each master, i.e. plus 6. | 0xFFFF |

Table 155: AHB_DMA_CCLM1_REG (0x30020054)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|--|-------|
| 31:16 | R | - | | 0x0 |
| 15:0 | R/W | AHB_DMA_CCLM | Number of tokens (counted in AHB clock cycles) that a master can use on the bus before it has to arbitrate on a bus master with low priority and having tokens. Masters with tokens remaining have priority over masters that have used all of their tokens. User should configure all the token values ensuring that the sum does not exceed the total allocated number of tokens. If a value of zero is configured, then the bus is deemed to have infinite tokens and will always operate in the upper-tier of arbitration. | 0xF |

Table 156: AHB_DMA_CCLM2_REG (0x30020058)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|----------------------------|-------|
| 31:16 | R | - | | 0x0 |
| 15:0 | R/W | AHB_DMA_CCLM | Refer to AHB_DMA_CCLM1_REG | 0xF |

Table 157: AHB_DMA_CCLM3_REG (0x3002005C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|-------------------|-------|
| 31:16 | R | - | | 0x0 |
| 15:0 | R/W | AHB_DMA_CCLM | AHB_DMA_CCLM1_REG | 0xF |

Table 158: AHB_DMA_CCLM4_REG (0x30020060)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|-------------------|-------|
| 31:16 | R | - | | 0x0 |
| 15:0 | R/W | AHB_DMA_CCLM | AHB_DMA_CCLM1_REG | 0xF |

Table 159: AHB_DMA_VERSION_REG (0x30020090)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|-------------|----------------|
| 31:0 | R | AHB_DMA_VERSIO N | | 0x32313 32A |

42.2 LCD Controller Registers

Table 160: Register map LCDC

| Address | Register | Description |
|------------|---------------------------|---|
| 0x30030000 | LCDC_MODE_REG | Display Mode |
| 0x30030004 | LCDC_CLKCTRL_REG | Clock Divider |
| 0x30030008 | LCDC_BGCOLOR_REG | Background Color |
| 0x3003000C | LCDC_RESXY_REG | Resolution X,Y |
| 0x30030014 | LCDC_FRONTPORCH_XY_REG | Front Porch X and Y |
| 0x30030018 | LCDC_BLANKINGXY_REG | Blanking X and Y |
| 0x3003001C | LCDC_BACKPORCH_XY_REG | Back Porch X and Y |
| 0x30030028 | LCDC_DBIB_CFG_REG | MIPi Config Register |
| 0x3003002C | LCDC_GPIO_REG | General Purpose IO (2-bits) |
| 0x30030030 | LCDC_LAYER0_MODE_REG | Layer0 Mode |
| 0x30030034 | LCDC_LAYER0_STAR_TXY_REG | Layer0 Start XY |
| 0x30030038 | LCDC_LAYER0_SIZE_XY_REG | Layer0 Size XY |
| 0x3003003C | LCDC_LAYER0_BASE_ADDR_REG | Layer0 Base Addr |
| 0x30030040 | LCDC_LAYER0_STRIDE_REG | Layer0 Stride |
| 0x30030044 | LCDC_LAYER0_RESXY_REG | Layer0 Res XY |
| 0x30030090 | LCDC_JDI_RESXY_REG | Resolution XY for the JDI parallel I/F |
| 0x30030094 | LCDC_JDI_FBX_BLANKING_REG | Horizontal front/back blanking (hck half periods) |
| 0x30030098 | LCDC_JDI_FBY_BLANKING_REG | Vertical front/back blanking (vck half periods) |
| 0x3003009C | LCDC_JDI_HCK_WIDTH_REG | HCK high/low width |
| 0x300300A0 | LCDC_JDI_XRST_WIDTH_REG | XRST width |
| 0x300300A4 | LCDC_JDI_VST_DELAY_REG | XRST-to-VST delay |
| 0x300300A8 | LCDC_JDI_VST_WIDTH_REG | VST width |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|------------------------------|---------------------------------|
| 0x300300AC | LCDC_JDI_VCK_DELAY_REG | XRST-to-VCK delay |
| 0x300300B0 | LCDC_JDI_HST_DELAY_REG | VCK-to-HST delay |
| 0x300300B4 | LCDC_JDI_HST_WIDTH_REG | HST width |
| 0x300300B8 | LCDC_JDI_ENB_START_HLINE_REG | ENB start horizontal line |
| 0x300300BC | LCDC_JDI_ENB_END_HLINE_REG | ENB end horizontal line |
| 0x300300C0 | LCDC_JDI_ENB_START_CLK_REG | ENB start delay |
| 0x300300C4 | LCDC_JDI_ENB_WIDTH_CLK_REG | ENB width |
| 0x300300E8 | LCDC_DBIB_CMD_REG | MIPI DBIB Command Register |
| 0x300300F4 | LCDC_IDREG_REG | Identification Register |
| 0x300300F8 | LCDC_INTERRUPT_REG | Interrupt Register |
| 0x300300FC | LCDC_STATUS_REG | Status Register |
| 0x30030184 | LCDC_CRC_REG | CRC check |
| 0x30030188 | LCDC_LAYER0_OFFSET_REG | Layer0 OffsetX and DMA prefetch |

Table 161: LCDC_MODE_REG (0x30030000)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 31 | R/W | LCDC_MODE_EN | Mode register. 0 : disable 1 : enable | 0x0 |
| 30:29 | R/W | - | | 0x0 |
| 28 | R/W | LCDC_VSYNC_POL | VSYNC polarity. 0: positive 1: negative | 0x0 |
| 27 | R/W | LCDC_HSYNC_POL | HSYNC polarity. 0: positive 1: negative | 0x0 |
| 26 | R/W | LCDC_DE_POL | DE polarity. 0: positive 1: negative | 0x0 |
| 25:24 | R/W | - | | 0x0 |
| 23 | R/W | LCDC_VSYNC_SCP_L | Set VSYNC for a single cycle per line. 0: disable 1: enable | 0x0 |
| 22 | R/W | LCDC_PIXCLKOUT | Pixel clock out polarity. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| | | _POL | 0: positive 1: negative | |
| 21:20 | R/W | - | | 0x0 |
| 19 | R/W | LCDC_FORCE_BLA_NK | Forces output to blank. 0: disable 1: enable | 0x0 |
| 18 | R/W | - | | 0x0 |
| 17 | R/W | LCDC_SFRAME_UPD | Single frame update. 0: disable 1: enable | 0x0 |
| 16:12 | R/W | - | | 0x0 |
| 11 | R/W | LCDC_PIXCLKOUT_SEL | Selects the pixel out clock for the display. 0: based on the pixel pipeline clock 1: based on the format pipeline clock See also the LCDC_CLKCTRL_REG. | 0x0 |
| 10:9 | R/W | - | | 0x0 |
| 8:5 | R/W | LCDC_OUT_MODE | Selection of the output mode 0000: Parallel RGB 1000: JDI MIP All the other values are reserved. | 0x0 |
| 4 | R/W | LCDC_MIPI_OFF | MIPI off. (SPI mode of MIPI standard) 0: disabled 1: enabled | 0x0 |
| 3 | R/W | LCDC_FORM_OFF | Formating off 0: disabled 1: enabled | 0x0 |
| 2 | R/W | - | | 0x0 |
| 1 | R/W | LCDC_DSCAN | Double horizontal scan 0: disabled 1: enabled | 0x0 |
| 0 | R/W | LCDC_TMODE | Test mode 0: disabled 1: enabled | 0x0 |

Table 162: LCDC_CLKCTRL_REG (0x30030004)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 31:27 | R/W | LCDC_SEC_CLK_DIV | Secondary clock divider that generates the format pipeline clock. Source clock of this divider is the main clock of LCD controller. The period of the generated clock is defined as : $(LCDC_SEC_CLK_DIV + 1) \times \text{period_of_main_clock}$. | 0x0 |
| 26:14 | R/W | - | | 0x0 |
| 13:8 | R/W | LCDC_DMA_HOLD | Hold time before DMA activated. | 0x4 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| 7:6 | R/W | - | | 0x0 |
| 5:0 | R/W | LCDC_CLK_DIV | Clock divider that generates the pixel pipeline clock. Source clock of this divider is the format pipeline clock (see also LCDC_SEC_CLK_DIV). The period of the generated clock is defines as : LCDC_CLK_DIV x period_of_format_clk. A zero value gives division by one. | 0x1 |

Table 163: LCDC_BGCOLOR_REG (0x30030008)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---------------------------------|-------|
| 31:24 | R/W | LCDC_BG_RED | Red color used as background. | 0x0 |
| 23:16 | R/W | LCDC_BG_GREEN | Green color used as background. | 0x0 |
| 15:8 | R/W | LCDC_BG_BLUE | Blue color used as background. | 0x0 |
| 7:0 | R/W | LCDC_BG_ALPHA | Alpha color used as background. | 0x0 |

Table 164: LCDC_RESXY_REG (0x3003000C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|-------------------------|-------|
| 31:16 | R/W | LCDC_RES_X | Resolution X in pixels. | 0x0 |
| 15:0 | R/W | LCDC_RES_Y | Resolution Y in pixels. | 0x0 |

Table 165: LCDC_FRONTPORCHXY_REG (0x30030014)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|------------------------------|-------|
| 31:16 | R/W | LCDC_FPORCH_X | Front porch X (lines) | 0x0 |
| 15:0 | R/W | LCDC_FPORCH_Y | Front porch Y (pixel clocks) | 0x0 |

Table 166: LCDC_BLANKINGXY_REG (0x30030018)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 31:16 | R/W | LCDC_BLANKING_X | Blanking period X (VSYNC lines) | 0x0 |
| 15:0 | R/W | LCDC_BLANKING_Y | Blanking period Y (HSYNC pulse length) | 0x0 |

Table 167: LCDC_BACKPORCHXY_REG (0x3003001C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|-----------------------------|-------|
| 31:16 | R/W | LCDC_BPORCH_X | Back porch X (lines) | 0x0 |
| 15:0 | R/W | LCDC_BPORCH_Y | Back porch Y (pixel clocks) | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 168: LCDC_DBIB_CFG_REG (0x30030028)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|--|-------|
| 31 | R/W | LCDC_DBIB_TE_DIS | Disable the sampling of the tearing effect signal, which is provided by the LCD device. 0: the tearing effect signal is sampled 1: the tearing effect signal is not sampled. | 0x0 |
| 30 | R/W | LCDC_DBIB_CSX_FORCE | Forces the DBIB CSX value. When is enabled the DBIB CSX takes the value of the LCDC_DBIB_CSX_FORCE_VAL. 0 : disable 1 : enable | 0x0 |
| 29 | R/W | LCDC_DBIB_CSX_FORCE_VAL | Value of DBIB CSX to be forced, if bit 30 is set. Defines also the active level of the DBIB CSX even if the bit 30 is not set. | 0x0 |
| 28 | R/W | LCDC_DBIB_SPI_PAD | Data padding : 0 : disable 1 : enable | 0x0 |
| 27:26 | R/W | - | | 0x0 |
| 25 | R/W | LCDC_DBIB_RESX | DBIB RESX, reset signal for MIPI DBIB display. | 0x0 |
| 24 | R/W | LCDC_DBIB_DMA_EN | Send pixels from DMA to DBIB display. 0 : disable 1 : enable | 0x0 |
| 23 | R/W | LCDC_DBIB_SPI3_EN | Enable SPI3 interface. 0 : disable 1 : enable | 0x0 |
| 22 | R/W | LCDC_DBIB_SPI4_EN | Enable SPI4 interface. 0 : disable 1 : enable | 0x0 |
| 21 | R/W | - | | 0x0 |
| 20 | R/W | LCDC_DBIB_SPI_CPHA | Sets the data phase for the SPI interface | 0x0 |
| 19 | R/W | LCDC_DBIB_SPI_CPOL | Sets the polarity of the clock (SCL) | 0x0 |
| 18 | R/W | LCDC_DBIB_SPI_JDI | Enables the line addressing between the horizontal lines (JDI SPI output format). 0 : disable 1 : enable | 0x0 |
| 17 | R/W | LCDC_DBIB_SPI_HOLD | Enables the command HOLD mode of operation. Commands and data transmissions binding. 0 : disable 1 : enable | 0x0 |
| 16 | R/W | LCDC_DBIB_SPI_INV_ADDR | Enables horizontal line address inversion. 0 : disable 1 : enable | 0x0 |
| 15 | R/W | LCDC_DBIB_INV_DATA | Data inversion 0 : disable 1 : enable | 0x0 |
| 14 | R/W | LCDC_DBIB_JDI_IN | MSB-LSB bit selection for JDI parallel interface | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------------|---|-------|
| | | V_PIX | 0 : disable (MSB - LSB) 1 : enable (LSB -MSB) | |
| 13 | R/W | LCDC_DBIB_JDI_S OFT_RST | JDI timing generation soft reset. 0 : disable 1 : enable | 0x0 |
| 12:5 | R/W | - | | 0x0 |
| 4:0 | R/W | LCDC_DBIB_FMT | Defines the output format and depends of the type of the output interface. For the SPI3/SPI4 are supported the following formats: 0x06 : RGB111-1 {2b00, R(n), G(n), B(n), R(n+1), G(n+1), B(n+1)} 0x07 : RGB111-2 {R(n), G(n), B(n), 1b0, R(n+1), G(n+1), B(n+1), 1b0} 0x08 : RGB111-3 {R(n), G(n), B(n), R(n+1), G(n+1), B(n+1), R(n+2), G(n+2), B(n+2), ... } 0x09 : RGB111-4 {D(n), D(n+1), D(n+2), ... } 0x10 : RGB332 0x11 : RGB444 0x12 : RGB565 0x13 : RGB666 0x14 : RGB888 For the JDI parallel interface should be used the format: 0xA : RGB222 | 0x0 |

Table 169: **LCDC_GPIO_REG** (0x3003002C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 31:2 | - | - | | 0x0 |
| 1 | R/W | LCDC_TE_INV | Applies an inversion on the TE (tearing effect) signal. 0 : the inversion is not applied on the TE signal 1 : the inversion is applied on TE signal | 0x0 |
| 0 | R/W | LCDC_PARIF_SEL | Selection of the parallel interface type that is forwarded to the gpio pins. 0 : JDI interface signals 1 : Clasic parallel interface | 0x0 |

Table 170: **LCDC_LAYER0_MODE_REG** (0x30030030)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31 | R/W | LCDC_L0_EN | Enable layer. 0 : disable 1 : enable | 0x0 |
| 30:5 | R/W | - | | 0x0 |
| 4:0 | R/W | LCDC_L0_COLOUR | Colour Mode: | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | _MODE | 00001: 16-bit RGBX5551 color format, 00010: 32-bit RGBX8888 color format, 00100: 8-bit RGB332 color format, 00101: 16-bit RGB565 color format, 00110: 32-bit XRGB8888, 00111: L8 Grayscale/Palette format, 01000: L1 Grayscale/Palette format, 01001: L4 Grayscale/Palette format, 01101: ABGR8888, 01110: BGRA8888 | |

Table 171: LCDC_LAYER0_STARTXY_REG (0x30030034)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|-------------------------|-------|
| 31:16 | R/W | LCDL_L0_START_X | Start X (offset pixels) | 0x0 |
| 15:0 | R/W | LCDL_L0_START_Y | Start Y (offset pixels) | 0x0 |

Table 172: LCDC_LAYER0_SIZEXY_REG (0x30030038)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|----------------------------------|-------|
| 31:16 | R/W | LCDL_L0_SIZE_X | Size X (Size of layer in pixels) | 0x0 |
| 15:0 | R/W | LCDL_L0_SIZE_Y | Size Y (Size of layer in pixels) | 0x0 |

Table 173: LCDC_LAYER0_BASEADDR_REG (0x3003003C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|----------------------------------|-------|
| 31:0 | R/W | LCDL_L0_FB_ADD_R | Base Address of the frame buffer | 0x0 |

Table 174: LCDC_LAYER0_STRIDE_REG (0x30030040)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| 31:21 | R/W | - | | 0x0 |
| 20:19 | R/W | LCDL_L0_FIFO_TH_R | Layer dma fifo threshold burst start 00: half fifo (default) 01: 2 burst size 10: 4 burst size 11: 8 burst size | 0x0 |
| 18:16 | R/W | LCDL_L0_BURST_L_EN | Layer burst length 000: 16-beats (default) 001: 2-beats 010: 4-beats 011: 8-beats 100: 16-beats | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | LCDC_L0_STRIDE | Layer Stride (distance from line to line in bytes) | 0x0 |

Table 175: LCDC_LAYER0_RESXY_REG (0x30030044)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|--|-------|
| 31:16 | R/W | LCDC_L0_RES_X | Resolution X (Resolution of layer in pixels) | 0x0 |
| 15:0 | R/W | LCDC_L0_RES_Y | Resolution Y (Resolution of layer in pixels) | 0x0 |

Table 176: LCDC_JDI_RESXY_REG (0x30030090)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|---|-------|
| 31:16 | R/W | LCDC_JDI_RES_X | Number of horizontal transfers. Should be equal to the half of the horizontal resolution (in pixels). | 0x0 |
| 15:0 | R/W | LCDC_JDI_RES_Y | Number of vertical transfers. Should be equal to the double of the vertical resolution (in lines). | 0x0 |

Table 177: LCDC_JDI_FBX_BLANKING_REG (0x30030094)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|---|-------|
| 31:16 | R/W | LCDC_JDI_FXBLANKING | Horizontal front blanking as a number of hck half periods | 0x0 |
| 15:0 | R/W | LCDC_JDI_BXBLANKING | Horizontal back blanking as a number of hck half periods | 0x0 |

Table 178: LCDC_JDI_FBY_BLANKING_REG (0x30030098)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|---|-------|
| 31:16 | R/W | LCDC_JDI_FYBLANKING | Vertical front blanking as a number of vck half periods | 0x0 |
| 15:0 | R/W | LCDC_JDI_BYBLANKING | Vertical back blanking as a number of vck half periods | 0x0 |

Table 179: LCDC_JDI_HCK_WIDTH_REG (0x3003009C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|---|-------|
| 31:0 | R/W | LCDC_JDI_HCK_WIDTH | Number of format pipeline clock cycles that define the half period of the HCK (high and low width). The minimum allowed value is 2. | 0x0 |

Table 180: LCDC_JDI_XRST_WIDTH_REG (0x300300A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|--|-------|
| 31:0 | R/W | LCDC_JDI_XRST_WIDTH | Number of format pipeline clock cycles of XRST width | 0x0 |

Table 181: LCDC_JDI_VST_DELAY_REG (0x300300A4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|---|-------|
| 31:0 | R/W | LCDC_JDI_VST_DELAY | XRST-to-VST delay in format pipeline clock cycles | 0x0 |

Table 182: LCDC_JDI_VST_WIDTH_REG (0x300300A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|---|-------|
| 31:0 | R/W | LCDC_JDI_VST_WIDTH | VST width in format pipeline clock cycles | 0x0 |

Table 183: LCDC_JDI_VCK_DELAY_REG (0x300300AC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|---|-------|
| 31:0 | R/W | LCDC_JDI_VCK_DELAY | XRST-to-VCK delay in format pipeline clock cycles | 0x0 |

Table 184: LCDC_JDI_HST_DELAY_REG (0x300300B0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|--|-------|
| 31:0 | R/W | LCDC_JDI_HST_DELAY | VCK-to-HST delay in format pipeline clock cycles | 0x0 |

Table 185: LCDC_JDI_HST_WIDTH_REG (0x300300B4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|---|-------|
| 31:0 | R/W | LCDC_JDI_HST_WIDTH | HST width in format pipeline clock cycles | 0x0 |

Table 186: LCDC_JDI_ENB_START_HLINE_REG (0x300300B8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------------|--|-------|
| 31:0 | R/W | LCDC_JDI_ENB_START_HLINE | The number of the first horizontal line where the ENB signal is asserted | 0x0 |

Table 187: LCDC_JDI_ENB_END_HLINE_REG (0x300300BC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------------|---|-------|
| 31:0 | R/W | LCDC_JDI_ENB_END_HLINE | The number of the last horizontal line where the ENB signal is asserted | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 188: LCDC_JDI_ENB_START_CLK_REG (0x300300C0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------------|--|-------|
| 31:0 | R/W | LCDC_JDI_ENB_START_CLK | Defines the number of the HCK half periods that should take place after a transition in the VCK and before to be enabled of the ENB. | 0x0 |

Table 189: LCDC_JDI_ENB_WIDTH_CLK_REG (0x300300C4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------------|--------------------------------------|-------|
| 31:0 | R/W | LCDC_JDI_ENB_WIDTH_CLK | ENB (high) width in HCK half periods | 0x0 |

Table 190: LCDC_DBIB_CMD_REG (0x300300E8)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|---|-------|
| 31 | R/W | - | | 0x0 |
| 30 | R/W | LCDC_DBIB_CMD_SEND | Send command to the DBI interface | 0x0 |
| 29 | R/W | - | | 0x0 |
| 28 | R/W | - | | 0x0 |
| 27 | R/W | LCDC_DBIB_CMD_STORE | This bit has meaning only when LCDC_DBIB_CFG_REG[LCDC_DBIB_SPI_JDI] = 1. When is enabled, stores the LCDC_DBIB_CMD_VAL to the register that keeps the Y position. | 0x0 |
| 26:16 | R/W | - | | 0x0 |
| 15:0 | R/W | LCDC_DBIB_CMD_VAL | Data to send to the DBI interface | 0x0 |

Table 191: LCDC_IDREG_REG (0x300300F4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|-------------------------|----------------|
| 31:0 | R | LCDC_ID | Identification register | 0x87452 365 |

Table 192: LCDC_INTERRUPT_REG (0x300300F8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|--|-------|
| 31 | R/W | LCDC_IRQ_TRIGGER_SEL | IRQ trigger control 0: Level triggering 1: Edge triggering In the case of the level triggering, the request remains active in the LCDC until to be cleared. The request can be cleared by performing a write access in the LCDC_INTERRUPT_REG. This is not required in the case of the edge triggering. | 0x0 |
| 30:6 | - | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------|--|-------|
| 5 | R/W | LCDC_FRAME_END_IRQ_EN | Continuous mode: frame end. Single mode: frame end or idle. | 0x0 |
| 4 | R/W | - | | 0x0 |
| 3 | R/W | LCDC_TE_IRQ_EN | TE interrupt enable. See also the configuration bit LCDC_DBIB_CFG_REG[LCDC_DBIB_TE_DIS] | 0x0 |
| 2 | R/W | - | | 0x0 |
| 1 | R/W | LCDC_HSYNC_IRQ_EN | Hsync interrupt enabled | 0x0 |
| 0 | R/W | LCDC_VSYNC_IRQ_EN | Vsync or TE interrupt enabled. See also the configuration bit LCDC_DBIB_CFG_REG[LCDC_DBIB_TE_DIS] for the TE signal. | 0x1 |

Table 193: LCDC_STATUS_REG (0x300300FC)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------------|--|-------|
| 31:16 | - | - | | 0x0 |
| 15 | R | LCDC_JDI_TIM_SW_RST | JDI timing generation soft reset (active high) | 0x0 |
| 14 | R | LCDC_FRAME_START | Frame start (active high) | 0x0 |
| 13 | R | LCDC_FRAME_END | Frame end (active high) | 0x0 |
| 12 | R | LCDC_DBIB_CMD_PENDING | Transferring of command in progress. 0: idle 1: in progress | 0x0 |
| 11 | R | LCDC_DBIB_CMD_FIFO_FULL | Command fifo full indication. 0: is not full 1: is full | 0x0 |
| 10 | R | LCDC_DBIB_CMD_FIFO_EMPTY_N | Command fifo empty indication (negative) 0: the fifo is empty 1: the fifo is not empty | 0x0 |
| 9 | - | - | | 0x0 |
| 8 | R | LCDC_DBIB_TE | The DBIB tearing effect signal | 0x0 |
| 7 | R | LCDC_STICKY_UNDERFLOW | Sticky underflow (clear with write in the LCDC_INTERRUPT_REG) 0: There is no underflow 1: Underflow has been detected. Remains high until to be cleared by performing a write access on the register LCDC_INTERRUPT_REG. | 0x0 |
| 6 | R | LCDC_UNDERFLOW | Underflow on the current transfer. 0: There is no underflow 1: Underflow has been detected. | 0x0 |
| 5 | R | LCDC_LAST_ROW | Last row (Last row is currently displayed) | 0x0 |
| 4 | R | LCDC_STAT_CSYN_C | Csync signal level | 0x0 |
| 3 | R | LCDC_STAT_VSYN | Vsync signal level | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|--|-------|
| | | C | | |
| 2 | R | LCDC_STAT_HSYN_C | Hsync signal level | 0x0 |
| 1 | R | LCDC_FRAMEGEN_BUSY | The frame generator is busy (active high). | 0x0 |
| 0 | R | LCDC_STAT_ACTIVE | Active (When not in vertical blanking) | 0x0 |

Table 194: LCDC_CRC_REG (0x30030184)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|-------------|-------|
| 31:0 | R | LCDC_CRC | CRC check. | 0x0 |

Table 195: LCDC_LAYER0_OFFSETX_REG (0x30030188)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------|--|-------|
| 31:16 | R/W | LCDC_L0_DMA_PREFETCH | DMA fifo prefetch level (range: 0-4) 0x0 : Prefetch mechanism is disabled 0x1 : Prefetch at least 44 bytes 0x2 : Prefetch at least 84 bytes 0x3 : Prefetch at least 116 bytes 0x4 : Prefetch at least 108 bytes Any other value : Reserved | 0x0 |
| 15:0 | R/W | LCDC_L0_OFFSET_X | Offset X (negative) of X start pixel (range [n-1,0], n : pixels /8) | 0x0 |

42.3 LRA/ERM Registers

Table 196: Register map LRA

| Address | Register | Description |
|------------|------------------|--------------------------------------|
| 0x50030A00 | LRA_CTRL1_REG | General Purpose LRA Control Register |
| 0x50030A04 | LRA_CTRL2_REG | General Purpose LRA Control Register |
| 0x50030A08 | LRA_CTRL3_REG | General Purpose LRA Control Register |
| 0x50030A0C | LRA_FLT_SMP1_REG | LRA Sample Register |
| 0x50030A10 | LRA_FLT_SMP2_REG | LRA Sample Register |
| 0x50030A14 | LRA_FLT_SMP3_REG | LRA Sample Register |
| 0x50030A18 | LRA_FLT_SMP4_REG | LRA Sample Register |
| 0x50030A1C | LRA_FLT_SMP5_REG | LRA Sample Register |
| 0x50030A20 | LRA_FLT_SMP6_REG | LRA Sample Register |
| 0x50030A24 | LRA_FLT_SMP7_REG | LRA Sample Register |
| 0x50030A28 | LRA_FLT_SMP8_REG | LRA Sample Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------|--------------------------------------|
| 0x50030A2C | LRA_FLT_COEF1_REG | LRA Filter Coefficient Register |
| 0x50030A30 | LRA_FLT_COEF2_REG | LRA Filter Coefficient Register |
| 0x50030A34 | LRA_FLT_COEF3_REG | LRA Filter Coefficient Register |
| 0x50030A38 | LRA_BRD_LS_REG | LRA Bridge Register |
| 0x50030A3C | LRA_BRD_HS_REG | LRA Bridge Register |
| 0x50030A40 | LRA_BRD_STAT_REG | LRA Bridge Status Register |
| 0x50030A44 | LRA_ADC_CTRL1_REG | General Purpose ADC Control Register |
| 0x50030A50 | LRA_ADC_RESULT_REG | General Purpose ADC Result Register |
| 0x50030A54 | LRA_LDO_REG | LRA LDO Register |
| 0x50030A58 | LRA_DFT_REG | LRA test Register |

Table 197: LRA_CTRL1_REG (0x50030A00)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 31:28 | - | - | | 0x0 |
| 27:24 | R | SMP_IDX | Current bin index (0-15). Check if equal to IRQ_IDX before and/or after updating HALF_PERIOD with ISR. | 0x0 |
| 23:19 | - | - | | 0x0 |
| 18 | R/W | IRQ_SCP_EVENT_EN | 0 = interrupt scp event disabled 1 = interrupt scp event enabled | 0x0 |
| 17 | R/W | IRQ_ADC_EN | 0 = interrupt adc disabled 1 = interrupt adc enabled | 0x0 |
| 16 | R/W | IRQ_CTRL_EN | 0 = interrupt controller disabled 1 = interrupt controller enabled | 0x0 |
| 15:12 | R/W | IRQ_IDX | At which sample index an IRQ will be generated (0-15). When IRQ_IDX < 8, IRQs are generated at both half cycles (IRQ_IDX and IRQ_IDX+8), otherwise only in the second half cycle. | 0x7 |
| 11:8 | R/W | IRQ_DIV | Divider value of the interrupt request. Number of LRA/ERM periods, between successive IRQs. 0,1=every (half) cycle, depending on IRQ_IDX; 2=every second cycle, IRQ at the end of first or both half cycles (based on IRQ_IDX), etc. | 0x1 |
| 7:6 | R/W | SMP_SEL | Select which samples to store for the resonance control algorithm. 0=Sense voltage after down-sampling 1=Error voltage (after subtraction of VREF and down-sampled sense voltage input) 2=Duty cycle signal after loop-filter 3=Duty cycle signal after summation with DREF | 0x3 |
| 5 | R/W | PULLDOWN_EN | LXP and LXN node pull down enable, when | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|---|-------|
| | | | SC_EVENT=0 && LOOP_EN=0 | |
| 4 | R/W | LOOP_EN | 0=disable loop 1=enable loop | 0x0 |
| 3 | R/W | LDO_EN | 0=lra ldo disabled 1=lra ldo enabled | 0x0 |
| 2 | R/W | ADC_EN | 0=lra adc disabled 1=lra adc enabled | 0x0 |
| 1 | R/W | HBRIDGE_EN | 0=hbridge disabled 1=hbridge enabled | 0x0 |
| 0 | R/W | LRA_EN | 0=lra controller disabled 1=lra controller enabled | 0x0 |

Table 198: LRA_CTRL2_REG (0x50030A04)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 31:16 | R/W | HALF_PERIOD | Half of the LRA period, in units of 4 μ s (= 125 kHz divided by the resonant frequency of the LRA). | 0x271 |
| 15:6 | - | - | | 0x0 |
| 5 | R/W | AUTO_MODE | Automatic frequency control (0=disabled;, 1=enabled, not yet implemented) | 0x0 |
| 4 | R/W | SMP_MODE | Sampling mode for data aiding automatic resonance control (0=averaging, 1=last sample) | 0x0 |
| 3 | R/W | POLARITY | Polarity of the square wave (0=normal; 1=inverted); Use for rapid stop. | 0x0 |
| 2 | R/W | FLT_IN_SEL | 0 = normal operation 1 = ADC output overruled by register field MAN_FLT_IN | 0x0 |
| 1:0 | R/W | PWM_MODE | PWM pulse placement: 0=middle, 1=left, 2=right, 3=alternate | 0x0 |

Table 199: LRA_CTRL3_REG (0x50030A08)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|--|--------|
| 31:16 | R/W | VREF | Voltage reference for haptic feedback driver (1 bit sign + 7 bits integer + 8 bits fractional. (range -128 ... +127.99). The voltage is computed based on the desired reference current (IREF) that flows into the haptic device. $VREF = IREF \times \text{TRIM_GAIN} \times \text{AGC_GAIN} \times 128 / 2.4$, with IREF in Amperes, $\text{TRIM_GAIN} = 6, 8 (\text{default}), 10 \text{ or } 12 \text{ as defined in LRA_BRD_HS_REG}$, $\text{AGC_GAIN is the normalized gain of the ADC (i.e. } \text{AGC_GAIN} = \text{GP_ADC_VALUE} \times 300 \text{ mA}/[\text{l}_{\text{LRA}} \times 128])$. | 0x0 |
| 15:0 | R/W | DREF | Duty cycle reference, start value from which the | 0x4000 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | current control loop settles (1 sign bit, 7 integer bits, 8 fractional bits, -128 ... +127.99). Valid settings 5% through 95% (0x0666 - 0x799A). | |

Table 200: LRA_FLT_SMP1_REG (0x50030A0C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---|-------|
| 31:16 | R | LRA_SMP_2 | Second sample in first half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_1 | First sample in first half-cycle used for resonance control algorithm. | 0x0 |

Table 201: LRA_FLT_SMP2_REG (0x50030A10)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---|-------|
| 31:16 | R | LRA_SMP_4 | Fourth sample in first half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_3 | Third sample in first half-cycle used for resonance control algorithm. | 0x0 |

Table 202: LRA_FLT_SMP3_REG (0x50030A14)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|--|-------|
| 31:16 | R | LRA_SMP_6 | Sixth sample in first half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_5 | Fifth sample in first half-cycle used for resonance control algorithm. | 0x0 |

Table 203: LRA_FLT_SMP4_REG (0x50030A18)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|--|-------|
| 31:16 | R | LRA_SMP_8 | Eighth sample in first half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_7 | Seventh sample in first half-cycle used for resonance control algorithm. | 0x0 |

Table 204: LRA_FLT_SMP5_REG (0x50030A1C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 31:16 | R | LRA_SMP_10 | Second sample in second half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_9 | First sample in second half-cycle used for resonance control algorithm. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 205: LRA_FLT_SMP6_REG (0x50030A20)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 31:16 | R | LRA_SMP_12 | Fourth sample in second half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_11 | Third sample in second half-cycle used for resonance control algorithm. | 0x0 |

Table 206: LRA_FLT_SMP7_REG (0x50030A24)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:16 | R | LRA_SMP_14 | Sixth sample in second half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_13 | Fifth sample in second half-cycle used for resonance control algorithm. | 0x0 |

Table 207: LRA_FLT_SMP8_REG (0x50030A28)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:16 | R | LRA_SMP_16 | Eighth sample in second half-cycle used for resonance control algorithm. | 0x0 |
| 15:0 | R | LRA_SMP_15 | Seventh sample in second half-cycle used for resonance control algorithm. | 0x0 |

Table 208: LRA_FLT_COEF1_REG (0x50030A2C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 31:16 | R/W | FLT_COEF_01 | Loop filter state-space coefficient a12 (1 sign bit, 1 integer bit, 14 fractional bits, range -2.000 .. +1.999). | 0x19A |
| 15:0 | R/W | FLT_COEF_00 | Loop filter state-space coefficient a11 (1 sign bit, 1 integer bit, 14 fractional bits, range -2.000 .. +1.999). | 0xE66 |

Table 209: LRA_FLT_COEF2_REG (0x50030A30)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 31:16 | R/W | FLT_COEF_10 | Loop filter state-space coefficient a21 (1 sign bit, 1 integer bit, 14 fractional bits, range -2.000 .. +1.999). | 0xE66 |
| 15:0 | R/W | FLT_COEF_02 | Loop filter state-space coefficient b1 (1 sign bit, 1 integer bit, 14 fractional bits, range -2.000 .. +1.999). Note: For correct intended loop gain, modify the intended value of b1 to b1/ADC_GAIN, where ADC_GAIN is the normalized gain of the ADC (i.e. ADC_GAIN = GP_ADC_VALUE×300 mA/[I _{LRA} ×128]). | 0xF3E |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 210: LRA_FLT_COEF3_REG (0x50030A34)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|--------|
| 31:16 | R/W | FLT_COEF_12 | Loop filter state-space coefficient b2 (1 sign bit, 1 integer bit, 14 fractional bits, range -2.000 .. +1.999). Note: For correct intended loop gain, modify the intended value of b1 to b1/ADC_GAIN, where ADC_GAIN is the normalized gain of the ADC (i.e. ADC_GAIN = GP_ADC_VALUE×300 mA/[I _{LRA} ×128]). | 0x16CE |
| 15:0 | R/W | FLT_COEF_11 | Loop filter state-space coefficient a22 (1 sign bit, 1 integer bit, 14 fractional bits, range -2.000 .. +1.999). | 0x19A |

Table 211: LRA_BRD_LS_REG (0x50030A38)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 14 | - | - | | 0x0 |
| 11:8 | R/W | SCP_LS_TRIM_N | LSN short-circuit protection limit trimming | 0x0 |
| 7:4 | R/W | SCP_LS_TRIM_P | LSP short-circuit protection limit trimming | 0x0 |
| 3 | R/W | SCP_LS_EN | LS short-circuit protection enable | 0x0 |
| 2:1 | R/W | ERC_LS_TRIM | LS edge-rate control trimming. High-to-Low switching slewing: 00: 25 MV/s 01: 50 MV/s 10: 75 MV/s 11: 100 MV/s | 0x0 |
| 0 | R/W | ERC_LS_EN | LS edge-rate control enable | 0x0 |

Table 212: LRA_BRD_HS_REG (0x50030A3C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 31:15 | - | - | | 0x0 |
| 14:11 | R/W | TRIM_GAIN | Current-sensing amplifier gain settings: 0001: x6 0010: x8 0100: x10 1000: x12 | 0x0 |
| 10:8 | R/W | HSGND_TRIM | HS gnd trim, default at 100 000: 2.2V and 111:3.6V with 0.2V per step | 0x0 |
| 7:4 | R/W | SCP_HS_TRIM | HS short-circuit protection limit trimming | 0x0 |
| 3 | R/W | SCP_HS_EN | HS short-circuit protection enable | 0x0 |
| 2:1 | R/W | ERC_HS_TRIM | HS edge-rate control trimming. Lowto-High switching slewing: 00: 25 MV/s 01: 50 MV/s 10: 75 MV/s | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|-----------------------------|-------|
| | | | 11: 100 MV/s | |
| 0 | R/W | ERC_HS_EN | HS edge-rate control enable | 0x0 |

Table 213: LRA_BRD_STAT_REG (0x50030A40)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:14 | - | - | | 0x0 |
| 13 | R | SCP_HS_OUT | HS short circuit comparator output | 0x0 |
| 12 | R | SCP_LS_COMP_OUT_N | LSN short circuit comparator output | 0x0 |
| 11 | R | SCP_LS_COMP_OUT_P | LSP short circuit comparator output | 0x0 |
| 10 | R | SC_EVENT_LS | 1: LS short-circuit event detected 0: no LS short-circuit event detected | 0x0 |
| 9 | R | SC_EVENT_HS | 1: HS short-circuit event detected 0: no HS short-circuit event detected | 0x0 |
| 8 | R | LOOP_STAT | 1: Loop saturation detected 0: Loop not saturated | 0x0 |
| 7 | R | LSN_ON | LSN control status | 0x0 |
| 6 | R | LSP_ON | LSP control status | 0x0 |
| 5 | R | HSN_ON | HSN control status | 0x0 |
| 4 | R | HSP_ON | HSP control status | 0x0 |
| 3 | R | LSN_STAT | LSN power FET gate actual status | 0x0 |
| 2 | R | LSP_STAT | LSP power FET gate actual status | 0x0 |
| 1 | R | HSN_STAT | HSN power FET gate actual status | 0x0 |
| 0 | R | HSP_STAT | HSP power FET gate actual status | 0x0 |

Table 214: LRA_ADC_CTRL1_REG (0x50030A44)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|--|-------|
| 31 | R | LRA_ADC_BUSY | 0:ADC conversion ready. 1:ADC conversion in progress. | 0x0 |
| 30:17 | - | - | | 0x0 |
| 16:9 | R/W | LRA_ADC_OFFSET | ADC offset compensation value. Signed value with 3 fractional bits. -16 (0x80) to +15.875 (0x7F) in intervals of 0.125 (0x01). Note: ADC gain error must be compensated in the calculation of VREF. | 0x0 |
| 8 | R/W | LRA_ADC_TEST_PARAM | Select which inputs will be enabled on the ADC. 0,1 = normal inputs (i.e. both I and Q inputs connected to LRA-current-sense voltage source) 2 = I channel connected to the analog input testbus on PORTS P14 and P15, Q channel is muted. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|--|-------|
| | | | 3 =Q channel connected to the analog input testbus on PORTS P14 and P15, I channel is muted. Note: The LRA_ADC_CTRL1_REG[ADC_MUTE] field takes precedence over this test functionality. | |
| 7 | R/W | LRA_ADC_TEST_IN_SEL | Select analog testbus on ADC input. | 0x0 |
| 6:3 | R/W | LRA_ADC_FREQ | ADC clock divider | 0x4 |
| 2 | R/W | LRA_ADC_SIGN | Change polarity of ADC input | 0x0 |
| 1 | R/W | LRA_ADC_MUTE | 0: Normal operation 1: Short the inputs of the ADC (used for DC offset cal) | 0x0 |
| 0 | R0/W | LRA_ADC_START | If a 1 is written the ADC conversion will start | 0x0 |

Table 215: LRA_ADC_RESULT_REG (0x50030A50)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|--------|
| 31:16 | R/W | MAN_FLT_IN | Manual value to replace the ADC output. Select its use by FLT_IN_SEL. | 0x0 |
| 15:0 | R | GP_ADC_VAL | Returns the 10 up to 16 bits linear value of the last AD conversion as a signed value. The most significant 11 bits are always valid, the lower 5 bits are only valid in case oversampling has been applied. Two samples results in one extra bit and 32 samples results in 5 extra bits. In the context of the LRA constant current or constant duty cycle control systems, the (non-oversampled) value is interpreted as a signed value with 7 integer bits and 3 fractional bits: -128.000 (0x8000) to +127.875 (0x7FE0) in steps of 0.125 (0x0010). Note that the measured values in this context are always positive. | 0x4000 |

Table 216: LRA_LDO_REG (0x50030A54)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| 31 | R | LDO_OK | 0: LDO not yet ok 1: LDO voltage is ready | 0x0 |
| 1 | R/W | LDO_TST | When set to 1, LDO output is connected to the testbus through a test switch | 0x0 |
| 0 | R/W | LDO_VREF_HOLD | 0: Indicates that the reference input is tracked, 1: Indicates that the reference input is sampled | 0x0 |

Table 217: LRA_DFT_REG (0x50030A58)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|---|-------|
| 31:29 | R/W | SPARE | spare registers bits , currently not used | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 28 | R/W | SWM_SEL | 0=use SWM from controller 1=use SWM_MAN | 0x0 |
| 27 | R/W | SWM_MAN | swm manual | 0x0 |
| 26 | R/W | PWM_SEL | 0=use PWM from controller 1=use PWM_MAN | 0x0 |
| 25 | R/W | PWM_MAN | pwm manual | 0x0 |
| 24:23 | R/W | TIMER_TRIM | 20ns unit delay cell trimming bits | 0x0 |
| 22:21 | R/W | TIMER_SCALE_TRIM | Selection of delay of MAG and DEMAG signal: 00: 60ns 01: 80ns 10: 100ns 11: 120ns | 0x0 |
| 20 | R/W | DFT_SEL | not used | 0x0 |
| 19 | R/W | DFT_FORCE_HSPN | Force HSP and HSN power FETs on: 0: not activated 1: HSP and HSN are forced on | 0x0 |
| 18 | R/W | DFT_EN_TIMER | Enable for the timer trimming | 0x0 |
| 17:16 | R/W | DFT_STALL | Force state machine in a certain state: 00: No test 01: High-Z 10: Mag 11: Demag | 0x0 |
| 15:0 | R/W | DFT_CTRL | Selection of test bus connection | 0x0 |

42.4 Memory Controller Registers

Table 218: Register map MEMCTRL

| Address | Register | Description |
|------------|---------------------|--|
| 0x50050004 | MEM_PRIO_REG | Priority Control Register |
| 0x50050008 | MEM_STALL_REG | Maximum Stall cycles Control Register |
| 0x5005000C | MEM_STATUS_REG | Memory Arbiter Status Register |
| 0x50050010 | MEM_STATUS2_REG | RAM cells Status Register |
| 0x50050020 | CMI_CODE_BASE_REG | CMAC code Base Address Register |
| 0x50050024 | CMI_DATA_BASE_REG | CMAC data Base Address Register |
| 0x50050028 | CMI_SHARED_BASE_REG | CMAC shared data Base Address Register |
| 0x5005002C | CMI_END_REG | CMAC end Address Register |
| 0x50050030 | SNC_BASE_REG | Sensor Node Controller Base Address Register |
| 0x50050074 | BUSY_SET_REG | BSR Set Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|----------------|---------------------|
| 0x50050078 | BUSY_RESET_REG | BSR Reset Register |
| 0x5005007C | BUSY_STAT_REG | BSR Status Register |

Table 219: MEM_PRIO_REG (0x50050004)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| 5:4 | R/W | AHB_PRIO | Priority for the AHB interface. 00: low priority (default) 01: mid priority 1x: high priority | 0x0 |
| 3:2 | R/W | AHB2_PRIO | Priority for the AHB2 interface. 00: low priority (default) 01: mid priority 1x: high priority | 0x0 |
| 1:0 | R/W | SNC_PRIO | Priority for the SNC interface. 00: low priority (default) 01: mid priority 1x: high priority | 0x0 |

Table 220: MEM_STALL_REG (0x50050008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 11:8 | R/W | AHB_MAX_STALL | Maximum allowed number of stall cycles for the AHB interface. If exceeded, the interface will get top priority (above high priority). Valid for a single access so the next access (of a burst) might end up in the que for the same number of wait cycles. 0: don't use, not feasible and can block other interfaces 1: max 1 stall cycle 15: max 15 stall cycles | 0xF |
| 7:4 | R/W | AHB2_MAX_STALL | Maximum allowed number of stall cycles for the AHB2 interface. If exceeded, the interface will get top priority (above high priority). Valid for a single access so the next access (of a burst) might end up in the que for the same number of wait cycles. 0: don't use, not feasible and can block other interfaces 1: max 1 stall cycle 15: max 15 stall cycles | 0xF |
| 3:0 | R/W | SNC_MAX_STALL | Maximum allowed number of stall cycles for the SNC interface. If exceeded, the interface will get top priority (above high priority). Valid for a single access so the next access (of a burst) might end up in the que for the same number of wait cycles. 0: don't use, not feasible and can block other interfaces 1: max 1 stall cycle 15: max 15 stall cycles | 0xF |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 221: MEM_STATUS_REG (0x5005000C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 13 | W | CMI_CLEAR_READ_Y | Writing a '1' clears CMI_NOT_READY bit. | 0x0 |
| 12 | R | CMI_NOT_READY | 0: Normal operation 1: CMI access performed which couldn't be handled right away (interface doesn't allow wait cycles) | 0x0 |
| 11:8 | R | AHB2_WR_BUFF_CNT | The maximum number of arbiter clock cycles that an AHB2 access has been buffered. | 0x0 |
| 7:4 | R | AHB_WR_BUFF_CNT | The maximum number of arbiter clock cycles that an AHB access has been buffered. | 0x0 |
| 3 | W | AHB2_CLR_WR_BUFF | Writing a '1' clears AHB2_WR_BUFF_CNT. | 0x0 |
| 2 | W | AHB_CLR_WR_BUF_F | Writing a '1' clears AHB_WR_BUFF_CNT. | 0x0 |
| 1 | R | AHB2_WRITE_BUF_F | 0: No AHB2 write access is buffered. 1: Currently a single AHB2 write access is buffered in the arbiter. | 0x0 |
| 0 | R | AHB_WRITE_BUFF | 0: No AHB write access is buffered. 1: Currently a single AHB write access is buffered in the arbiter. | 0x0 |

Table 222: MEM_STATUS2_REG (0x50050010)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|--|-------|
| 7 | RW1C | RAM8_OFF_BUT_ACCESS | Reading a '1' indicates RAM8 was off but still access was performed. Writing a '1' will clear the status back to '0'. | 0x0 |
| 6 | RW1C | RAM7_OFF_BUT_ACCESS | Reading a '1' indicates RAM7 was off but still access was performed. Writing a '1' will clear the status back to '0'. | 0x0 |
| 5 | RW1C | RAM6_OFF_BUT_ACCESS | Reading a '1' indicates RAM6 was off but still access was performed. Writing a '1' will clear the status back to '0'. | 0x0 |
| 4 | RW1C | RAM5_OFF_BUT_ACCESS | Reading a '1' indicates RAM5 was off but still access was performed. Writing a '1' will clear the status back to '0'. | 0x0 |
| 3 | RW1C | RAM4_OFF_BUT_ACCESS | Reading a '1' indicates RAM4 was off but still access was performed. Writing a '1' will clear the status back to '0'. | 0x0 |
| 2 | RW1C | RAM3_OFF_BUT_ACCESS | Reading a '1' indicates RAM3 was off but still access was performed. Writing a '1' will clear the status back to '0'. | 0x0 |
| 1 | RW1C | RAM2_OFF_BUT_ACCESS | Reading a '1' indicates RAM2 was off but still access was performed. Writing a '1' will clear the status back to '0'. | 0x0 |
| 0 | RW1C | RAM1_OFF_BUT_ACCESS | Reading a '1' indicates RAM1 was off but still | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | CCESS | access was performed. Writing a '1' will clear the status back to '0'. | |

Table 223: CMI_CODE_BASE_REG (0x50050020)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| 18:10 | R/W | CMI_CODE_BASE_ADDR | Base address for CMAC code with steps of 1 kB. 0x001: 1 kB base address 0x010: 16 kB base address 0x100: 256 kB base address | 0x0 |
| 9:0 | R | - | | 0x0 |

Table 224: CMI_DATA_BASE_REG (0x50050024)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|---|-------|
| 18:2 | R/W | CMI_DATA_BASE_ADDR | Base address for CMAC data with steps of 4 bytes. 0x00001: 4 byte base address 0x00010: 64 byte base address 0x00100: 1 kB base address 0x01000: 16 kB base address 0x10000: 256 kB base address | 0x0 |
| 1:0 | R | - | | 0x0 |

Table 225: CMI_SHARED_BASE_REG (0x50050028)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------|--|-------|
| 18:10 | R/W | CMI_SHARED_BASE_ADDR | Base address for CMAC shared data with steps of 1 kB. 0x001: 1 kB base address 0x010: 16 kB base address 0x100: 256 kB base address | 0x0 |
| 9:0 | R | - | | 0x0 |

Table 226: CMI_END_REG (0x5005002C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|--|-------|
| 18:10 | R/W | CMI_END_ADDR | End address for CMAC code and data accesses with steps of 1 kB. 0x000: accesses up to 1kB are allowed 0x001: accesses up to 2kB are allowed 0x01F: accesses up to 32kB are allowed 0x1FF: accesses up to 512kB are allowed | 0xFF |
| 9:0 | R | - | | 0x3FF |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 227: SNC_BASE_REG (0x50050030)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 18:2 | R/W | SNC_BASE_ADDRESS | Base address for SNC interface with steps of 4 bytes. 0x00001: 4 byte base address 0x00010: 64 byte base address 0x00100: 1 kB base address 0x01000: 16 kB base address 0x10000: 256 kB base address | 0x0 |
| 1:0 | R | - | | 0x0 |

Table 228: BUSY_SET_REG (0x50050074)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 31:30 | WS | BUSY_SPARE | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 29:28 | WS | BUSY_MOTOR | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 27:26 | WS | BUSY_TIMER2 | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 25:24 | WS | BUSY_TIMER | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 23:22 | WS | BUSY_UART3 | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 21:20 | WS | BUSY_GPADC | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 19:18 | WS | BUSY_PDM | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 17:16 | WS | BUSY_SRC | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| | | | Reading returns 0 to allow read/modify/write to the register. | |
| 15:14 | WS | BUSY_PCM | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 13:12 | WS | BUSY_SDADC | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 11:10 | WS | BUSY_I2C2 | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 9:8 | WS | BUSY_I2C | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 7:6 | WS | BUSY_SPI2 | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 5:4 | WS | BUSY_SPI | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 3:2 | WS | BUSY_UART2 | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 1:0 | WS | BUSY_UART | Writing a non-zero value to this field sets the corresponding BUSY bit, but only if it was not claimed (BUSY=0). Reading returns 0 to allow read/modify/write to the register. | 0x0 |

Table 229: BUSY_RESET_REG (0x50050078)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 31:30 | RW1C | BUSY_SPARE | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 29:28 | RW1C | BUSY_MOTOR | Clear the BUSY bitfield, by writing the master code which has claimed to this field | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| | | | Reading returns 0 to allow read/modify/write to the register. | |
| 27:26 | RW1C | BUSY_TIMER2 | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 25:24 | RW1C | BUSY_TIMER | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 23:22 | RW1C | BUSY_UART3 | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 21:20 | RW1C | BUSY_GPADC | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 19:18 | RW1C | BUSY_PDM | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 17:16 | RW1C | BUSY_SRC | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 15:14 | RW1C | BUSY_PCM | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 13:12 | RW1C | BUSY_SDADC | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 11:10 | RW1C | BUSY_I2C2 | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 9:8 | RW1C | BUSY_I2C | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 7:6 | RW1C | BUSY_SPI2 | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 5:4 | RW1C | BUSY_SPI | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |
| 3:2 | RW1C | BUSY_UART2 | Clear the BUSY bitfield, by writing the master code which has claimed to this field | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | Reading returns 0 to allow read/modify/write to the register. | |
| 1:0 | RW1C | BUSY_UART | Clear the BUSY bitfield, by writing the master code which has claimed to this field Reading returns 0 to allow read/modify/write to the register. | 0x0 |

Table 230: BUSY_STAT_REG (0x5005007C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 31:30 | R | BUSY_SPARE | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 29:28 | R | BUSY_MOTOR | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 27:26 | R | BUSY_TIMER2 | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 25:24 | R | BUSY_TIMER | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 23:22 | R | BUSY_UART3 | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 21:20 | R | BUSY_GPADC | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 19:18 | R | BUSY_PDM | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 17:16 | R | BUSY_SRC | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 15:14 | R | BUSY_PCM | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 13:12 | R | BUSY_SDADC | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 11:10 | R | BUSY_I2C2 | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 9:8 | R | BUSY_I2C | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 7:6 | R | BUSY_SPI2 | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 5:4 | R | BUSY_SPI | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 3:2 | R | BUSY_UART2 | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |
| 1:0 | R | BUSY_UART | A non-zero value indicates the resource is busy. The value represents which master is using it. | 0x0 |

42.5 OTP Controller Registers

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 231: Register map OTPC

| Address | Register | Description |
|------------|----------------|--|
| 0x30070000 | OTPC_MODE_REG | Mode register |
| 0x30070004 | OTPC_STAT_REG | Status register |
| 0x30070008 | OTPC_PADDR_REG | The address of the word that will be programmed, when the PROG mode is used. |
| 0x3007000C | OTPC_PWORD_REG | The 32-bit word that will be programmed, when the PROG mode is used. |
| 0x30070010 | OTPC_TIM1_REG | Various timing parameters of the OTP cell. |
| 0x30070014 | OTPC_TIM2_REG | Various timing parameters of the OTP cell. |

Table 232: OTPC_MODE_REG (0x30070000)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:6 | 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 is programmed. 0x3 : Reserved The value of this configuration field can be modified only when the controller is in an inactive mode (PDOWN, DSTBY, STBY). The setting will take effect when will be enabled again the PROG mode. | 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 an inactive mode (PDOWN, DSTBY, STBY). The selection will take 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 The value of this configuration field can be modified only when the controller is in an inactive mode (PDOWN, DSTBY, STBY). The selection will take effect at the next programming or reading mode that will be enabled. | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R/W | OTPC_MODE_MOD | Defines the mode of operation of the OTPC | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | E | <p>controller. The encoding of the modes is as follows:</p> <ul style="list-style-type: none"> 0x0: PDOWN. The power supply of the OTP memory is OFF. 0x1: DSTBY. The OTP memory is in deep standby mode (power supply ON and internal LDO OFF). 0x2: STBY. The OTP memory is powered (power supply ON and internal LDO ON, but is not selected). 0x3: READ. The OTP memory is in the normal read mode. 0x4: PROG. The OTP memory is in programming mode. 0x5: PVFY. The OTP memory is in programming verification mode (margin read after programming). 0x6: RINI. The OTP memory is in initial read mode (initial margin read). 0x7: Reserved. <p>Whenever the OTPC_MODE_REG[MODE] is changing, the status bit OTPC_STAT_REG[OTPC_STAT_MRDY] gets the value zero. The new mode will be ready for use when the OTPC_STAT_MRDY become again 1. During the mode transition the OTPC_MODE_REG[MODE] become read only. Do not try to use or change any function of the controller until the OTPC_STAT_MRDY bit to become equal to 1.</p> | |

Table 233: OTPC_STAT_REG (0x30070004)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|--|-------|
| 31:3 | - | - | | 0x0 |
| 2 | R | OTPC_STAT_MRDY | <p>Indicates the progress of the transition from a mode of operation to a new mode of operation.</p> <p>0 : There is a transition in progress in a new mode of operation . Wait until the transition to be completed.</p> <p>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.</p> <p>This status bit gets the value zero every time where the OTPC_MODE_REG[MODE] is changing. Do not try to use or change any function of the controller until this status bit to becomes equal to 1.</p> | 0x1 |
| 1 | R | OTPC_STAT_PBUF_EMPTY | <p>Indicates the status of the programming buffer (PBUF).</p> <p>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.</p> <p>1 : The PBUF is empty and a new programming request can be registered in the PBUF by using the OTPC_PADDR_REG and the</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|---|-------|
| | | | OTPC_PWORD_REG registers. This status bit gets the value zero every time where 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 234: OTPC_PADDR_REG (0x30070008)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:10 | - | - | | 0x0 |
| 9:0 | R/W | OTPC_PADDR | The OTPC_PADDR_REG and the OTPC_PWORD_REG consist 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 written 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). | 0x0 |

Table 235: OTPC_PWORD_REG (0x3007000C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:0 | R/W | OTPC_PWORD | The OTPC_PADDR_REG and the OTPC_PWORD_REG consist 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 236: OTPC_TIM1_REG (0x30070010)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|---|-------|
| 31 | - | - | | 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 10us - no more than 100us | 0x9 |
| 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 10us. | 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 10us. | 0x9 |
| 15 | - | - | | 0x0 |
| 14:12 | 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 60ns. This timing parameter refers to the access time of the OTP memory. | 0x1 |
| 11:10 | - | - | | 0x0 |
| 9:8 | 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 | - | - | | 0x0 |
| 6:0 | R/W | OTPC_TIM1_CC_T_1US | The number of hclk_c clock periods (minus one) that give a time interval equal to 1us. 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. | 0x1F |

Table 237: OTPC_TIM2_REG (0x30070014)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|---|-------|
| 31 | R/W | OTPC_TIM2_US_A_DD_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 to become ready to enter in an active mode (reading or programming). It must be at least 2us. | 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 to be disabled the programming mode in the OTP memory. It must be: - at least 5us - no more than 20us | 0x4 |
| 23:21 | R/W | OTPC_TIM2_US_T_ | The number of microseconds (minus one) that are required after the enabling of the power supply of | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|--|-------|
| | | VDS | the OTP memory and before to become ready for the enabling of the internal LDO. It must be at least 1us. | |
| 20:16 | R/W | OTPC_TIM2_US_T_PPS | The number of microseconds (minus one) that are required after the enabling of the programming in the OTP memory and before to be applied the first programming pulse. It must be : - at least 5us - no more than 20us | 0x4 |
| 15 | - | - | | 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 5us - no more than 100us | 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 1us - no more than 5us | 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 10us - no more than 20us | 0x9 |

42.6 LED Controller Registers

Table 238: Register map PWM for LEDs

| Address | Register | Description |
|------------|----------------------------|-----------------------------|
| 0x50030500 | PWMLED_DUTY_CYCLE_LED1_REG | Defines duty cycle for PWM1 |
| 0x50030504 | PWMLED_DUTY_CYCLE_LED2_REG | Defines duty cycle for PWM2 |
| 0x50030508 | PWMLED_FREQUENCY_REG | Defines the PWM frequency |
| 0x5003050C | PWMLED_CTRL_REG | PWM Control register |

Table 239: PWMLED_DUTY_CYCLE_LED1_REG (0x50030500)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|---|-------|
| 15:8 | R/W | LED1_PWM_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 |
| 7:0 | R/W | LED1_PWM_END_CYCLE | Defines the cycle in which the PWM becomes low. If end_cycle is larger than freq and start_cycle is not larger than freq, output is always 1 | 0x0 |

Table 240: PWMLED_DUTY_CYCLE_LED2_REG (0x50030504)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|---|-------|
| 15:8 | R/W | LED2_PWM_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 |
| 7:0 | R/W | LED2_PWM_END_CYCLE | Defines the cycle in which the PWM becomes low. If end_cycle is larger than freq and start_cycle is not larger than freq, output is always 1 | 0x0 |

Table 241: PWMLED_FREQUENCY_REG (0x50030508)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|--|-------|
| 7:0 | R/W | LED_PWM_FREQUENCY | Defines the frequency of PWM 1 2, period = PWM_CLK * (FREQ+1) | 0x0 |

Table 242: PWMLED_CTRL_REG (0x5003050C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---|-------|
| 13:11 | R/W | LED2_LOAD_SEL | Defines LED2 output current: 2.5mA + (LED2_LOAD_SEL*2.5mA). Max = 20mA. | 0x0 |
| 10:8 | R/W | LED1_LOAD_SEL | Defines LED1 output current: 2.5mA + (LED1_LOAD_SEL*2.5mA). Max = 20mA. | 0x0 |
| 7 | R/W | LED2_EN | 0 = LED2 disabled 1 = LED2 enabled | 0x0 |
| 6 | R/W | LED1_EN | 0 = LED1 disabled 1 = LED1 enabled | 0x0 |
| 5:2 | R/W | LED_TRIM | LED current trimming bits | 0x0 |
| 1 | R/W | SW_PAUSE_EN | 0 = PWM are not blocked by SW 1 = PWM 1 and 2 are paused | 0x0 |
| 0 | R/W | PWM_ENABLE | 0 = PWM 1,2 are disabled 1 = PWM 1,2 are enabled | 0x0 |

42.7 QSPI Flash Registers

Table 243: Register map QSPIC

| Address | Register | Description |
|------------|--------------------|--|
| 0x38000000 | QSPIC_CTRLBUS_REG | SPI Bus control register for the Manual mode |
| 0x38000004 | QSPIC_CTRLMODE_REG | Mode Control register |
| 0x38000008 | QSPIC_RECVDATA_REG | Received data for the Manual mode |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|-------------------------|---|
| 0x3800000C | QSPIC_BURSTCMDA_REG | The way of reading in Auto mode (command register A) |
| 0x38000010 | QSPIC_BURSTCMDB_REG | The way of reading in Auto mode (command register B) |
| 0x38000014 | QSPIC_STATUS_REG | The status register of the QSPI controller |
| 0x38000018 | QSPIC_WRITEDATA_REG | Write data to SPI Bus for the Manual mode |
| 0x3800001C | QSPIC_READDATA_R EG | Read data from SPI Bus for the Manual mode |
| 0x38000020 | QSPIC_DUMMYDATA _REG | Send dummy clocks to SPI Bus for the Manual mode |
| 0x38000024 | QSPIC_ERASECTRL_ REG | QSPI Erase control register |
| 0x38000028 | QSPIC_ERASECMDA_ REG | The way of erasing in Auto mode (command register A) |
| 0x3800002C | QSPIC_ERASECMDB_ REG | The way of erasing in Auto mode (command register B) |
| 0x38000030 | QSPIC_BURSTBRK_R EG | Read break sequence in Auto mode |
| 0x38000034 | QSPIC_STATUSCMD_ REG | The way of reading the status of external device in Auto mode |
| 0x38000038 | QSPIC_CHCKERASE_ REG | Check erase progress in Auto mode |
| 0x3800003C | QSPIC_GP_REG | QSPI General Purpose control register |
| 0x38000040 | QSPIC_UCODE_STAR T | QSPIC uCode memory |
| 0x38000080 | QSPIC_CTR_CTRL_R EG | Control register for the decryption engine of the QSPIC |
| 0x38000084 | QSPIC_CTR_SADDR_ REG | Start address of the encrypted content in the QSPI flash |
| 0x38000088 | QSPIC_CTR_EADDR_ REG | End address of the encrypted content in the QSPI flash |
| 0x3800008C | QSPIC_CTR_NONCE_0_3_REG | Nonce bytes 0 to 3 for the AES-CTR algorithm |
| 0x38000090 | QSPIC_CTR_NONCE_4_7_REG | Nonce bytes 4 to 7 for the AES-CTR algorithm |
| 0x38000094 | QSPIC_CTR_KEY_0_3 _REG | Key bytes 0 to 3 for the AES-CTR algorithm |
| 0x38000098 | QSPIC_CTR_KEY_4_7 _REG | Key bytes 4 to 7 for the AES-CTR algorithm |
| 0x3800009C | QSPIC_CTR_KEY_8_1 1_REG | Key bytes 8 to 11 for the AES-CTR algorithm |
| 0x380000A0 | QSPIC_CTR_KEY_12_15_REG | Key bytes 12 to 15 for the AES-CTR algorithm |
| 0x380000A4 | QSPIC_CTR_KEY_16_19_REG | Key bytes 16 to 19 for the AES-CTR algorithm |
| 0x380000A8 | QSPIC_CTR_KEY_20_23_REG | Key bytes 20 to 23 for the AES-CTR algorithm |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|-------------------------|--|
| 0x380000AC | QSPIC_CTR_KEY_24_27_REG | Key bytes 24 to 27 for the AES-CTR algorithm |
| 0x380000B0 | QSPIC_CTR_KEY_28_31_REG | Key bytes 28 to 31 for the AES-CTR algorithm |

Table 244: QSPIC_CTRLBUS_REG (0x38000000)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4 | W | QSPIC_DIS_CS | Write 1 to disable the chip select (active low) when the controller is in Manual mode. | 0x0 |
| 3 | W | QSPIC_EN_CS | Write 1 to enable the chip select (active low) when the controller is in Manual mode. | 0x0 |
| 2 | W | QSPIC_SET_QUAD | Write 1 to set the bus mode in Quad mode when the controller is in Manual mode. | 0x0 |
| 1 | W | QSPIC_SET_DUAL | Write 1 to set the bus mode in Dual mode when the controller is in Manual mode. | 0x0 |
| 0 | W | QSPIC_SET_SINGL_E | Write 1 to set the bus mode in Single SPI mode when the controller is in Manual mode. | 0x0 |

Table 245: QSPIC_CTRLMODE_REG (0x38000004)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 31:14 | - | - | | 0x0 |
| 13 | R/W | QSPIC_USE_32BA | Controls the length of the address that the external memory device uses. 0: The external memory device uses 24 bits address. 1: The external memory device uses 32 bits address. The controller uses this bit in order to decide the number of the address bytes that has to transfer to the external device during Auto mode. | 0x0 |
| 12 | R/W | QSPIC_BUF_LIM_EN | This bit has meaning only for the read in auto mode. Defines the behavior of the controller when the internal buffer is full and there are more data to be retrieved for the current burst. 0: The access in the flash device is not terminated when the internal buffer has no empty space. In this case the QSPI_SCK clock is blocked until to free space in the internal buffer. 1: The access in the flash device is terminated when the internal buffer has no empty space. A new access in the flash device will be initiated when will be requested addresses that are not present in the internal buffer. In both cases the access in the flash device is terminated when there is no any read request. | 0x0 |
| 11:9 | R/W | QSPIC_PCLK_MD | Read pipe clock delay relative to the falling edge of QSPI_SCK. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|--|-------|
| | | | Refer to QSPI Timing for timing parameters and recommended values: 0 to 7 | |
| 8 | R/W | QSPIC_RPIPE_EN | <p>Controls the use of the data read pipe.</p> <p>0: The read pipe is disabled; the sampling clock is defined according to the QSPIC_RXD_NEG setting.</p> <p>1: The read pipe is enabled. The delay of the sampling clock is defined according to the QSPIC_PCLK_MD setting. (Recommended)</p> | 0x0 |
| 7 | R/W | QSPIC_RXD_NEG | <p>Defines the clock edge that is used for the capturing of the received data, when the read pipe is not active (QSPIC_RPIPE_EN = 0).</p> <p>0: Sampling of the received data with the positive edge of the QSPI_SCK</p> <p>1: Sampling of the received data with the negative edge of the QSPI_SCK</p> <p>The internal QSPI_SCK clock that is used by the controller for the capturing of the received data has a skew in respect of the QSPI_SCK that is received by the external memory device. In order to be improved the timing requirements of the read path, the controller supports a read pipe register with programmable clock delay. See also the QSPIC_RPIPE_EN register.</p> | 0x0 |
| 6 | R/W | QSPIC_HRDY_MD | <p>This configuration bit is useful when the frequency of the QSPI clock is much lower than the clock of the AMBA bus, in order to not locks the AMBA bus for a long time.</p> <p>0: Adds wait states via hready signal when an access is performed on the QSPIC_WRITEDATA, QSPIC_READDATA and QSPIC_DUMMYDATA registers. It is not needed to checked the QSPIC_BUSY of the QSPIC_STATUS_REG.</p> <p>1: The controller don't adds wait states via the hready signal, when is performed access on the QSPIC_WRITEDATA, QSPIC_READDATA and QSPIC_DUMMYDATA registers. The QSPIC_BUSY bit of the QSPIC_STATUS_REG must be checked in order to be detected the completion of the requested access.</p> <p>It is applicable only when the controller is in Manual mode. In the case of the Auto mode, the controller always adds wait states via the hready signal.</p> | 0x0 |
| 5 | R/W | QSPIC_IO3_DAT | The value of QSPI_IO3 pad if QSPI_IO3_OEN is 1 | 0x0 |
| 4 | R/W | QSPIC_IO2_DAT | The value of QSPI_IO2 pad if QSPI_IO2_OEN is 1 | 0x0 |
| 3 | R/W | QSPIC_IO3_OEN | <p>QSPI_IO3 output enable. Use this only in SPI or Dual SPI mode to control /HOLD signal. When the Auto Mode is selected (QSPIC_AUTO_MD = 1) and the QUAD SPI is used, set this bit to zero.</p> <p>0: The QSPI_IO3 pad is input.</p> <p>1: The QSPI_IO3 pad is output.</p> | 0x0 |
| 2 | R/W | QSPIC_IO2_OEN | QSPI_IO2 output enable. Use this only in SPI or | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| | | | Dual SPI mode to control /WP signal. When the Auto Mode is selected (QSPIC_AUTO_MD = 1) and the QUAD SPI is used, set this bit to zero. 0: The QSPI_IO2 pad is input. 1: The QSPI_IO2 pad is output. | |
| 1 | R/W | QSPIC_CLK_MD | Mode of the generated QSPI_SCK clock 0: Use Mode 0 for the QSPI_CLK. The QSPI_SCK is low when QSPI_CS is high. 1: Use Mode 3 for the QSPI_CLK. The QSPI_SCK is high when QSPI_CS is high. | 0x0 |
| 0 | R/W | QSPIC_AUTO_MD | Mode of operation 0: The Manual Mode is selected. 1: The Auto Mode is selected. During an erasing the QSPIC_AUTO_MD goes in read only mode (see QSPIC_ERASE_EN) | 0x0 |

Table 246: QSPIC_RECVDATA_REG (0x38000008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:0 | R | QSPIC_RECVDATA | This register contains the received data when the QSPIC_READDATA_REG register is used in Manual mode, in order to be retrieved data from the external memory device and QSPIC_HRDY_MD=1 && QSPIC_BUSY=0. | 0x0 |

Table 247: QSPIC_BURSTCMDA_REG (0x3800000C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 31:30 | R/W | QSPIC_DMY_TX_MD | It describes the mode of the SPI bus during the Dummy bytes phase. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 29:28 | R/W | QSPIC_EXT_TX_MD | It describes the mode of the SPI bus during the Extra Byte phase. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 27:26 | R/W | QSPIC_ADR_TX_MD | It describes the mode of the SPI bus during the address phase. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 25:24 | R/W | QSPIC_INST_TX_MD | It describes the mode of the SPI bus during the instruction phase. 0x0: Single SPI | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|--|-------|
| | | | 0x1: Dual 0x2: Quad 0x3: Reserved | |
| 23:16 | R/W | QSPIC_EXT_BYTE | The value of an extra byte which will be transferred after address (only if QSPIC_EXT_BYTE_EN= 1). Usually this is the Mode Bits in Dual/Quad SPI I/O instructions. | 0x0 |
| 15:8 | R/W | QSPIC_INST_WB | Instruction Value for Wrapping Burst. This value is the selected instruction when QSPIC_WRAP_MD is equal to 1 and the access is a wrapping burst of length and size described by the bit fields QSPIC_WRAP_LEN and QSPIC_WRAP_SIZE respectively. | 0x0 |
| 7:0 | R/W | QSPIC_INST | Instruction Value for Incremental Burst or Single read access. This value is the selected instruction at the cases of incremental burst or single read access. Also this value is used when a wrapping burst is not supported (QSPIC_WRAP_MD) | 0x0 |

Table 248: QSPIC_BURSTCMDB_REG (0x38000010)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|--|-------|
| 31:16 | - | - | | 0x0 |
| 15 | R/W | QSPIC_DMY_FORCE | By setting this bit, the number of dummy bytes is forced to be equal to 3. In this case the QSPIC_DMY_NUM field is overruled and has no function. 0: The number of dummy bytes is controlled by the QSPIC_DMY_NUM field 1: Three dummy bytes are used. The QSPIC_DMY_NUM is overruled. | 0x0 |
| 14:12 | R/W | QSPIC_CS_HIGH_MIN | Between the transmissions of two different instructions to the flash memory, the SPI bus stays in idle state (QSPI_CS high) for at least this number of QSPI_SCK clock cycles. See the QSPIC_ERS_CS_HI register for some exceptions. | 0x0 |
| 11:10 | R/W | QSPIC_WRAP_SIZE | It describes the selected data size of a wrapping burst (QSPIC_WRAP_MD). 0x0: byte access (8-bits) 0x1: half word access (16 bits) 0x2: word access (32-bits) 0x3: Reserved | 0x0 |
| 9:8 | R/W | QSPIC_WRAP_LEN | It describes the selected length of a wrapping burst (QSPIC_WRAP_MD). 0x0: 4 beat wrapping burst 0x1: 8 beat wrapping burst 0x2: 16 beat wrapping burst 0x3: Reserved | 0x0 |
| 7 | R/W | QSPIC_WRAP_MD | Wrap mode 0: The QSPIC_INST is the selected instruction at any access. 1: The QSPIC_INST_WB is the selected instruction | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|--|-------|
| | | | at any wrapping burst access of length and size described by the registers QSPIC_WRAP_LEN and QSPIC_WRAP_SIZE respectively. In all other cases the QSPIC_INST is the selected instruction. Use this feature only when the serial FLASH memory supports a special instruction for wrapping burst access. | |
| 6 | R/W | QSPIC_INST_MD | Instruction mode 0: Transmit instruction at any burst access. 1: Transmit instruction only in the first access after the selection of Auto Mode. | 0x0 |
| 5:4 | R/W | QSPIC_DMY_NUM | Number of Dummy Bytes 0x0: Zero Dummy Bytes (Don't Send Dummy Bytes) 0x1: Send 1 Dummy Byte 0x2: Send 2 Dummy Bytes 0x3: Send 4 Dummy Bytes When QSPIC_DMY_FORCE is enabled, the QSPIC_DMY_NUM is overruled. In this case the number of dummy bytes is defined by the QSPIC_DMY_FORCE and is equal to 3, independent of the value of the QSPIC_DMY_NUM. | 0x0 |
| 3 | R/W | QSPIC_EXT_HF_DS | Extra Half Disable Output 0: if QSPIC_EXT_BYTE_EN=1, is transmitted the complete QSPIC_EXT_BYTE 1: if QSPIC_EXT_BYTE_EN=1, the output is disabled (hi-z) during the transmission of bits [3:0] of QSPIC_EXT_BYTE | 0x0 |
| 2 | R/W | QSPIC_EXT_BYTE_EN | Extra Byte Enable 0: Don't Send QSPIC_EXT_BYTE 1: Send QSPIC_EXT_BYTE | 0x0 |
| 1:0 | R/W | QSPIC_DAT_RX_M_D | It describes the mode of the SPI bus during the data phase. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |

Table 249: QSPIC_STATUS_REG (0x38000014)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | QSPIC_BUSY | The status of the SPI Bus. 0: The SPI Bus is idle 1: The SPI Bus is active. Read data, write data or dummy data activity is in progress. Has meaning only in Manual mode and only when | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--------------------|-------|
| | | | QSPIC_HRDY_MD = 1. | |

Table 250: QSPIC_WRITEDATA_REG (0x38000018)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|--|-------|
| 31:0 | W | QSPIC_WRITEDATA | <p>Writing to this register is generating a data transfer from the controller to the external memory device. The data written in this register, is then transferred to the memory using the selected mode of the SPI bus (SPI, Dual SPI, Quad SPI). The data size of the access to this register can be 32-bits / 16-bits/ 8-bits and is equal to the number of the transferred bits.</p> <p>This register has meaning only when the controller is in Manual mode.</p> | 0x0 |

Table 251: QSPIC_READDATA_REG (0x3800001C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:0 | R | QSPIC_READDATA | <p>A read access at this register generates a data transfer from the external memory device to the QSPIC controller. The data is transferred using the selected mode of the SPI bus (SPI, Dual SPI, Quad SPI). The data size of the access to this register can be 32-bits / 16-bits / 8-bits and is equal to the number of the transferred bits.</p> <p>This register has meaning only when the controller is in Manual mode.</p> | 0x0 |

Table 252: QSPIC_DUMMYDATA_REG (0x38000020)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|---|-------|
| 31:0 | W | QSPIC_DUMMYDATA | <p>Writing to this register generates a number of clock pulses to the SPI bus. During the last clock of this activity in the SPI bus, the QSPI_IOx data pads are in hi-z state. The data size of the access to this register can be 32-bits / 16-bits/ 8-bits. The number of generated pulses is equal to: (size of AHB bus access) / (size of SPI bus). The size of SPI bus is equal to 1, 2 or 4 for Single, Dual or Quad SPI mode respectively.</p> <p>This register has meaning only when the controller is in Manual mode.</p> | 0x0 |

Table 253: QSPIC_ERASECTRL_REG (0x38000024)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|--|-------|
| 31:28 | - | - | | 0x0 |
| 27:25 | R | QSPIC_ERS_STAT | It shows the progress of sector/block erasing (read only). | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| | | | 0x0: No Erase. 0x1: Pending erase request 0x2: Erase procedure is running 0x3: Suspended Erase procedure 0x4: Finishing the Erase procedure 0x5..0x7: Reserved | |
| 24 | R/W | QSPIC_ERASE_EN | During Manual mode (QSPIC_AUTO_MD = 0). This bit is in read only mode. During Auto mode (QSPIC_AUTO_MD = 1). To request the erasing of the block/sector (QSPIC_ERS_ADDR, 12'b0) write 1 to this bit. This bit is cleared automatically with the end of the erasing. Until the end of erasing the QSPIC_ERASE_EN remains in read only mode. During the same period of time the controller remains in Auto Mode (QSPIC_AUTO_MD goes in read only mode). | 0x0 |
| 23:4 | R/W | QSPIC_ERS_ADDR | Defines the address of the block/sector that is requested to be erased. If QSPIC_USE_32BA = 0 (24 bits addressing), bits QSPIC_ERASECTRL_REG[23-12] determine the block/ sector address bits [23-12]. QSPIC_ERASECTRL_REG[11-4] are ignored by the controller. If QSPIC_USE_32BA = 1 (32 bits addressing) bits QSPIC_ERASECTRL_REG[23-4] determine the block / sectors address bits [31:12] | 0x0 |
| 3:0 | - | - | | 0x0 |

Table 254: QSPIC_ERASECMDA_REG (0x38000028)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|--|-------|
| 31:24 | R/W | QSPIC_RES_INST | The code value of the erase resume instruction | 0x0 |
| 23:16 | R/W | QSPIC_SUS_INST | The code value of the erase suspend instruction. | 0x0 |
| 15:8 | R/W | QSPIC_WEN_INST | The code value of the write enable instruction. | 0x0 |
| 7:0 | R/W | QSPIC_ERS_INST | The code value of the erase instruction. | 0x0 |

Table 255: QSPIC_ERASECMDB_REG (0x3800002C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 31:30 | - | - | | 0x0 |
| 29:24 | R/W | QSPIC_RESSUS_DLY | Defines a timer that counts the minimum allowed delay between an erase suspend command and the previous erase resume command (or the initial erase command). 0: Dont wait. The controller starts immediately to suspend the erase procedure. 1..63: The controller waits for at least this number of 222kHz clock cycles before the suspension of erasing. Time starts counting after the end of the | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|--|-------|
| | | | previous erase resume command (or the initial erase command) | |
| 23:20 | - | - | | 0x0 |
| 19:16 | R/W | QSPIC_ERSRES_HLD | The controller must stay without flash memory reading requests for this number of AMBA hclk clock cycles, before to perform the command of erase or erase resume 15 - 0 | 0x0 |
| 15 | - | - | | 0x0 |
| 14:10 | R/W | QSPIC_ERS_CS_HI | After the execution of instructions: write enable, erase, erase suspend and erase resume, the QSPI_CS remains high for at least this number of qspi bus clock cycles. | 0x0 |
| 9:8 | R/W | QSPIC_EAD_TX_MODE | The mode of the QSPI Bus during the address phase of the erase instruction 0x0: Single 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 7:6 | R/W | QSPIC_RES_TX_MODE | The mode of the QSPI Bus during the transmission of the resume instruction 0x0: Single 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 5:4 | R/W | QSPIC_SUS_TX_MODE | The mode of the QSPI Bus during the transmission of the suspend instruction. 0x0: Single 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 3:2 | R/W | QSPIC_WEN_TX_MODE | The mode of the QSPI Bus during the transmission of the write enable instruction. 0x0: Single 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 1:0 | R/W | QSPIC_ERS_TX_MODE | The mode of the QSPI Bus during the instruction phase of the erase instruction 0x0: Single 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |

Table 256: QSPIC_BURSTBRK_REG (0x38000030)

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

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 20 | R/W | QSPIC_SEC_HF_DS | <p>Disable output during the transmission of the second half (QSPIC_BRK_WRD[3:0]). Setting this bit is only useful if QSPIC_BRK_EN =1 and QSPIC_BRK_SZ= 1.</p> <p>0: The controller drives the QSPI bus during the transmission of the QSPIC_BRK_WRD[3:0].</p> <p>1: The controller leaves the QSPI bus in Hi-Z during the transmission of the QSPIC_BRK_WORD[3:0].</p> | 0x0 |
| 19:18 | R/W | QSPIC_BRK_TX_MD | <p>The mode of the QSPI Bus during the transmission of the burst break sequence.</p> <p>0x0: Single 0x1: Dual 0x2: Quad 0x3: Reserved</p> | 0x0 |
| 17 | R/W | QSPIC_BRK_SZ | <p>The size of Burst Break Sequence</p> <p>0: One byte (Send QSPIC_BRK_WRD[15:8]) 1: Two bytes (Send QSPIC_BRK_WRD[15:0])</p> | 0x0 |
| 16 | R/W | QSPIC_BRK_EN | <p>Controls the application of a special command (read burst break sequence) that is used in order to force the device to abandon the continuous read mode.</p> <p>0: The special command is not applied 1: The special command is applied</p> <p>This special command is applied by the controller to the external device under the following conditions:</p> <ul style="list-style-type: none"> - the controller is in Auto mode - the QSPIC_INST_MD = 1 - the previous command that has been applied in the external device was read - the controller want to apply to the external device a command different than the read. | 0x0 |
| 15:0 | R/W | QSPIC_BRK_WRD | <p>This is the value of a special command (read burst break sequence) that is applied by the controller to the external memory device, in order to force the memory device to abandon the continuous read mode.</p> | 0x0 |

Table 257: QSPIC_STATUSCMD_REG (0x38000034)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 31:23 | - | - | | 0x0 |
| 22 | R/W | QSPIC_STSDLY_SEL | <p>Defines the timer which is used to count the delay that it has to wait before to read the FLASH Status Register, after an erase or an erase resume command.</p> <p>0: The delay is controlled by the QSPIC_RESSTS_DLY which counts on the qspi clock. 1: The delay is controlled by the QSPIC_RESSUS_DLY which counts on the 222 kHz clock.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|--|-------|
| 21:16 | R/W | QSPIC_RESSTS_DL | <p>Defines a timer that counts the minimum required delay between the reading of the status register and of the previous erase or erase resume instruction.</p> <p>0: Dont wait. The controller starts to reading the Flash memory status register immediately.</p> <p>1..63: The controller waits for at least this number of QSPI_CLK cycles and afterwards it starts to reading the Flash memory status register. The timer starts to count after the end of the previous erase or erase resume command.</p> <p>The actual timer that will be used by the controller before the reading of the Flash memory status register is defined by the QSPIC_STSDLY_SEL.</p> | 0x0 |
| 15 | R/W | QSPIC_BUSY_VAL | <p>Defines the value of the Busy bit which means that the flash is busy.</p> <p>0: The flash is busy when the Busy bit is equal to 0.</p> <p>1: The flash is busy when the Busy bit is equal to 1.</p> | 0x0 |
| 14:12 | R/W | QSPIC_BUSY_POS | It describes who from the bits of status represents the Busy bit (7 - 0). | 0x0 |
| 11:10 | R/W | QSPIC_RSTAT_RX_M | <p>The mode of the QSPI Bus during the receive status phase of the read status instruction</p> <p>0x0: Single</p> <p>0x1: Dual</p> <p>0x2: Quad</p> <p>0x3: Reserved</p> | 0x0 |
| 9:8 | R/W | QSPIC_RSTAT_TX_M | <p>The mode of the QSPI Bus during the instruction phase of the read status instruction.</p> <p>0x0: Single</p> <p>0x1: Dual</p> <p>0x2: Quad</p> <p>0x3: Reserved</p> | 0x0 |
| 7:0 | R/W | QSPIC_RSTAT_INS_T | <p>The code value of the read status instruction.</p> <p>It is transmitted during the instruction phase of the read status instruction.</p> | 0x0 |

Table 258: QSPIC_CHCKERASE_REG (0x38000038)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 31:0 | W | QSPIC_CHCKERAS_E | Writing any value to this register during erasing, forces the controller to read the flash memory status register. Depending on the value of the Busy bit, it updates the QSPIC_ERASE_EN. | 0x0 |

Table 259: QSPIC_GP_REG (0x3800003C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|--|-------|
| 4:3 | R/W | QSPIC_PADS_SLE_W | QSPI pads slew rate control. Indicative values under certain conditions: 0: Rise=1.7 V/ns, Fall=1.9 V/ns (weak) | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|---|-------|
| | | | 1: Rise=2.0 V/ns, Fall=2.3 V/ns 2: Rise=2.3 V/ns, Fall=2.6 V/ns 3: Rise=2.4 V/ns, Fall=2.7 V/ns (strong) Conditions: FLASH pin capacitance 6 pF, Vcc=1.8V, T=25C and Idrive=16mA. | |
| 2:1 | R/W | QSPIC_PADS_DRV | QSPIC pads drive current 0: 4 mA 1: 8 mA 2: 12 mA 3: 16 mA | 0x0 |
| 0 | - | - | | 0x0 |

Table 260: QSPIC_UCODE_START (0x38000040)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 31:0 | R/W | QSPIC_UCODE_X | The controller has a dedicated memory cell of 16 words x 32 bits that is used for the storing of the microcode that describes the initialization process of the external flash device. The first word (word 0) of this memory can be accessed by accessing the QSPIC_UCODE_START register. The next words can be accessed by accessing the QSPIC_UCODE_START + 4*X (X=1 .. 15). | 0x0 |

Table 261: QSPIC_CTR_CTRL_REG (0x38000080)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| 0 | R/W | QSPIC_CTR_EN | Controls the AES-CTR decryption feature of the QSPIC, which enables the decryption (on-the-fly) of the data that are retrieved from the flash memory device. 0: The AES-CTR decryption is disabled. 1: The controller will decrypt the content of the flash memory device that is placed in the address space that is defined by the QSPIC_CTR_SADDR_REG and QSPIC_CTR_EADDR_REG registers. The data that are placed outside the previous space are not decrypted by the QSPIC. The decryption is performed by using the AES-CTR algorithm. The AES key is defined by the QSPIC_CTR_KEY_x_y_REG registers and the nonce value by the QSPIC_CTR_NONCE_x_y_REG registers. This configuration bit has meaning only while the controller is in Auto mode. The on-the-fly decryption is not provided in Manual mode. | 0x0 |

Table 262: QSPIC_CTR_SADDR_REG (0x38000084)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|--|-------|
| 31:10 | R/W | QSPIC_CTR_SADD | Defines the bits [31:10] of the start address in the | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | R | flash memory, where an encrypted image is placed. The bits [9:0] are considered always as zero. This has meaning only when the decryption is active. See also the register QSPIC_CTR_CTRL_REG[QSPIC_CTR_EN]. | |
| 9:0 | - | - | | 0x0 |

Table 263: QSPIC_CTR_EADDR_REG (0x38000088)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 31:10 | R/W | QSPIC_CTR_EADD R | Defines the bits [31:10] of the end address in the flash memory, where an encrypted image is placed. The bits [9:0] are considered always as 0x3ff. This has meaning only when the decryption is active. See also the register QSPIC_CTR_CTRL_REG[QSPIC_CTR_EN]. | 0x0 |
| 9:0 | - | - | | 0x3FF |

Table 264: QSPIC_CTR_NONCE_0_3_REG (0x3800008C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|--|-------|
| 31:0 | R/W | QSPIC_CTR_NONCE_0_3 | <p>Defines the 8 bytes of the nonce value (N0 - N7) that is used by the AES-CTR algorithm in order to be constructed the counter block (CTRB). The total size of the counter block is 128 bits or 16 bytes :</p> <p>CTRB0 CTRB1 CTRB2 CTRB3...CTRB14 CTRB15.</p> <p>The first 8 bytes (CTRB0 - CTRB7) of the counter block consisted by the nonce value.</p> <p>The next 8 bytes of the counter block (CTRB8- CTRB15), are produced automatically by the hardware based on the address offset inside the encrypted image, from where are retrieved the requested data.</p> <p>The mapping of the nonce bytes to the corresponding QSPIC_NONCE_X_Y_REG registers is the following :</p> <p>{CTRB0, CTRB1, CTRB2, CTRB3} = {N0, N1, N2, N3} = QSPIC_NONCE_0_3_REG[31:0]</p> <p>{CTRB4, CTRB5, CTRB6, CTRB7} = {N4, N5, N6, N7} = QSPIC_NONCE_4_7_REG[31:0]</p> <p>All these registers make sense only when QSPIC_CTR_CTRL_REG[QSPIC_CTR_EN] = 1. Do not perform access to an encrypted address range while the updating process of the nonce value is in progress.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 265: QSPIC_CTR_NONCE_4_7_REG (0x38000090)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:0 | R/W | QSPIC_CTR_NONCE_4_7 | See the description in the QSPIC_NONCE_0_3. | 0x0 |

Table 266: QSPIC_CTR_KEY_0_3_REG (0x38000094)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|--|-------|
| 31:0 | W | QSPIC_CTR_KEY_0_3 | <p>Defines the key that is used by the AES-CTR algorithm, when the on-the-fly decryption is enabled (QSPIC_CTR_CTRL_REG[QSPIC_CTR_EN] = 1). The size of the decryption key is 256bits or 32 bytes :</p> <p>K0 K1 K2 K3...K30 K31.</p> <p>The mapping of the bytes to the corresponding QSPIC_CTR_KEY_X_Y_REG registers is the following :</p> <pre> {K0, K1, K2, K3} = QSPIC_CTR_KEY_0_3_REG[31:0] {K4, K5, K6, K7} = QSPIC_CTR_KEY_4_7_REG[31:0] {K8, K9, K10, K11} = QSPIC_CTR_KEY_8_11_REG[31:0] {K12, K13, K14, K15} = QSPIC_CTR_KEY_12_15_REG[31:0] {K16, K17, K18, K19} = QSPIC_CTR_KEY_16_19_REG[31:0] {K20, K21, K22, K23} = QSPIC_CTR_KEY_20_23_REG[31:0] {K24, K25, K26, K27} = QSPIC_CTR_KEY_24_27_REG[31:0] {K28, K29, K30, K31} = QSPIC_CTR_KEY_28_31_REG[31:0] </pre> <p>All these registers make sense only when QSPIC_CTR_CTRL_REG[QSPIC_CTR_EN] = 1. Do not perform access to an encrypted address range while the updating process of the decryption key is in progress.</p> | 0x0 |

Table 267: QSPIC_CTR_KEY_4_7_REG (0x38000098)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|---|-------|
| 31:0 | W | QSPIC_CTR_KEY_4_7 | See the description in the QSPIC_CTR_KEY_0_3. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 268: QSPIC_CTR_KEY_8_11_REG (0x3800009C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|---|-------|
| 31:0 | W | QSPIC_CTR_KEY_8_11 | See the description in the QSPIC_CTR_KEY_0_3. | 0x0 |

Table 269: QSPIC_CTR_KEY_12_15_REG (0x380000A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:0 | W | QSPIC_CTR_KEY_12_15 | See the description in the QSPIC_CTR_KEY_0_3. | 0x0 |

Table 270: QSPIC_CTR_KEY_16_19_REG (0x380000A4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:0 | W | QSPIC_CTR_KEY_16_19 | See the description in the QSPIC_CTR_KEY_0_3. | 0x0 |

Table 271: QSPIC_CTR_KEY_20_23_REG (0x380000A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:0 | W | QSPIC_CTR_KEY_20_23 | See the description in the QSPIC_CTR_KEY_0_3. | 0x0 |

Table 272: QSPIC_CTR_KEY_24_27_REG (0x380000AC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:0 | W | QSPIC_CTR_KEY_24_27 | See the description in the QSPIC_CTR_KEY_0_3. | 0x0 |

Table 273: QSPIC_CTR_KEY_28_31_REG (0x380000B0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:0 | W | QSPIC_CTR_KEY_28_31 | See the description in the QSPIC_CTR_KEY_0_3. | 0x0 |

42.8 QSPI Ram Registers

Table 274: Register map QSPIC2

| Address | Register | Description |
|------------|--------------------|--|
| 0x34000000 | QSPIC2_CTRLBUS_REG | SPI Bus control register for the Manual mode |
| 0x34000004 | QSPIC2_MODE_REG | Mode control register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|----------------------|---|
| 0x34000008 | QSPIC2_RECVDATA_REG | Received data for the Manual mode |
| 0x3400000C | QSPIC2_BURSTCMDA_REG | The way of reading in Auto mode (command register A) |
| 0x34000010 | QSPIC2_BURSTCMDB_REG | The way of reading in Auto mode (command register B) |
| 0x34000014 | QSPIC2_STATUS_REG | The status register of the QSPI controller |
| 0x34000018 | QSPIC2_WRITEDATA_REG | Write data to SPI Bus for the Manual mode |
| 0x3400001C | QSPIC2_READDATA_REG | Read data from SPI Bus for the Manual mode |
| 0x34000020 | QSPIC2_DUMMYDATA_REG | Send dummy clocks to SPI Bus for the Manual mode |
| 0x34000024 | QSPIC2_ERASECTRL_REG | Erase control register |
| 0x34000028 | QSPIC2_ERASECMDA_REG | The way of erasing in Auto mode (command register A) |
| 0x3400002C | QSPIC2_ERASECMDB_REG | The way of erasing in Auto mode (command register B) |
| 0x34000030 | QSPIC2_BURSTBRK_REG | Read break sequence in Auto mode |
| 0x34000034 | QSPIC2_STATUSCMD_REG | The way of reading the status of external device in Auto mode |
| 0x34000038 | QSPIC2_CHCKERASE_REG | Check erase progress in Auto mode |
| 0x3400003C | QSPIC2_GP_REG | General purpose QSPIC2 register |
| 0x34000040 | QSPIC2_AWRITECMD_REG | The way of writing in Auto mode when the external device is a serial SRAM |
| 0x34000044 | QSPIC2_MEMBLEN_REG | External memory burst length configuration |

Table 275: QSPIC2_CTRLBUS_REG (0x34000000)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4 | W | QSPIC_DIS_CS | Write 1 to disable the chip select (active low) when the controller is in Manual mode. | 0x0 |
| 3 | W | QSPIC_EN_CS | Write 1 to enable the chip select (active low) when the controller is in Manual mode. | 0x0 |
| 2 | W | QSPIC_SET_QUAD | Write 1 to set the bus mode in Quad mode when the controller is in Manual mode. | 0x0 |
| 1 | W | QSPIC_SET_DUAL | Write 1 to set the bus mode in Dual mode when the controller is in Manual mode. | 0x0 |
| 0 | W | QSPIC_SET_SINGL_E | Write 1 to set the bus mode in Single SPI mode when the controller is in Manual mode. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 276: QSPIC2_CTRLMODE_REG (0x34000004)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------|---|-------|
| 31:17 | - | - | | 0x0 |
| 16 | R/W | QSPIC_CLK_FREE_EN | <p>Controls the behavior of the QSPI_SCK when the QSPI_CS is high and the QSPIC_CS_MD=1.</p> <p>0: Is produced one QSPI_SCK clock pulse after each 0 to 1 transition in the QSPI_CS.</p> <p>1: The QSPI_SCK clock remains always active, while the QSPI_CS is inactive.</p> <p>This setting has meaning only when the QSPIC_CS_MD=1.</p> | 0x0 |
| 15 | R/W | QSPIC_CS_MD | <p>Controls the clock edge with which is produced the QSPI_CS signal.</p> <p>0: The QSPI_CS is produced with the rising edge of the QSPI_SCK. The QSPI_SCK is always inactive while the QSPI_CS is high.</p> <p>1: The QSPI_CS is produced with the falling edge of the QSPI_SCK. The behavior of the QSPI_SCK while the QSPI_CS is high, is controlled by the QSPIC_CLK_FREE_EN.</p> | 0x0 |
| 14 | R/W | QSPIC_SRAM_EN | <p>Defines the type of the external device that is connected on the QSPIC controller</p> <p>0: The external memory device is a serial Flash</p> <p>1: The external memory device is a serial SRAM</p> <p>When the external device is a serial SRAM, the erase suspend/ resume functionality of the controller is disabled. In this case the writing of the QSPIC_ERASECTRL_REG[QSPIC_ERASE_EN] bit has no effect. Also, the memory space where the external device is mapped, is considered as writable.</p> | 0x0 |
| 13 | R/W | QSPIC_USE_32BA | <p>Controls the length of the address that the external memory device uses.</p> <p>0: The external memory device uses 24 bits address.</p> <p>1: The external memory device uses 32 bits address.</p> <p>The controller uses this bit in order to decide the number of the address bytes that has to transfer to the external device during Auto mode.</p> | 0x0 |
| 12 | R/W | QSPIC_FORCE_NSE_Q_EN | <p>Controls the way with which is addressed by the QSPI controller a burst request from the AMBA bus.</p> <p>0: The controller translates a burst access on the AMBA bus as a burst access on the QSPI bus. That results to the minimum number of command/address phases.</p> <p>1: The controller will split a burst access on the AMBA bus into a number of single accesses on the QSPI bus. That results to a separate command for each beat of the burst. E.g a 4-beat word incremental AMBA read access will be split into 4 different sequences on the QSPI bus: command/address/extra clock/read data. The</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| | | | <p>QSPI_CS will be low only for the time that is needed for each of these single access.</p> <p>This configuration bit is useful when the clock frequency of the QSPI bus is much higher than the clock of the AMBA bus. In this case the interval for which the CS remains low is minimized, achieving lower power dissipation with respect of the case where the QSPIC_FORCENSEQ_EN=0, at cost of performance.</p> | |
| 11:9 | R/W | QSPIC_PCLK_MD | Controls the read pipe clock delay relative to the falling edge of QSPI_SCK. Refer to QSPI Timing for timing parameters | 0x0 |
| 8 | R/W | QSPIC_RPIPE_EN | <p>Controls the use of the data read pipe.</p> <p>0: The read pipe is disabled, the sampling clock is defined according to the QSPIC_RXD_NEG setting.</p> <p>1: The read pipe is enabled. The delay of the sampling clock is defined according to the QSPIC_PCLK_MD setting. (Recommended)</p> | 0x0 |
| 7 | R/W | QSPIC_RXD_NEG | <p>Defines the clock edge that is used for the capturing of the received data, when the read pipe is not active (QSPIC_RPIPE_EN = 0).</p> <p>0: Sampling of the received data with the positive edge of the QSPI_SCK</p> <p>1: Sampling of the received data with the negative edge of the QSPI_SCK</p> <p>The internal QSPI_SCK clock that is used by the controller for the capturing of the received data has a skew in respect of the QSPI_SCK that is received by the external memory device. In order to be improved the timing requirements of the read path, the controller supports a read pipe register with programmable clock delay. See also the QSPIC_RPIPE_EN register.</p> | 0x0 |
| 6 | R/W | QSPIC_HRDY_MD | <p>This configuration bit is useful when the frequency of the QSPI clock is much lower than the clock of the AMBA bus, in order to not lock the AMBA bus for a long time.</p> <p>0: Adds wait states via hready signal when an access is performed on the QSPIC_WRITEDATA, QSPIC_READDATA and QSPIC_DUMMYDATA registers. It is not needed to check the QSPIC_BUSY of the QSPIC_STATUS_REG.</p> <p>1: The controller don't adds wait states via the hready signal, when is performed access on the QSPIC_WRITEDATA, QSPIC_READDATA and QSPIC_DUMMYDATA registers. The QSPIC_BUSY bit of the QSPIC_STATUS_REG must be checked in order to be detected the completion of the requested access.</p> <p>It is applicable only when the controller is in Manual mode. In the case of the Auto mode, the controller always adds wait states via the hready signal.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| 5 | R/W | QSPIC_IO3_DAT | The value of QSPI_IO3 pad if QSPI_IO3_OEN is 1 | 0x0 |
| 4 | R/W | QSPIC_IO2_DAT | The value of QSPI_IO2 pad if QSPI_IO2_OEN is 1 | 0x0 |
| 3 | R/W | QSPIC_IO3_OEN | QSPI_IO3 output enable. Use this only in SPI or Dual SPI mode to control /HOLD signal. When the Auto Mode is selected (QSPIC_AUTO_MD = 1) and the QUAD SPI is used, set this bit to zero. 0: The QSPI_IO3 pad is input. 1: The QSPI_IO3 pad is output. | 0x0 |
| 2 | R/W | QSPIC_IO2_OEN | QSPI_IO2 output enable. Use this only in SPI or Dual SPI mode to control /WP signal. When the Auto Mode is selected (QSPIC_AUTO_MD = 1) and the QUAD SPI is used, set this bit to zero. 0: The QSPI_IO2 pad is input. 1: The QSPI_IO2 pad is output. | 0x0 |
| 1 | R/W | QSPIC_CLK_MD | Mode of the generated QSPI_SCK clock 0: Use Mode 0 for the QSPI_CLK. The QSPI_SCK is low when QSPI_CS is high. 1: Use Mode 3 for the QSPI_CLK. The QSPI_SCK is high when QSPI_CS is high. See also the register QSPIC_CS_MD and the QSPIC_CLK_FREE_EN | 0x0 |
| 0 | R/W | QSPIC_AUTO_MD | Mode of operation 0: The Manual Mode is selected. 1: The Auto Mode is selected. During an erasing the QSPIC_AUTO_MD goes in read only mode (see QSPIC_ERASE_EN) | 0x0 |

Table 277: QSPIC2_RECVDATA_REG (0x34000008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:0 | R | QSPIC_RECVDATA | This register contains the received data when the QSPIC_READDATA_REG register is used in Manual mode, in order to be retrieved data from the external memory device and QSPIC_HRDY_MD=1 && QSPIC_BUSY=0. | 0x0 |

Table 278: QSPIC2_BURSTCMDA_REG (0x3400000C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|--|-------|
| 31:30 | R/W | QSPIC_DMY_TX_M D | It describes the mode of the SPI bus during the Dummy bytes phase. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 29:28 | R/W | QSPIC_EXT_TX_M D | It describes the mode of the SPI bus during the Extra Byte phase. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| | | | 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | |
| 27:26 | R/W | QSPIC_ADR_TX_MD | It describes the mode of the SPI bus during the address phase. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 25:24 | R/W | QSPIC_INST_TX_MD | It describes the mode of the SPI bus during the instruction phase. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 23:16 | R/W | QSPIC_EXT_BYTE | The value of an extra byte which will be transferred after address (only if QSPIC_EXT_BYTE_EN= 1). Usually this is the Mode Bits in Dual/Quad SPI I/O instructions. | 0x0 |
| 15:8 | R/W | QSPIC_INST_WB | Instruction Value for Wrapping Burst. This value is the selected instruction when QSPIC_WRAP_MD is equal to 1 and the access is a wrapping burst of length and size described by the bit fields QSPIC_WRAP_LEN and QSPIC_WRAP_SIZE respectively. | 0x0 |
| 7:0 | R/W | QSPIC_INST | Instruction Value for Incremental Burst or Single read access. This value is the selected instruction at the cases of incremental burst or single read access. Also this value is used when a wrapping burst is not supported (QSPIC_WRAP_MD) | 0x0 |

Table 279: QSPIC2_BURSTCMDB_REG (0x340000010)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|--|-------|
| 31:16 | - | - | | 0x0 |
| 15 | R/W | QSPIC_DMY_FORCE | By setting this bit, the number of dummy bytes is forced to be equal to 3. In this case the QSPIC_DMY_NUM field is overruled and has no function. 0: The number of dummy bytes is controlled by the QSPIC_DMY_NUM field 1: Three dummy bytes are used. The QSPIC_DMY_NUM is overruled. | 0x0 |
| 14:12 | R/W | QSPIC_CS_HIGH_MIN | Between the transmission of two different instructions to the flash memory, the qspi bus stays in idle state (QSPI_CS high) for at least this number of QSPI_SCK clock cycles. See the QSPIC_ERS_CS_HI and the QSPIC_WR_CS_HIGH_MIN registers for some exceptions. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|--|-------|
| 11:10 | R/W | QSPIC_WRAP_SIZE | <p>It describes the selected data size of a wrapping burst (QSPIC_WRAP_MD).</p> <p>0x0: Byte access (8-bits) 0x1: Half word access (16 bits) 0x2: Word access (32-bits) 0x3: Reserved</p> | 0x0 |
| 9:8 | R/W | QSPIC_WRAP_LEN | <p>It describes the selected length of a wrapping burst (QSPIC_WRAP_MD).</p> <p>0x0: 4 beat wrapping burst 0x1: 8 beat wrapping burst 0x2: 16 beat wrapping burst 0x3: Reserved</p> | 0x0 |
| 7 | R/W | QSPIC_WRAP_MD | <p>Wrap mode</p> <p>0: The QSPIC_INST is the selected instruction at any access.</p> <p>1: The QSPIC_INST_WB is the selected instruction at any wrapping burst access of length and size described by the registers QSPIC_WRAP_LEN and QSPIC_WRAP_SIZE respectively. In all other cases the QSPIC_INST is the selected instruction. Use this feature only when the serial FLASH memory supports a special instruction for wrapping burst access.</p> | 0x0 |
| 6 | R/W | QSPIC_INST_MD | <p>Instruction mode</p> <p>0: Transmit instruction at any burst access. 1: Transmit instruction only in the first access after the selection of Auto Mode.</p> | 0x0 |
| 5:4 | R/W | QSPIC_DMY_NUM | <p>Number of Dummy Bytes</p> <p>0x0: Zero Dummy Bytes (Don't Send Dummy Bytes) 0x1: Send 1 Dummy Byte 0x2: Send 2 Dummy Bytes 0x3: Send 4 Dummy Bytes</p> <p>When QSPIC_DMY_FORCE is enabled, the QSPIC_DMY_NUM is overruled. In this case the number of dummy bytes is defined by the QSPIC_DMY_FORCE and is equal to 3, independent of the value of the QSPIC_DMY_NUM.</p> | 0x0 |
| 3 | R/W | QSPIC_EXT_HF_DS | <p>Extra Half Disable Output</p> <p>0: if QSPIC_EXT_BYTE_EN=1 then transmit the complete QSPIC_EXT_BYTE 1: if QSPIC_EXT_BYTE_EN=1 then disable (hi-z) output during the transmission of bits [3:0] of QSPIC_EXT_BYTE</p> | 0x0 |
| 2 | R/W | QSPIC_EXT_BYTE_EN | <p>Extra Byte Enable</p> <p>0: Don't Send QSPIC_EXT_BYTE 1: Send QSPIC_EXT_BYTE</p> | 0x0 |
| 1:0 | R/W | QSPIC_DAT_RX_M_D | It describes the mode of the SPI bus during the data phase. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | |

Table 280: QSPIC2_STATUS_REG (0x34000014)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | QSPIC_BUSY | The status of the SPI Bus. 0: The SPI Bus is idle 1: The SPI Bus is active. Read data, write data or dummy data activity is in progress. Has meaning only in Manual mode and only when QSPIC_HRDY_MD = 1. | 0x0 |

Table 281: QSPIC2_WRITEDATA_REG (0x34000018)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|---|-------|
| 31:0 | W | QSPIC_WRITEDATA | Writing to this register is generating a data transfer from the controller to the external memory device. The data written in this register, is then transferred to the memory using the selected mode of the SPI bus (SPI, Dual SPI, Quad SPI). The data size of the access to this register can be 32-bits / 16-bits/ 8-bits and is equal to the number of the transferred bits. This register has meaning only when the controller is in Manual mode. | 0x0 |

Table 282: QSPIC2_READDATA_REG (0x3400001C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 31:0 | R | QSPIC_READDATA | A read access at this register generates a data transfer from the external memory device to the QSPIC controller. The data is transferred using the selected mode of the SPI bus (SPI, Dual SPI, Quad SPI). The data size of the access to this register can be 32-bits / 16-bits / 8-bits and is equal to the number of the transferred bits. This register has meaning only when the controller is in Manual mode. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 283: QSPIC2_DUMMYDATA_REG (0x34000020)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|---|-------|
| 31:0 | W | QSPIC_DUMMYDATA | <p>Writing to this register generates a number of clock pulses to the SPI bus. During the last clock of this activity in the SPI bus, the QSPI_IOx data pads are in hi-z state. The data size of the access to this register can be 32-bits / 16-bits/ 8-bits. The number of generated pulses is equal to: (size of AHB bus access) / (size of SPI bus). The size of SPI bus is equal to 1, 2 or 4 for Single, Dual or Quad SPI mode respectively.</p> <p>This register has meaning only when the controller is in Manual mode.</p> | 0x0 |

Table 284: QSPIC2_ERASECTRL_REG (0x34000024)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 31:28 | - | - | | 0x0 |
| 27:25 | R | QSPIC_ERS_STAT_E | <p>It shows the progress of sector/block erasing (read only).</p> <p>0x0: No Erase. 0x1: Pending erase request 0x2: Erase procedure is running 0x3: Suspended Erase procedure 0x4: Finishing the Erase procedure 0x5..0x7: Reserved</p> | 0x0 |
| 24 | R/W | QSPIC_ERASE_EN | <p>This bit has meaning only when the external device is a serial FLASH (QSPIC_SRAM_EN=0).</p> <p>During Manual mode (QSPIC_AUTO_MD = 0) : This bit is in read only mode.</p> <p>During Auto mode (QSPIC_AUTO_MD = 1). To request the erasing of the block/sector (QSPIC_ERS_ADDR, 12'b0) write 1 to this bit. This bit is cleared automatically with the end of the erasing. Until the end of erasing the QSPIC_ERASE_EN remains in read only mode.</p> <p>During the same period of time the controller remains in Auto Mode (QSPIC_AUTO_MD goes in read only mode).</p> <p>In the case where the external device is a serial SRAM (QSPIC_SRAM_EN=1) this bit is in read only mode.</p> | 0x0 |
| 23:4 | R/W | QSPIC_ERS_ADDR | <p>Defines the address of the block/sector that is requested to be erased.</p> <p>If QSPIC_USE_32BA = 0 (24 bits addressing), bits QSPIC_ERASECTRL_REG[23-12] determine the block/ sector address bits [23-12].</p> <p>QSPIC_ERASECTRL_REG[11-4] are ignored by the controller.</p> <p>If QSPIC_USE_32BA = 1 (32 bits addressing) bits QSPIC_ERASECTRL_REG[23-4] determine the block / sectors address bits [31:12]</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------|-------|
| 3:0 | - | - | | 0x0 |

Table 285: QSPIC2_ERASECMDA_REG (0x34000028)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|--|-------|
| 31:24 | R/W | QSPIC_RES_INST | The code value of the erase resume instruction | 0x0 |
| 23:16 | R/W | QSPIC_SUS_INST | The code value of the erase suspend instruction. | 0x0 |
| 15:8 | R/W | QSPIC_WEN_INST | The code value of the write enable instruction. | 0x0 |
| 7:0 | R/W | QSPIC_ERS_INST | The code value of the erase instruction. | 0x0 |

Table 286: QSPIC2_ERASECMDDB_REG (0x3400002C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 31:30 | - | - | | 0x0 |
| 29:24 | R/W | QSPIC_RESSUS_DLY | Defines a timer that counts the minimum allowed delay between an erase suspend command and the previous erase resume command (or the initial erase command). 0x00: Dont wait. The controller starts immediately to suspend the erase procedure. 0x01..0x3F: The controller waits for at least this number of 288 KHz clock cycles before the suspension of erasing. Time starts counting after the end of the previous erase resume command (or the initial erase command) | 0x0 |
| 23:20 | - | - | | 0x0 |
| 19:16 | R/W | QSPIC_ERSRES_HLD | The controller must stay without flash memory reading requests for this number of AMBA hclk clock cycles, before to perform the command of erase or erase resume. Allowable range : 0xF - 0x0 | 0x0 |
| 15 | - | - | | 0x0 |
| 14:10 | R/W | QSPIC_ERS_CS_HI | After the execution of instructions: write enable, erase, erase suspend and erase resume, the QSPI_CS remains high for at least this number of QSPI_SCK clock cycles. | 0x0 |
| 9:8 | R/W | QSPIC_EAD_TX_M | The mode of the SPI Bus during the address phase of the erase instruction 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 7:6 | R/W | QSPIC_RES_TX_M | The mode of the SPI Bus during the transmission of the resume instruction 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 5:4 | R/W | QSPIC_SUS_TX_M | The mode of the SPI Bus during the transmission of | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|--|-------|
| | | D | the suspend instruction. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | |
| 3:2 | R/W | QSPIC_WEN_TX_M D | The mode of the SPI Bus during the transmission of the write enable instruction. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 1:0 | R/W | QSPIC_ERS_TX_M D | The mode of the SPI Bus during the instruction phase of the erase instruction 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |

Table 287: QSPIC2_BURSTBRK_REG (0x34000030)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|---|-------|
| 31:21 | - | - | | 0x0 |
| 20 | R/W | QSPIC_SEC_HF_D S | Disable output during the transmission of the second half (QSPIC_BRK_WRD[3:0]). Setting this bit is only useful if QSPIC_BRK_EN =1 and QSPIC_BRK_SZ= 1. 0: The controller drives the SPI bus during the transmission of the QSPIC_BRK_WRD[3:0]. 1: The controller leaves the SPI bus in Hi-Z during the transmission of the QSPIC_BRK_WORD[3:0]. | 0x0 |
| 19:18 | R/W | QSPIC_BRK_TX_M D | The mode of the SPI Bus during the transmission of the read break sequence. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 17 | R/W | QSPIC_BRK_SZ | The size of the read break sequence. 0: One byte (Send QSPIC_BRK_WRD[15:8]) 1: Two bytes (Send QSPIC_BRK_WRD[15:0]) | 0x0 |
| 16 | R/W | QSPIC_BRK_EN | Controls the application of a special command (read break sequence) that is used in order to force the device to abandon the continuous read mode. 0: The special command is not applied 1: The special command is applied This special command is applied by the controller to the external device under the following conditions: - the controller is in Auto mode - the QSPIC_INST_MD = 1 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| | | | - the previous command that has been applied in the external device was read - the controller want to apply to the external device a command different than the read. | |
| 15:0 | R/W | QSPIC_BRK_WRD | This is the value of a special command (read break sequence) that is applied by the controller to the external memory device, in order to force the memory device to abandon the continuous read mode. | 0x0 |

Table 288: QSPIC2_STATUSCMD_REG (0x34000034)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:23 | - | - | | 0x0 |
| 22 | R/W | QSPIC_STSDLY_SEL | Defines the timer which is used to count the delay that it has to wait before to read the FLASH Status Register, after an erase or an erase resume command. 0: The delay is controlled by the QSPIC_RESSTS_DLY which counts on the qspi clock. 1: The delay is controlled by the QSPIC_RESSUS_DLY which counts on the 288 kHz clock. | 0x0 |
| 21:16 | R/W | QSPIC_RESSTS_DLY | Defines a timer that counts the minimum required delay between the reading of the status register and of the previous erase or erase resume instruction. 0x00: Dont wait. The controller starts to reading the Flash memory status register immediately. 0x01..0x3F: The controller waits for at least this number of QSPI_CLK cycles and afterwards it starts to reading the Flash memory status register. The timer starts to count after the end of the previous erase or erase resume command. The actual timer that will be used by the controller before the reading of the Flash memory status register is defined by the QSPIC_STSDLY_SEL. | 0x0 |
| 15 | R/W | QSPIC_BUSY_VAL | Defines the value of the Busy bit which means that the flash is busy. 0: The flash is busy when the Busy bit is equal to 0. 1: The flash is busy when the Busy bit is equal to 1. | 0x0 |
| 14:12 | R/W | QSPIC_BUSY_POS | Defines the bit of the Flash status register which represents the Busy bit (0x7 - 0x0). | 0x0 |
| 11:10 | R/W | QSPIC_RSTAT_RX_MD | The mode of the SPI Bus during the reception phase of the read status instruction, where the value of status register is retrieved. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 9:8 | R/W | QSPIC_RSTAT_TX_MD | The mode of the SPI Bus during the instruction phase of the read status instruction. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|--|-------|
| | | | 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | |
| 7:0 | R/W | QSPIC_RSTAT_INS_T | The code value of the read status instruction. It is transmitted during the instruction phase of the read status instruction. | 0x0 |

Table 289: QSPIC2_CHCKERASE_REG (0x34000038)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|---|-------|
| 31:0 | W | QSPIC_CHCKERASE | Writing any value to this register during erasing, forces the controller to read the flash memory status register. Depending on the value of the Busy bit, it updates the QSPIC_ERASE_EN. This register has meaning only when the controller is in Auto mode and there is an erase in progress (QSPIC_ERASE_EN =1). It has no meaning when the external device is a serial SRAM. | 0x0 |

Table 290: QSPIC2_GP_REG (0x3400003C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|--|-------|
| 4:3 | R/W | QSPIC_PADS_SLEW | QSPI pads slew rate control. Indicative values under certain conditions: 0x0 : Rise=1.7 V/ns, Fall=1.9 V/ns (weak) 0x1 : Rise=2.0 V/ns, Fall=2.3 V/ns 0x2 : Rise=2.3 V/ns, Fall=2.6 V/ns 0x3 : Rise=2.4 V/ns, Fall=2.7 V/ns (strong) Conditions: FLASH pin capacitance 6pF, Vcc=1.8V, T=25C and Idrive=16mA | 0x0 |
| 2:1 | R/W | QSPIC_PADS_DRV | QSPI pads drive current 0x0 : 4 mA 0x1 : 8 mA 0x2 : 12 mA 0x3 : 16 mA | 0x0 |
| 0 | R/W | - | | 0x0 |

Table 291: QSPIC2_AWRITECMD_REG (0x34000040)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------------|---|-------|
| 31:19 | - | - | | 0x0 |
| 18:14 | R/W | QSPIC_WR_CS_HI_GH_MIN | After the execution of the write command, the QSPI_CS remains high for at least this number of QSPI_SCK clock cycles. | 0x0 |
| 13:12 | R/W | QSPIC_WR_DAT_T_X_MD | The mode of the SPI Bus during the data phase of the write command. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------|--|-------|
| | | | 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | |
| 11:10 | R/W | QSPIC_WR_ADR_T_X_MD | The mode of the SPI Bus during the address phase of the write command. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 9:8 | R/W | QSPIC_WR_INST_T_X_MD | The mode of the SPI Bus during the instruction phase of the write command. 0x0: Single SPI 0x1: Dual 0x2: Quad 0x3: Reserved | 0x0 |
| 7:0 | R/W | QSPIC_WR_INST | This is the value of the instruction that is used, in order to be programmed the external SRAM device. | 0x0 |

Table 292: QSPIC2_MEMBLEN_REG (0x34000044)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|---|-------|
| 31:14 | - | - | | 0x0 |
| 13:4 | R/W | QSPIC_T_CEM_CC | <p>Defines the maximum allowed time tCEM for which the QSPI_CS can stay active (QSPI_CS=0). It has meaning only when QSPIC_T_CEM_EN is equal to 1. See also the description of the QSPIC_T_CEM_EN for more details.</p> <p>The tCEM is expressed in number of qspi clock cycles and can be calculated as follows :</p> $\text{tCEM} / (\text{qspi_clock_period})$ <p>If the result of the above equation is higher than 0x3FF, use the value 0x3FF.</p> | 0x0 |
| 3 | R/W | QSPIC_T_CEM_EN | <p>This bit enables the controlling of the maximum time tCEM for which the QSPI_CS remains active. It has meaning only when the Auto mode is active (QSPIC_AUTO_MD=1) and the external device is a serial SRAM (QSPIC_SRAM_EN=1). In the case where the external device is a serial Flash (QSPIC_SRAM_EN=0) or the controller is in Manual mode (QSPIC_AUTO_MD=0), this field has no any effect.</p> <p>This feature is useful in the case where the external serial device is a dynamic RAM that requires refresh. If the refresh is applied only when the device is in the idle state (QSPI_CS = 1), the time for which the device remains in the active state (QSPI_CS = 0) should be limited by a maximum</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| | | | <p>threshold.</p> <p>0:There is no any constraint regarding the maximum allowed time for which the QSPI_CS can stay active. This is the case also when QSPIC_SRAM_EN=0 or QSPIC_AUTO_MD=0.</p> <p>1:There is a maximum allowed time interval tCEM for which the QSPI_CS can stay active during a burst access (for reading or writting of data). For the controller this is considered as equal to QSPIC_T_CEM_CC x qspi_clock_period. In the case where the data transfer requires the QSPI_CS to stays active for more than QSPIC_T_CEM_CC qspi clock cycles, the QSPI controller splits the access on the SPI bus in more than one bursts, by inserting inactive periods (QSPI_CS = 0) between them. This will cost extra clock cycles for the realization of the original acceess, due to the additional commands that are required in the SPI bus.</p> <p>The value in the QSPIC_T_CEM_CC should be updated every time where the frequency of the qspi clock is modified. The qspi clock frequency should not be decreased more than a lowest frequency. This is the lowest frequency that enables the be performed a 32-bit word read and write access, without violating the tCEM timing requirement (the QSPI controller allows to be performed at least the transferring of one beat of the requested burst, independent of the QSPIC_T_CEM_CC limit).</p> | |
| 2:0 | R/W | QSPIC_MEMBLEN | <p>In this register is defined the expected behavior of the external memory device regarding the length of a burst operation :</p> <p>0x0: The external memory device is capable to implement incremental burst of unspecified length.</p> <p>0x1: The external memory device implements a wrapping burst of length 4 bytes.</p> <p>0x2: The external memory device implements a wrapping burst of length 8 bytes.</p> <p>0x3: The external memory device implements a wrapping burst of length 16 bytes.</p> <p>0x4: The external memory device implements a wrapping burst of length 32 bytes.</p> <p>0x5: The external memory device implements a wrapping burst of length 64 bytes.</p> <p>0x6 - 0x7 : Reserved</p> <p>This setting is used by the QSPI controller when the Auto mode is enabled (QSPIC_AUTO_MD=1), in order to handle the various burst requests of the AHB bus, in respect of the requirements of the external memory device.</p> <p>The external memory device may need to be configured by applying special instruction, in order to be defined the kind of the burst operation. This can be implemented by applying this special</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | instruction with the QSPI controller in Manual mode (QSPIC_AUTO_MD=1). Refer to the datasheet of the external device for more information. | |

42.9 RF Monitor Registers

Table 293: Register map RFPT

| Address | Register | Description |
|------------|--------------------|--------------------------------------|
| 0x50040600 | RFMON_CTRL_REG | Control register |
| 0x50040604 | RFMON_ADDR_REG | AHB master start address |
| 0x50040608 | RFMON_LEN_REG | Data length register |
| 0x5004060C | RFMON_STAT_REG | Status register |
| 0x50040610 | RFMON_CRV_ADDR_REG | AHB master current address |
| 0x50040614 | RFMON_CRV_LEN_REG | The remaining data to be transferred |

Table 294: RFMON_CTRL_REG (0x50040600)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|--|-------|
| 31:3 | - | - | | 0x0 |
| 2 | R/W | RFMON_BREQ_FO RCE | Write this bit with 1, when the required throughput for the transferring of the captured data is close to the capacity of the system bus/memory. The controller will be aggressive in the usage of the bus. The availability of the bus will be affected for the remaining masters. | 0x0 |
| 1 | R/W | RFMON_CIRC_EN | Write with 1 to enable the circular mode. In circular mode the controller continuously writes data in to the memory until being disabled by software. Data are transferred in the circular buffer in the memory, as defined by RFMON_ADDR_REG and RFMON_LEN_REG registers. Disabling of the controller is realized by writing RFMON_PACK_EN with 0. | 0x0 |
| 0 | R/W | RFMON_PACK_EN | Starts capturing data from the test bus 0 : No data captured. 1 : Data captured. Should be written with 1 to start data acquisition. When the controller is not in circular mode (RFMON_CIRC_EN = 0) and after capturing a predefined number of words (RFMON_LEN), this bit will be auto cleared. In circular mode (RFMON_CIRC_EN = 1) the RFMON_PACK_EN remains 1 to be cleared by software. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 295: RFMON_ADDR_REG (0x50040604)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:2 | R/W | RFMON_ADDR | It is the bits [31:2] of base address that is used by the AHB master interface of the controller. Defines the AHB address where the controller will start storing data at. Bits [1:0] of the address are always considered to be 0. | 0x0 |
| 1:0 | - | - | | 0x0 |

Table 296: RFMON_LEN_REG (0x50040608)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|--|-------|
| 31:17 | - | - | | 0x0 |
| 16:0 | R/W | RFMON_LEN | The number of words (minus one) that should be captured. | 0x0 |

Table 297: RFMON_STAT_REG (0x5004060C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|---|-------|
| 31:2 | - | - | | 0x0 |
| 1 | RW1C | RFMON_OFLOW_STK | Indicates that during transfer of data, at least one overflow has been detected. 0 : The transfer completed without overflows. 1 : At least one overflow occurred in the fifo. Write 1 to clear this bit. | 0x0 |
| 0 | R | RFMON_ACTIVE | Indicates the state of the controller. 0 : The controller is idle. 1 : The controller is active. The capturing process and/or the dma activity is in progress. The controller will be activated (RFMON_ACTIVE == 1), when RFMON_PACK_EN will be written with 1. Will return to inactive state, after the end of the capturing process (RFMON_PACK_EN==0) and the completion of the transfer of all data to memory. | 0x0 |

Table 298: RFMON_CRV_ADDR_REG (0x50040610)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:2 | R | RFMON_CRV_ADDR | Bits [31:2] of AHB address that will be used by the controller in the next memory access. The bits [1:0] are always 0. | 0x0 |
| 1:0 | - | - | | 0x0 |

Table 299: RFMON_CRV_LEN_REG (0x50040614)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|-------------|-------|
| 31:17 | - | - | | 0x0 |

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 16:0 | R | RFMON_CRV_LEN | Indicates the number of words (minus 1) that remain to be transferred. | 0x0 |

42.10 Real Time Clock Registers

Table 300: Register map RTC

| Address | Register | Description |
|------------|----------------------------|----------------------------------|
| 0x50000400 | RTC_CONTROL_REG | RTC Control Register |
| 0x50000404 | RTC_HOUR_MODE_REG | RTC Hour Mode Register |
| 0x50000408 | RTC_TIME_REG | RTC Time Register |
| 0x5000040C | RTC_CALENDAR_REG | RTC Calendar Register |
| 0x50000410 | RTC_TIME_ALARM_REG | RTC Time Alarm Register |
| 0x50000414 | RTC_CALENDAR_ALARM_REG | RTC Calendar Alram Register |
| 0x50000418 | RTC_ALARM_ENABLE_REG | RTC Alarm Enable Register |
| 0x5000041C | RTC_EVENT_FLAGS_REG | RTC Event Flags Register |
| 0x50000420 | RTC_INTERRUPT_ENABLE_REG | RTC Interrupt Enable Register |
| 0x50000424 | RTC_INTERRUPT_DISABLE_REG | RTC Interrupt Disable Register |
| 0x50000428 | RTC_INTERRUPT_MASK_REG | RTC Interrupt Mask Register |
| 0x5000042C | RTC_STATUS_REG | RTC Status Register |
| 0x50000430 | RTC_KEEP_RTC_REG | RTC Keep RTC Register |
| 0x50000480 | RTC_EVENT_CTRL_REG | RTC Event Control Register |
| 0x50000484 | RTC_MOTOR_EVENT_PERIOD_REG | RTC Motor Event Period Register |
| 0x50000488 | RTC_PDC_EVENT_PERIOD_REG | RTC PDC Event Period Register |
| 0x5000048C | RTC_PDC_EVENT_CLEAR_REG | RTC PDC Event Clear Register |
| 0x50000490 | RTC_MOTOR_EVENT_CNT_REG | RTC Motor Event Counter Register |
| 0x50000494 | RTC_PDC_EVENT_COUNTER_REG | RTC PDC Event Counter Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 301: RTC_CONTROL_REG (0x50000400)

| 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 302: RTC_HOUR_MODE_REG (0x50000404)

| 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 303: RTC_TIME_REG (0x50000408)

| 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 |
| 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 | - | - | | 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 | - | - | | 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 304: RTC_CALENDAR_REG (0x5000040C)

| 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 | - | - | | 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 | - | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 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 |
| 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 305: RTC_TIME_ALARM_REG (0x50000410)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---|-------|
| 31 | - | - | | 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 | - | - | | 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 | - | - | | 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 306: RTC_CALENDAR_ALARM_REG (0x50000414)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 31:14 | R/W | - | | 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). | 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 | - | - | | 0x0 |

Table 307: RTC_ALARM_ENABLE_REG (0x50000418)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|---|-------|
| 5 | R/W | RTC_ALARM_MNT_H_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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|--|-------|
| 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 308: RTC_EVENT_FLAGS_REG (0x5000041C)

| 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 |
| 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 309: RTC_INTERRUPT_ENABLE_REG (0x50000420)

| 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|---|-------|
| 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 310: RTC_INTERRUPT_DISABLE_REG (0x50000424)

| 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 | Interrupt on hour disable. Disable to issue the interrupt when hour event occurred. | 0x0 |
| 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 311: RTC_INTERRUPT_MASK_REG (0x50000428)

| 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 | IMask 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 | IMask 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 | IMask 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 | IMask 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 | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------|-------|
| | | | Register). | |

Table 312: RTC_STATUS_REG (0x5000042C)

| 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 |
| 0 | R | RTC_VALID_TIME | Valid Time. If cleared then indicates that invalid entry occurred when writing to Time Register. | 0x1 |

Table 313: RTC_KEEP_RTC_REG (0x50000430)

| 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 314: RTC_EVENT_CTRL_REG (0x50000480)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|---|-------|
| 1 | R/W | RTC_PDC_EVENT_EN | 0 = Event to PDC is disabled. No clear any pending event 1 = Event to PDC is enabled | 0x0 |
| 0 | R/W | RTC_MOTOR_EVENT_EN | 0 = Event to Motor is disabled 1 = Event to Motor is enabled | 0x0 |

Table 315: RTC_MOTOR_EVENT_PERIOD_REG (0x50000484)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------------|---|-------|
| 11:0 | R/W | RTC_MOTOR_EVENT_PERIOD | RTC will send an event to motor (if RTC_MOTOR_EVENT_EN=1) every (RTC_MOTOR_EVENT_PERIOD+1)*10ms | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 316: RTC_PDC_EVENT_PERIOD_REG (0x50000488)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|---|-------|
| 12:0 | R/W | RTC_PDC_EVENT_PERIOD | RTC will send an event to PDC (if RTC_PDC_EVENT_EN=1) every (RTC_PDC_EVENT_PERIOD+1)*10ms | 0x0 |

Table 317: RTC_PDC_EVENT_CLEAR_REG (0x5000048C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|-------------------------------|-------|
| 0 | R | PDC_EVENT_CLEA_R | On read, PDC event is cleared | 0x0 |

Table 318: RTC_MOTOR_EVENT_CNT_REG (0x50000490)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 11:0 | R | RTC_MOTOR_EVENT_CNT | It gives the current value of the Motor event counter (0 to RTC_MOTOR_EVENT_PERIOD) | 0x0 |

Table 319: RTC_PDC_EVENT_CNT_REG (0x50000494)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|---|-------|
| 12:0 | R | RTC_PDC_EVENT_CNT | It gives the current value of the PDC event counter (0 to RTC_PDC_EVENT_PERIOD) | 0x0 |

42.11 Motor Controller Registers

Table 320: Register map SMOTOR

| Address | Register | Description |
|------------|--------------------------|--------------------------------------|
| 0x50030E00 | SMOTOR_CTRL_REG | Motor control register |
| 0x50030E04 | PG0_CTRL_REG | Pattern generator 0 control register |
| 0x50030E08 | PG1_CTRL_REG | Pattern generator 1 control register |
| 0x50030E0C | PG2_CTRL_REG | Pattern generator 2 control register |
| 0x50030E10 | PG3_CTRL_REG | Pattern generator 3 control register |
| 0x50030E14 | PG4_CTRL_REG | Pattern generator 4 control register |
| 0x50030E18 | SMOTOR_TRIGGER_REG | Motor controller trigger register |
| 0x50030E20 | SMOTOR_CMD_FIFO_REG | Motor control command FIFO register |
| 0x50030E24 | SMOTOR_CMD_READ_PTR_REG | Command read pointer register |
| 0x50030E28 | SMOTOR_CMD_WRITE_PTR_REG | Command write pointer register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|----------------------------------|-----------------------------------|
| 0x50030E2C | SMOTOR_STATUS_R _{EG} | Motor controller status register |
| 0x50030E30 | SMOTOR_IRQ_CLEAR _{_REG} | Motor control IRQ clear register |
| 0x50030E40 | WAVETABLE_BASE | Base address of the wavetable |
| 0x50030E80 | CMD_TABLE_BASE | Base address of the command table |

Table 321: SMOTOR_CTRL_REG (0x50030E00)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------------|--|-------|
| 28 | R/W | TRIG_RTC_EVENT_EN | 0 = RTC event does not trigger command pop 1 = RTC event triggers command pop | 0x0 |
| 27 | R/W | MC_LP_CLK_TRIG_EN | 0 = Divided sleep clock does not trigger command pop 1 = Divided sleep clock triggers command pop | 0x0 |
| 26 | R/W | SMOTOR_THRESH_OLD_IRQ_EN | IRQ in the event of the FIFO level (write pointer - read pointer) reaching, or is below the threshold determined by SMOTOR_THRESHOLD. 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 25:21 | R/W | SMOTOR_THRESH_OLD | Determines the FIFO level (write pointer - read pointer) at or below which an IRQ can be triggered using SMOTOR_THRESHOLD_IRQ_EN. | 0x0 |
| 20 | R/W | SMOTOR_FIFO_UNR_IRQ_EN | IRQ in the event of FIFO underrun: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 19 | R/W | SMOTOR_FIFO_OVF_IRQ_EN | IRQ in the event of FIFO overflow: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 18 | R/W | SMOTOR_GENEND_IRQ_EN | IRQ in the event a pattern generator (configured to do so through its corresponding GENEND_IRQ_EN bit) has ended generating a pattern: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 17 | R/W | SMOTOR_GENSTART_IRQ_EN | IRQ in the event a pattern generator (configured to do so through its corresponding GENSTART_IRQ_EN bit) has just started generating a pattern: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 16:7 | R/W | SMOTOR_MOI | Idle time of a PG after generating a waveform. A PG will remain busy for the last signal's MOI to finish. | 0x0 |
| 6:1 | R/W | CYCLIC_SIZE | Depth of the cyclic buffer, only valid if CYCLIC_MODE is 1. | 0x0 |
| 0 | R/W | CYCLIC_MODE | Determines operation mode of command FIFO: 0 = Normal FIFO mode 1 = Cyclic buffer mode, CYCLIC_SIZE determines | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--------------|-------|
| | | | buffer depth | |

Table 322: PG0_CTRL_REG (0x50030E04)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 15 | R/W | GENEND_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it is done generating a pattern. It is only valid if SMOTOR_GENEND_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 14 | R/W | GENSTART_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it starts generating a pattern. It is only valid if SMOTOR_GENSTART_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 13 | R/W | PG_START_MODE | 0 = Auto start mode: pattern generator will start whenever all enabled signals have received a command 1 = Manual start mode: pattern generator will only start if it has been given a PG_START, and all enabled signals have received a command | 0x0 |
| 12 | R/W | PG_MODE | 0 = Flex mode 1 = Pair mode | 0x0 |
| 11 | R/W | SIG3_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 10 | R/W | SIG2_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 9 | R/W | SIG1_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 8 | R/W | SIG0_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 7:6 | R/W | OUT3_SIG | Selects which signal is routed to the output. | 0x3 |
| 5:4 | R/W | OUT2_SIG | Selects which signal is routed to the output. | 0x2 |
| 3:2 | R/W | OUT1_SIG | Selects which signal is routed to the output. | 0x1 |
| 1:0 | R/W | OUT0_SIG | Selects which signal is routed to the output. | 0x0 |

Table 323: PG1_CTRL_REG (0x50030E08)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| 15 | R/W | GENEND_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it is done generating a pattern. It is only valid if SMOTOR_GENEND_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 14 | R/W | GENSTART_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it starts generating a pattern. It is only valid if SMOTOR_GENSTART_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 13 | R/W | PG_START_MODE | 0 = Auto start mode: pattern generator will start whenever all enabled signals have received a command 1 = Manual start mode: pattern generator will only start if it has been given a PG_START, and all enabled signals have received a command | 0x0 |
| 12 | R/W | PG_MODE | 0 = Flex mode 1 = Pair mode | 0x0 |
| 11 | R/W | SIG3_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 10 | R/W | SIG2_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 9 | R/W | SIG1_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 8 | R/W | SIG0_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 7:6 | R/W | OUT3_SIG | Selects which signal is routed to the output. | 0x3 |
| 5:4 | R/W | OUT2_SIG | Selects which signal is routed to the output. | 0x2 |
| 3:2 | R/W | OUT1_SIG | Selects which signal is routed to the output. | 0x1 |
| 1:0 | R/W | OUT0_SIG | Selects which signal is routed to the output. | 0x0 |

Table 324: PG2_CTRL_REG (0x50030E0C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 15 | R/W | GENEND_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it is done generating a pattern. It is only valid if SMOTOR_GENEND_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 14 | R/W | GENSTART_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it starts generating a pattern. It is only valid if SMOTOR_GENSTART_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 13 | R/W | PG_START_MODE | 0 = Auto start mode: pattern generator will start whenever all enabled signals have received a command 1 = Manual start mode: pattern generator will only start if it has been given a PG_START, and all enabled signals have received a command | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| 12 | R/W | PG_MODE | 0 = Flex mode 1 = Pair mode | 0x0 |
| 11 | R/W | SIG3_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 10 | R/W | SIG2_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 9 | R/W | SIG1_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 8 | R/W | SIG0_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 7:6 | R/W | OUT3_SIG | Selects which signal is routed to the output. | 0x3 |
| 5:4 | R/W | OUT2_SIG | Selects which signal is routed to the output. | 0x2 |
| 3:2 | R/W | OUT1_SIG | Selects which signal is routed to the output. | 0x1 |
| 1:0 | R/W | OUT0_SIG | Selects which signal is routed to the output. | 0x0 |

Table 325: PG3_CTRL_REG (0x50030E10)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 15 | R/W | GENEND_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it is done generating a pattern. It is only valid if SMOTOR_GENEND_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 14 | R/W | GENSTART_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it starts generating a pattern. It is only valid if SMOTOR_GENSTART_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 13 | R/W | PG_START_MODE | 0 = Auto start mode: pattern generator will start whenever all enabled signals have received a command 1 = Manual start mode: pattern generator will only start if it has been given a PG_START, and all enabled signals have received a command | 0x0 |
| 12 | R/W | PG_MODE | 0 = Flex mode 1 = Pair mode | 0x0 |
| 11 | R/W | SIG3_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 10 | R/W | SIG2_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 9 | R/W | SIG1_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 8 | R/W | SIG0_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| 7:6 | R/W | OUT3_SIG | Selects which signal is routed to the output. | 0x3 |
| 5:4 | R/W | OUT2_SIG | Selects which signal is routed to the output. | 0x2 |
| 3:2 | R/W | OUT1_SIG | Selects which signal is routed to the output. | 0x1 |
| 1:0 | R/W | OUT0_SIG | Selects which signal is routed to the output. | 0x0 |

Table 326: PG4_CTRL_REG (0x50030E14)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 15 | R/W | GENEND_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it is done generating a pattern. It is only valid if SMOTOR_GENEND_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 14 | R/W | GENSTART_IRQ_EN | Determines if the corresponding pattern generator will contribute to the generation of the IRQ when it starts generating a pattern. It is only valid if SMOTOR_GENSTART_IRQ_EN is enabled: 0 = Interrupt requests disabled 1 = Interrupt requests enabled | 0x0 |
| 13 | R/W | PG_START_MODE | 0 = Auto start mode: pattern generator will start whenever all enabled signals have received a command 1 = Manual start mode: pattern generator will only start if it has been given a PG_START, and all enabled signals have received a command | 0x0 |
| 12 | R/W | PG_MODE | 0 = Flex mode 1 = Pair mode | 0x0 |
| 11 | R/W | SIG3_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 10 | R/W | SIG2_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 9 | R/W | SIG1_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 8 | R/W | SIG0_EN | 0 = Signal disabled 1 = Signal enabled | 0x1 |
| 7:6 | R/W | OUT3_SIG | Selects which signal is routed to the output. | 0x3 |
| 5:4 | R/W | OUT2_SIG | Selects which signal is routed to the output. | 0x2 |
| 3:2 | R/W | OUT1_SIG | Selects which signal is routed to the output. | 0x1 |
| 1:0 | R/W | OUT0_SIG | Selects which signal is routed to the output. | 0x0 |

Table 327: SMOTOR_TRIGGER_REG (0x50030E18)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| 5 | R0/W | PG4_START | Writing 1 to this bit will start PG4, only effective in manual mode. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| 4 | R0/W | PG3_START | Writing 1 to this bit will start PG3, only effective in manual mode. | 0x0 |
| 3 | R0/W | PG2_START | Writing 1 to this bit will start PG2, only effective in manual mode. | 0x0 |
| 2 | R0/W | PG1_START | Writing 1 to this bit will start PG1, only effective in manual mode. | 0x0 |
| 1 | R0/W | PG0_START | Writing 1 to this bit will start PG0, only effective in manual mode. | 0x0 |
| 0 | R0/W | POP_CMD | Writing 1 will pop one (or more, depending on the N_CMDS field of the first) command(s) from the command buffer into its corresponding pattern generator. | 0x0 |

Table 328: SMOTOR_CMD_FIFO_REG (0x50030E20)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|--|-------|
| 15:0 | W | SMOTOR_CMD_FIF O | Writing to this address will push a command into the command FIFO. | 0x0 |

Table 329: SMOTOR_CMD_READ_PTR_REG (0x50030E24)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------|--|-------|
| 5:0 | R | SMOTOR_CMD_RE AD_PTR | Pointer to the next command to be popped from the FIFO. The command at SMOTOR_CMD_READ_PTR-1 is the last command that has been popped from the FIFO into its corresponding PG. | 0x0 |

Table 330: SMOTOR_CMD_WRITE_PTR_REG (0x50030E28)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------|--|-------|
| 5:0 | R/W | SMOTOR_CMD_WR ITE_PTR | Pointer to the location in the FIFO where the next command will be pushed at. The last command pushed to the FIFO is at SMOTOR_CMD_WRITE_PTR - 1. Can only be changed in cyclic mode | 0x0 |

Table 331: SMOTOR_STATUS_REG (0x50030E2C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| 9 | R | PG4_BUSY | Tells whether the PG is busy/generating a waveform. | 0x0 |
| 8 | R | PG3_BUSY | Tells whether the PG is busy/generating a waveform. | 0x0 |
| 7 | R | PG2_BUSY | Tells whether the PG is busy/generating a waveform. | 0x0 |
| 6 | R | PG1_BUSY | Tells whether the PG is busy/generating a waveform. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------|--|-------|
| | | | waveform. | |
| 5 | R | PG0_BUSY | Tells whether the PG is busy/generating a waveform. | 0x0 |
| 4 | R | THRESHOLD_IRQ_STATUS | Tells whether the THRESHOLD_IRQ fired. Can be cleared with corresponding _CLEAR bit. | 0x0 |
| 3 | R | FIFO_UNR_IRQ_STATUS | Tells whether the FIFO_UNR_IRQ fired. Can be cleared with corresponding _CLEAR bit. | 0x0 |
| 2 | R | FIFO_OVF_IRQ_STATUS | Tells whether the FIFO_OVF_IRQ fired. Can be cleared with corresponding _CLEAR bit. | 0x0 |
| 1 | R | GENEND_IRQ_STATUS | Tells whether the GENEND_IRQ fired. Can be cleared with corresponding _CLEAR bit. | 0x0 |
| 0 | R | GENSTART_IRQ_STATUS | Tells whether the GENSTART_IRQ fired. Can be cleared with corresponding _CLEAR bit. | 0x0 |

Table 332: SMOTOR_IRQ_CLEAR_REG (0x50030E30)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|--------------------------------------|-------|
| 4 | R0/W | THRESHOLD_IRQ_CLEAR | Clears the THRESHOLD_IRQ_STATUS bit. | 0x0 |
| 3 | R0/W | FIFO_UNR_IRQ_CLEAR | Clears the FIFO_UNR_IRQ_STATUS bit. | 0x0 |
| 2 | R0/W | FIFO_OVF_IRQ_CLEAR | Clears the FIFO_OVF_IRQ_STATUS bit. | 0x0 |
| 1 | R0/W | GENEND_IRQ_CLEAR | Clears the GENEND_IRQ_STATUS bit. | 0x0 |
| 0 | R0/W | GENSTART_IRQ_CLEAR | Clears the GENSTART_IRQ_STATUS bit. | 0x0 |

Table 333: WAVETABLE_BASE (0x50030E40)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|-------------|-------|
| 28:24 | - | - | | 0x0 |
| 20:16 | - | - | | 0x0 |
| 12:8 | - | - | | 0x0 |
| 4:0 | - | - | | 0x0 |

Table 334: CMD_TABLE_BASE (0x50030E80)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|-------------|-------|
| 31:0 | - | - | | N/A |

42.12 Sensor Node Controller Registers

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 335: Register map SNC

| Address | Register | Description |
|------------|------------------|---|
| 0x50020C00 | SNC_CTRL_REG | Sensor Node Control Register |
| 0x50020C04 | SNC_STATUS_REG | Sensor Node Status Register |
| 0x50020C08 | SNC_LP_TIMER_REG | Sensor Node Low-Power Timer Register |
| 0x50020C0C | SNC_PC_REG | Sensor Node Program Counter |
| 0x50020C10 | SNC_R1_REG | Sensor Node core - Operand 1 Register |
| 0x50020C14 | SNC_R2_REG | Sensor Node core - Operand 2 Register |
| 0x50020C18 | SNC_TMP1_REG | Sensor Node core - Temporary Register 1 |
| 0x50020C1C | SNC_TMP2_REG | Sensor Node core - Temporary Register 2 |

Table 336: SNC_CTRL_REG (0x50020C00)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:9 | R | - | | 0x0 |
| 8 | R/W | SNC_IRQ_ACK | <p>When set, the specific bit-field auto-clears the SNC_IRQ_EN field, if the latter is already set. By this way, the IRQ line towards either the CM33 and/or the PDC is cleared. Hence, the CM33 should set this bit-field as soon as it captures the interrupt from the Sensor Node.</p> <p>Note: Any SW writes to this bit-field will be discarded if the SNC_IRQ_EN bit-field is not set. It is finally noted that the SNC_IRQ_ACK bit-field is also auto-clear and it is de-asserted together with SNC_IRQ_EN.</p> | 0x0 |
| 7:6 | R/W | SNC_IRQ_CONFIG | <p>The specific bit-field determines if the IRQ line of the Sensor Node will be routed towards either the host processor (CM33) or the Power Domains Controller (PDC), or to both of them, according to the following configuration:</p> <p>0x0 = Neither the CM33 nor the PDC are triggered, both IRQ lines are low regardless of the value of SNC_IRQ_EN bit-field.</p> <p>0x1 = CM33 should be triggered, provided that SNC_IRQ_EN is set</p> <p>0x2 = PDC should be triggered, provided that SNC_IRQ_EN is set</p> <p>0x3 = Both CM33 and PDC should be triggered, provided that SNC_IRQ_EN is set</p> <p>Note: It must be noted that the specific bit-field is locked after set the SNC_IRQ_EN field of the same register. Hence, the SNC IRQ configuration cannot be changed after the IRQ bit-field is set and before the IRQ is acknowledged (by CM33). It is also noted that after having set SNC_IRQ_EN via SW, the specific bit-field can be de-asserted only by setting the SNC_IRQ_ACK bit-field (see also the description of this bit-field, also residing in SNC_CTRL_REG).</p> | 0x0 |
| 5 | R/W | SNC_IRQ_EN | When set, the specific bit-field may generate a (level-sensitive) IRQ to trigger either the host processor (CM33) or the Power Domains Controller | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------|---|-------|
| | | | (PDC) or both, depending on the configuration set in the SNC_IRQ_CONFIG bit-field of SNC_CTRL_REG. As soon as the SNC_IRQ_EN is set, it can be cleared only by setting the SNC_IRQ_ACK bit-field. | |
| 4 | R/W | SNC_BRANCH_LO_OP_INIT | When set, it clears the value of the counter used in the Sensor Node's branch command (COBR), when performing an iterative branch of up to 128 times. This bit-field is auto-cleared with the next SNC clock. | 0x0 |
| 3 | R/W | SNC_RESET | This is the Sensor Node Controller's synchronous clear bit-field. When set, it resets the state of the Sensor Node Controller and sets back its program counter (SNC_PC_REG) to the programmed base address, as determined by SNC_BASE_REG register (located in memory controller). This bit-field is auto-cleared with the next SNC clock. Note: Setting this bit-field may interrupt the Sensor Node's regular execution and any command currently being executed may be abnormally terminated. | 0x0 |
| 2 | R/W | BUS_ERROR_DETECT_EN | When set, it enables the detection of system bus errors that may occur in case a non-mapped address is used by the Sensor Node controller, when performing a register access. Note: In case of a bus error detection, the Sensor Node will set to '1' the BUS_ERROR_STATUS bit-field of SNC_STATUS_REG and will continue normally to the next command. | 0x0 |
| 1 | R/W | SNC_SW_CTRL | When set, this bit-field bypasses the enable of Sensor Node that comes from the PDC. In this mode, the Sensor Node can be started and stopped via the SNC_EN bit-field of SNC_CTRL_REG. Note: This mode is suggested to be used for debugging purposes. Also, the base address of the Sensor Node should have been programmed to the target value, before this bit-field is set. | 0x0 |
| 0 | R/W | SNC_EN | Sensor Node Controller's enable bit-field. When set, it may activate the Sensor Node, provided that the SNC_SW_CTRL bit-field is also set. If not, then the specific bit-field is not effective and Sensor Node's actual enable is controlled by the Power Domains Controller (PDC). Note: When SNC_SW_CTRL bit-field is set, the Sensor Node is controlled by the user. Thus, in that mode, it can be started and stopped by setting and resetting the SNC_EN field. When SNC_EN is reset, the Sensor Node will first complete the last on-going command before being halted. | 0x0 |

Table 337: SNC_STATUS_REG (0x50020C04)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:7 | R | - | | 0x0 |
| 6 | R | SNC_PC_LOADED | 0 : Sensor node's program counter is controlled by the Sensor Node's FSM, incremented by 4 after the | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|--|-------|
| | | | <p>fetching of each 32-bit command word.</p> <p>1 : Sensor node's program counter is loaded with a new value. The assertion of this signal requires the Sensor Node to have been first stopped, so the user must first check that the SNC_IS_STOPPED bit-field of this register is asserted, before writing the program counter. The SNC_PC_LOADED bit-field is auto-clear and it is reset to '0' as soon as the user has re-started the Sensor Node.</p> <p>Note: To start and stop the Sensor Node manually, the SNC_SW_CTRL and SNC_EN bit-fields of SNC_CTRL_REG must have been set by the user. This mode of operation is bypassing the Power Domains Controller and it is to be used for debugging purposes.</p> | |
| 5 | R | SNC_IS_STOPPED | <p>0 : Sensor Node is operational and its FSM is running.</p> <p>1 : Sensor Node is stopped and its FSM is halted. To leave this state, the SNC_EN bit-field of SNC_CTRL_REG must be set, provided that the SNC_SW_CTRL bit-field of the same register is also set. This mode is used for debugging purposes, bypassing the enable of SNC coming from the Power Domains Controller.</p> <p>Note: The SNC_PC_REG register can be modified by SW if and only if the SNC_IS_STOPPED bit is set. Otherwise, the writes to SNC_PC_REG are discarded.</p> | 0x1 |
| 4 | R | HARD_FAULT_STATUS | <p>0 : No opcode error has occurred, Sensor Node continues normally.</p> <p>1 : An opcode error has occurred. Sensor Node will continue its execution, but will set also the specific bit-field to '1', for debugging purposes.</p> <p>Note: After being set, this bit-field will be cleared only when the Sensor Node is re-initialized, by starting again from its base address. The latter can happen either by activating the SNC_RESET bit-field of SNC_CTRL_REG or by stopping and starting again the Sensor Node. This is possible only when the PDC is bypassed, so when the Sensor Node is controlled by SNC_EN and SNC_SW_CTRL bit-fields of SNC_CTRL_REG.</p> | 0x0 |
| 3 | R | BUS_ERROR_STATUS | <p>0 : No system bus error detected, Sensor Node continues normally</p> <p>1 : Bus error occurred. Sensor Node will continue, but it will also set the specific flag, which can be used for debugging purposes.</p> <p>Note: This bit-field will be reset to '0' only when the Sensor Node is re-initialized, by starting again from its base address.</p> | 0x0 |
| 2 | R | SNC_DONE_STATUS | <p>0 : Sensor Node has not yet completed the target program's execution.</p> <p>1 : Sensor Node has completed the target program's execution. Together with the update of the status bit, a pulse is also generated to notify the PDC that the Sensor Node is done.</p> <p>Note: This bit-field is set only when the "SLP" (sleep) command is executed, which should be</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|--|-------|
| | | | <p>issued after the completion of all pending tasks of the Sensor Node. It will be reset to '0' only when the Sensor Node re-starts, by executing from the base address.</p> <p>This can be done by either toggling (de-asserting and re-asserting again) the SNC_EN bit-field of SNC_CTRL_REG, if the SNC is controlled by SW, or by just re-setting the SNC state via the SNC_RESET bit-field of the same register.</p> | |
| 1 | R/W | GR_FLAG | <p>Sensor Node's 'GR' (greater) flag. It can be modified either by the Sensor Node's core (by executing an "RDCGR" command) or by the Sensor Node's microcode, when the latter directly modifies the specific bit-field of SNC_STATUS_REG.</p> <p>When the Sensor Node's FSM is in its initial state (which may happen either by switching-on the PD_COM power domain or by resetting the Sensor Node via SNC_CTRL_REG.SNC_RESET), the specific bit-field is kept to '0', for initialization purposes.</p> <p>When the Sensor Node is stopped and then reset, the Sensor Node's FSM is not in its initial state and in that case, the GR_FLAG bit-field should be reset by the user (if the application needs this to be initialized to '0'). Otherwise, it can be left as it is, until being updated by the Sensor Node itself (upon executing an "RDCGR" command).</p> <p>In general, however, this bit-field should not be modified by either the host processor (CM33) or the CMAC processor (CM0+), and especially when the Sensor Node is enabled and operational.</p> | 0x0 |
| 0 | R/W | EQ_FLAG | <p>Sensor Node's 'EQ' (equalhigh) flag. It can be modified either by the Sensor Node's core (by executing an "RDCBI" command) or by the Sensor Node's microcode, when the latter directly modifies the specific bit-field of SNC_STATUS_REG.</p> <p>When the Sensor Node's FSM is in its initial state (which may happen either by switching-on the PD_COM power domain or by resetting the Sensor Node via SNC_CTRL_REG.SNC_RESET), the specific bit-field is kept to '0', for initialization purposes.</p> <p>When the Sensor Node is stopped and then reset, the Sensor Node's FSM is not in its initial state and in that case, the EQ_FLAG bit-field should be reset by the user (if the application needs this to be initialized to '0'). Otherwise, it can be left as it is, until being updated by the Sensor Node itself (upon executing an "RDCBI" command).</p> <p>In general, however, this bit-field should not be modified by either the host processor (CM33) or the CMAC processor (CM0+), and especially when the Sensor Node is enabled and operational.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 338: SNC_LP_TIMER_REG (0x50020C08)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:8 | R | - | | 0x0 |
| 7:0 | R | LP_TIMER | This bit-field returns the current value of the Sensor Node's 8-bit timer, running with the low-power clock and may be used for debugging purposes. The specific timer is used to implement a delay of up to 256 ticks of the low-power clock. | 0x0 |

Table 339: SNC_PC_REG (0x50020C0C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|---|-------|
| 31:19 | R | - | | 0x400 |
| 18:2 | R/W | PC_REG | <p>This bit-field returns the Sensor Node's program counter bits [18:2], which at the same time is the program counter's offset from the starting address of SYSRAM (0x20.000.000), and it can be set by the user, as soon as Sensor Node has been stopped.</p> <p>The 13 MSBs of the program counter are tied to '0x400', since the Sensor Node always executes from SYSRAM, while its 2 LSBs are always tied to '0', since memory accesses are always of 32-bit.</p> <p>NOTE: The Sensor Node can be stopped by clearing the SNC_EN bit-field of SNC_CTRL_REG and provided that the Power Domains Controller (PDC) is bypassed. The latter can be done by setting to '1' the SNC_SW_CTRL bit-field of SNC_CTRL_REG.</p> | 0x0 |
| 1:0 | R | - | | 0x0 |

Table 340: SNC_R1_REG (0x50020C10)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:0 | R | R1_REG | Returns the current value of the first 32-bit of the last SNC command executed. | 0x0 |

Table 341: SNC_R2_REG (0x50020C14)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:0 | R | R2_REG | Returns the current value of the second 32-bit word of the last SNC command executed. This is useful for the SNC commands composed by two 32-bit words. | 0x0 |

Table 342: SNC_TMP1_REG (0x50020C18)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:0 | R | TMP1_REG | Returns the current value of the Sensor Node's first temporary register. To be used for debugging | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------|-------|
| | | | purposes. | |

Table 343: SNC_TMP2_REG (0x50020C1C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:0 | R | TMP2_REG | Returns the current value of the Sensor Node's second temporary register. To be used for debugging purposes. | 0x0 |

42.13 SPI Controller Registers

Table 344: Register map SPI

| Address | Register | Description |
|------------|--------------------|------------------------------|
| 0x50020300 | SPI_CTRL_REG | SPI control register 0 |
| 0x50020304 | SPI_RX_TX_REG | SPI RX/TX register0 |
| 0x50020308 | SPI_CLEAR_INT_REG | SPI clear interrupt register |
| 0x50020400 | SPI2_CTRL_REG | SPI control register 0 |
| 0x50020404 | SPI2_RX_TX_REG | SPI RX/TX register0 |
| 0x50020408 | SPI2_CLEAR_INT_REG | SPI clear interrupt register |

Table 345: SPI_CTRL_REG (0x50020300)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------------|---|-------|
| 31:26 | - | - | | 0x0 |
| 25 | R/W | SPI_TX_FIFO_NOT_FULL_MASK | When 1, SPI Interrupt is generated when TX fifo is not full | 0x0 |
| 24 | R/W | SPI_DMA_TXREQ_MODE | In case SPI_FIFO_MODE=3 0 = DMA TX request is generated when transaction is finished 1 = DMA TX request is generated when tx buffer is free | 0x0 |
| 23 | R | SPI_TX_FIFO_EMPTY | 0 = Trasmit fifo is not empty 1 = Trasmit fifo is empty | 0x1 |
| 22 | R | SPI_RX_FIFO_FULL | 0 = Receive fifo is not full 1 = Receive fifo is full | 0x0 |
| 21 | R | SPI_RX_FIFO_EMPTY | 0 = Receive fifo is not empty 1 = Receive fifo is empty | 0x1 |
| 20 | R/W | SPI_9BIT_VAL | Determines the value of the first bit in 9 bits SPI mode. | 0x0 |
| 19 | R | SPI_BUSY | 0 = The SPI is not busy with a transfer. This means that either no TX-data is available or that the transfers have been suspended due to a full RX- | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---|-------|
| | | | FIFO. The SPI_CTRL_REG[SPI_INT_BIT] can be used to distinguish between these situations. 1 = The SPI is busy with a transfer. | |
| 18 | R/W | SPI_PRIORITY | 0 = The SPI has low priority, the DMA request signals are reset after the corresponding acknowledge. 1 = The SPI has high priority, DMA request signals remain active until the FIFOs are filled/emptied, so the DMA holds the AHB bus. | 0x0 |
| 17:16 | R/W | SPI_FIFO_MODE | 0 = TX-FIFO and RX-FIFO used (Bidirectional mode). 1 = RX-FIFO used (Read Only Mode) TX-FIFO single depth, no flow control 2 = TX-FIFO used (Write Only Mode), RX-FIFO single depth, no flow control 3 = No FIFOs used (backwards compatible mode) | 0x3 |
| 15 | R/W | SPI_EN_CTRL | 0 = SPI_EN pin disabled in slave mode. Pin SPI_EN is don't care. 1 = SPI_EN pin enabled in slave mode. | 0x0 |
| 14 | R/W | SPI_MINT | 0 = Disable SPI_INT_BIT to ICU 1 = Enable SPI_INT_BIT to ICU. | 0x0 |
| 13 | R | SPI_INT_BIT | 0 = RX Register or FIFO is empty. 1 = SPI interrupt. Data has been transmitted and receivedMust be reset by SW by writing to SPI_CLEAR_INT_REG. | 0x0 |
| 12 | R | SPI_DI | Returns the actual value of pin SPI_DIN (delayed with two internal SPI clock cycles) | 0x0 |
| 11 | R | SPI_TXH | 0 = TX-FIFO is not full, data can be written. 1 = TX-FIFO is full, data can not be written. | 0x0 |
| 10 | R/W | SPI_FORCE_DO | 0 = normal operation 1 = Force SPIDO output level to value of SPI_DO. | 0x0 |
| 9:8 | R/W | SPI_WORD | 00 = 8 bits mode 01 = 16 bit mode 10 = 32 bits mode 11 = 9 bits mode. Only valid in master mode. | 0x0 |
| 7 | R/W | SPI_RST | 0 = normal operation 1 = Reset SPI. Same function as SPI_ON except that internal clock remain active. | 0x0 |
| 6 | R/W | SPI_SMN | Master/slave mode 0 = Master 1 = Slave | 0x0 |
| 5 | R/W | SPI_DO | Pin SPI_DO output level when SPI is idle or when SPI_FORCE_DO=1 | 0x0 |
| 4:3 | R/W | SPI_CLK | Select SPI_CLK clock output frequency in master mode: 00 = SPI_CLK / 8 01 = SPI_CLK / 4 10 = SPI_CLK / 2 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|---|-------|
| | | | 11 = SPI_CLK / 14 | |
| 2 | R/W | SPI_POL | Select SPI_CLK polarity. 0 = SPI_CLK is initially low. 1 = SPI_CLK is initially high. | 0x0 |
| 1 | R/W | SPI_PHA | Select SPI_CLK phase. See functional timing diagrams in SPI chapter | 0x0 |
| 0 | R/W | SPI_ON | 0 = SPI Module switched off (power saving). Everything is reset except SPI_CTRL_REG. When this bit is cleared the SPI will remain active in master mode until the shift register and holding register are both empty. 1 = SPI Module switched on. Should only be set after all control bits have their desired values. So two writes are needed! | 0x0 |

Table 346: SPI_RX_TX_REG (0x50020304)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:0 | R0/W | SPI_DATA | Write: SPI_RX_REG output register 0 (TX-FIFO) Read: SPI_RX_REG input register 0 (RX-FIFO) In 8 or 9 bits mode bits 31 to 8 are not used, they contain old data. In 16 bits mode bits 31 to 16 are not used, they contain old data. | 0x0 |

Table 347: SPI_CLEAR_INT_REG (0x50020308)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 31:0 | R0/W | SPI_CLEAR_INT | Writing any value to this register will clear the SPI_CTRL_REG[SPI_INT_BIT] Reading returns 0. | 0x0 |

Table 348: SPI2_CTRL_REG (0x50020400)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------------|---|-------|
| 31:26 | - | - | | 0x0 |
| 25 | R/W | SPI_TX_FIFO_NOT_FULL_MASK | When 1, SPI Interrupt is generated when TX fifo is not full | 0x0 |
| 24 | R/W | SPI_DMA_TXREQ_MODE | In case SPI_FIFO_MODE=3 0 = DMA TX request is generated when transaction is finished 1 = DMA TX request is generated when tx buffer is free | 0x0 |
| 23 | R | SPI_TX_FIFO_EMPTY | 0 = Trasmit fifo is not empty 1 = Trasmit fifo is empty | 0x1 |
| 22 | R | SPI_RX_FIFO_FULL | 0 = Receive fifo is not full 1 = Receive fifo is full | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 21 | R | SPI_RX_FIFO_EMPTY | 0 = Receive fifo is not empty 1 = Receive fifo is empty | 0x1 |
| 20 | R/W | SPI_9BIT_VAL | Determines the value of the first bit in 9 bits SPI mode. | 0x0 |
| 19 | R | SPI_BUSY | 0 = The SPI is not busy with a transfer. This means that either no TX-data is available or that the transfers have been suspended due to a full RX-FIFO. The SPI_CTRL_REG[SPI_INT_BIT] can be used to distinguish between these situations. 1 = The SPI is busy with a transfer. | 0x0 |
| 18 | R/W | SPI_PRIORITY | 0 = The SPI has low priority, the DMA request signals are reset after the corresponding acknowledge. 1 = The SPI has high priority, DMA request signals remain active until the FIFOs are filled/emptied, so the DMA holds the AHB bus. | 0x0 |
| 17:16 | R/W | SPI_FIFO_MODE | 0 = TX-FIFO and RX-FIFO used (Bidirectional mode). 1 = RX-FIFO used (Read Only Mode) TX-FIFO single depth, no flow control 2 = TX-FIFO used (Write Only Mode), RX-FIFO single depth, no flow control 3 = No FIFOs used (backwards compatible mode) | 0x3 |
| 15 | R/W | SPI_EN_CTRL | 0 = SPI_EN pin disabled in slave mode. Pin SPI_EN is don't care. 1 = SPI_EN pin enabled in slave mode. | 0x0 |
| 14 | R/W | SPI_MINT | 0 = Disable SPI_INT_BIT to ICU 1 = Enable SPI_INT_BIT to ICU. | 0x0 |
| 13 | R | SPI_INT_BIT | 0 = RX Register or FIFO is empty. 1 = SPI interrupt. Data has been transmitted and receivedMust be reset by SW by writing to SPI_CLEAR_INT_REG. | 0x0 |
| 12 | R | SPI_DI | Returns the actual value of pin SPI_DIN (delayed with two internal SPI clock cycles) | 0x0 |
| 11 | R | SPI_TXH | 0 = TX-FIFO is not full, data can be written. 1 = TX-FIFO is full, data can not be written. | 0x0 |
| 10 | R/W | SPI_FORCE_DO | 0 = normal operation 1 = Force SPIDO output level to value of SPI_DO. | 0x0 |
| 9:8 | R/W | SPI_WORD | 00 = 8 bits mode 01 = 16 bit mode 10 = 32 bits mode 11 = 9 bits mode. Only valid in master mode. | 0x0 |
| 7 | R/W | SPI_RST | 0 = normal operation 1 = Reset SPI. Same function as SPI_ON except that internal clock remain active. | 0x0 |
| 6 | R/W | SPI_SMN | Master/slave mode 0 = Master 1 = Slave | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|---|-------|
| 5 | R/W | SPI_DO | Pin SPI_DO output level when SPI is idle or when SPI_FORCE_DO=1 | 0x0 |
| 4:3 | R/W | SPI_CLK | Select SPI_CLK clock output frequency in master mode: 00 = SPI_CLK / 8 01 = SPI_CLK / 4 10 = SPI_CLK / 2 11 = SPI_CLK / 14 | 0x0 |
| 2 | R/W | SPI_POL | Select SPI_CLK polarity. 0 = SPI_CLK is initially low. 1 = SPI_CLK is initially high. | 0x0 |
| 1 | R/W | SPI_PHA | Select SPI_CLK phase. See functional timing diagrams in SPI chapter | 0x0 |
| 0 | R/W | SPI_ON | 0 = SPI Module switched off (power saving). Everything is reset except SPI_CTRL_REG. When this bit is cleared the SPI will remain active in master mode until the shift register and holding register are both empty. 1 = SPI Module switched on. Should only be set after all control bits have their desired values. So two writes are needed! | 0x0 |

Table 349: SPI2_RX_TX_REG (0x50020404)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:0 | R0/W | SPI_DATA | Write: SPI_TX_REG output register 0 (TX-FIFO) Read: SPI_RX_REG input register 0 (RX-FIFO) In 8 or 9 bits mode bits 31 to 8 are not used, they contain old data. In 16 bits mode bits 31 to 16 are not used, they contain old data. | 0x0 |

Table 350: SPI2_CLEAR_INT_REG (0x50020408)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 31:0 | R0/W | SPI_CLEAR_INT | Writing any value to this register will clear the SPI_CTRL_REG[SPI_INT_BIT] Reading returns 0. | 0x0 |

42.14 Timers Registers

Table 351: Register map Timer1

| Address | Register | Description |
|------------|----------------|------------------------|
| 0x50010200 | TIMER_CTRL_REG | Timer control register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|----------------------------|---|
| 0x50010204 | TIMER_TIMER_VAL_REG | Timer counter value |
| 0x50010208 | TIMER_STATUS_REG | Timer status register |
| 0x5001020C | TIMER_GPIO1_CONF_REG | Timer gpio1 selection |
| 0x50010210 | TIMER_GPIO2_CONF_REG | Timer gpio2 selection |
| 0x50010214 | TIMER_RELOAD_REG | Timer reload value and Delay in shot mode |
| 0x50010218 | TIMER_SHOTWIDTH_REG | Timer Shot duration in shot mode |
| 0x5001021C | TIMER_PRESCALER_REG | Timer prescaler value |
| 0x50010220 | TIMER_CAPTURE_GPIO1_REG | Timer value for event on GPIO1 |
| 0x50010224 | TIMER_CAPTURE_GPIO2_REG | Timer value for event on GPIO2 |
| 0x50010228 | TIMER_PRESCALER_VAL_REG | Timer prescaler counter value |
| 0x5001022C | TIMER_PWM_FREQ_REG | Timer pwm frequency register |
| 0x50010230 | TIMER_PWM_DC_REG | Timer pwm dc register |
| 0x50010234 | TIMER_GPIO3_CONF_REG | Timer gpio3 selection |
| 0x50010238 | TIMER_GPIO4_CONF_REG | Timer gpio4 selection |
| 0x5001023C | TIMER_CAPTURE_GPIO3_REG | Timer value for event on GPIO1 |
| 0x50010240 | TIMER_CAPTURE_GPIO4_REG | Timer value for event on GPIO1 |
| 0x50010244 | TIMER_CLEAR_GPIO_EVENT_REG | Timer clear gpio event register |
| 0x50010248 | TIMER_CLEAR_IRQ_REG | Timer clear interrupt |
| 0x50010300 | TIMER2_CTRL_REG | Timer control register |
| 0x50010304 | TIMER2_TIMER_VAL_REG | Timer counter value |
| 0x50010308 | TIMER2_STATUS_REG | Timer status register |
| 0x5001030C | TIMER2_GPIO1_CONF_REG | Timer gpio1 selection |
| 0x50010310 | TIMER2_GPIO2_CONF_REG | Timer gpio2 selection |
| 0x50010314 | TIMER2_RELOAD_REG | Timer reload value and Delay in shot mode |
| 0x50010318 | TIMER2_SHOTWIDTH_REG | Timer Shot duration in shot mode |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------------|---|
| 0x5001031C | TIMER2_PRESCALER_REG | Timer prescaler value |
| 0x50010320 | TIMER2_CAPTURE_GPIO1_REG | Timer value for event on GPIO1 |
| 0x50010324 | TIMER2_CAPTURE_GPIO2_REG | Timer value for event on GPIO2 |
| 0x50010328 | TIMER2_PRESCALER_VAL_REG | Timer prescaler counter valuew |
| 0x5001032C | TIMER2_PWM_FREQ_REG | Timer pwm frequency register |
| 0x50010330 | TIMER2_PWM_DC_REG | Timer pwm dc register |
| 0x50010334 | TIMER2_CLEAR_IRQ_REG | Timer clear interrupt |
| 0x50040A00 | TIMER3_CTRL_REG | Timer control register |
| 0x50040A04 | TIMER3_TIMER_VAL_REG | Timer counter value |
| 0x50040A08 | TIMER3_STATUS_REG | Timer status register |
| 0x50040A0C | TIMER3_GPIO1_CON_F_REG | Timer gpio1 selection |
| 0x50040A10 | TIMER3_GPIO2_CON_F_REG | Timer gpio2 selection |
| 0x50040A14 | TIMER3_RELOAD_REG | Timer reload value and Delay in shot mode |
| 0x50040A1C | TIMER3_PRESCALER_REG | Timer prescaler value |
| 0x50040A20 | TIMER3_CAPTURE_GPIO1_REG | Timer value for event on GPIO1 |
| 0x50040A24 | TIMER3_CAPTURE_GPIO2_REG | Timer value for event on GPIO2 |
| 0x50040A28 | TIMER3_PRESCALER_VAL_REG | Timer prescaler counter valuew |
| 0x50040A2C | TIMER3_PWM_FREQ_REG | Timer pwm frequency register |
| 0x50040A30 | TIMER3_PWM_DC_REG | Timer pwm dc register |
| 0x50040A34 | TIMER3_CLEAR_IRQ_REG | Timer clear interrupt |
| 0x50040B00 | TIMER4_CTRL_REG | Timer control register |
| 0x50040B04 | TIMER4_TIMER_VAL_REG | Timer counter value |
| 0x50040B08 | TIMER4_STATUS_REG | Timer status register |
| 0x50040B0C | TIMER4_GPIO1_CON_F_REG | Timer gpio1 selection |
| 0x50040B10 | TIMER4_GPIO2_CON_F_REG | Timer gpio2 selection |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|---------------------------|---|
| 0x50040B14 | TIMER4_RELOAD_REG | Timer reload value and Delay in shot mode |
| 0x50040B1C | TIMER4_PRESCALER_REG | Timer prescaler value |
| 0x50040B20 | TIMER4_CAPTURE_G_PIO1_REG | Timer value for event on GPIO1 |
| 0x50040B24 | TIMER4_CAPTURE_G_PIO2_REG | Timer value for event on GPIO2 |
| 0x50040B28 | TIMER4_PRESCALER_VAL_REG | Timer prescaler counter value |
| 0x50040B2C | TIMER4_PWM_FREQ_REG | Timer pwm frequency register |
| 0x50040B30 | TIMER4_PWM_DC_REG | Timer pwm dc register |
| 0x50040B34 | TIMER4_CLEAR_IRQ_REG | Timer clear interrupt |

Table 352: TIMER_CTRL_REG (0x50010200)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------|--|-------|
| 31:15 | - | - | | 0x0 |
| 14 | R/W | TIM_CAP_GPIO4_I_RQ_EN | 0 = Event on GPIO4 does not create a CAPTIM interrupt 1 = Event on GPIO4 creates a CAPTIM interrupt | 0x0 |
| 13 | R/W | TIM_CAP_GPIO3_I_RQ_EN | 0 = Event on GPIO3 does not create a CAPTIM interrupt 1 = Event on GPIO3 creates a CAPTIM interrupt | 0x0 |
| 12 | R/W | TIM_CAP_GPIO2_I_RQ_EN | 0 = Event on GPIO2 does not create a CAPTIM interrupt 1 = Event on GPIO2 creates a CAPTIM interrupt | 0x0 |
| 11 | R/W | TIM_CAP_GPIO1_I_RQ_EN | 0 = Event on GPIO1 does not create a CAPTIM interrupt 1 = Event on GPIO1 creates a CAPTIM interrupt | 0x0 |
| 10 | R/W | TIM_IN4_EVENT_F_ALL_EN | Event input 4 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 9 | R/W | TIM_IN3_EVENT_F_ALL_EN | Event input 3 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 8 | R/W | TIM_CLK_EN | Timer clock enable 1 = clock enabled 0 = clock disabled | 0x0 |
| 7 | R/W | TIM_SYS_CLK_EN | Select clock 1 = Timer uses the DIVN clock 0 = Timer uses the lp clock | 0x0 |
| 6 | R/W | TIM_FREE_RUN_MODE_EN | Valid when timer counts up, if it is '1' timer does not zero when reaches to reload value. it becomes zero | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------------|--|-------|
| | | | only when it reaches the max value. | |
| 5 | R/W | TIM_IRQ_EN | Interrupt mask 1 = timer IRQ is unmasked 0 = timer IRQ is masked | 0x0 |
| 4 | R/W | TIM_IN2_EVENT_F_ALL_EN | Event input 2 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 3 | R/W | TIM_IN1_EVENT_F_ALL_EN | Event input 1 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 2 | R/W | TIM_COUNT_DOW_N_EN | Timer count direction 1 = down 0 = up | 0x0 |
| 1 | R/W | TIM_ONESHOT_MODE_EN | Timer mode 1 = One shot enabled 0 = Counter enabled | 0x0 |
| 0 | R/W | TIM_EN | Timer enable 1 = On 0 = Off | 0x0 |

Table 353: TIMER_TIMER_VAL_REG (0x50010204)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|-------------------------------|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_TIMER_VALUE | Gives the current timer value | 0x0 |

Table 354: TIMER_STATUS_REG (0x50010208)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | TIM_GPIO4_EVENT_PENDING | When 1, GPIO4 event is pending. | 0x0 |
| 6 | R | TIM_GPIO3_EVENT_PENDING | When 1, GPIO3 event is pending. | 0x0 |
| 5 | R | TIM_GPIO2_EVENT_PENDING | When 1, GPIO2 event is pending. | 0x0 |
| 4 | R | TIM_GPIO1_EVENT_PENDING | When 1, GPIO1 event is pending. | 0x0 |
| 3:2 | R | TIM_ONESHOT_PHASE | OneShot phase 0 = Wait for event 1 = Delay phase 2 = Start Shot 3 = Shot phase | 0x0 |
| 1 | R | TIM_IN2_STATE | Gives the logic level of the IN1 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|----------------------------------|-------|
| 0 | R | TIM_IN1_STATE | Gives the logic level of the IN2 | 0x0 |

Table 355: **TIMER_GPIO1_CONF_REG** (0x5001020C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO1_CONF | Select one of the 32 GPIOs as IN1, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 356: **TIMER_GPIO2_CONF_REG** (0x50010210)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO2_CONF | Select one of the 32 GPIOs as IN2, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 357: **TIMER_RELOAD_REG** (0x50010214)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R/W | TIM_RELOAD | Reload or max value in timer mode, Delay phase duration in oneshot mode. Actual delay is the register value plus synchronization time (3 clock cycles) | 0x0 |

Table 358: **TIMER_SHOTWIDTH_REG** (0x50010218)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|-------------------------------------|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R/W | TIM_SHOTWIDTH | Shot phase duration in oneshot mode | 0x0 |

Table 359: **TIMER_PRESCALER_REG** (0x5001021C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R/W | TIM_PRESCALER | Defines the timer count frequency. CLOCK frequency / (TIM_PRESCALER+1) | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 360: TIMER_CAPTURE_GPIO1_REG (0x50010220)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO1 | Gives the Capture time for event on GPIO1 | 0x0 |

Table 361: TIMER_CAPTURE_GPIO2_REG (0x50010224)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO2 | Gives the Capture time for event on GPIO2 | 0x0 |

Table 362: TIMER_PRESCALER_VAL_REG (0x50010228)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|---|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R | TIM_PRESCALER_VAL | Gives the current prescaler counter value | 0x0 |

Table 363: TIMER_PWM_FREQ_REG (0x5001022C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|---|-------|
| 31:16 | - | - | | 0x0 |
| 15:0 | R/W | TIM_PWM_FREQ | Defines the PWM frequency. Timer clock frequency / (TIM_PWM_FREQ+1) Timer clock is clock after prescaler | 0x0 |

Table 364: TIMER_PWM_DC_REG (0x50010230)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:16 | - | - | | 0x0 |
| 15:0 | R/W | TIM_PWM_DC | Defines the PWM duty cycle. TIM_PWM_DC / (TIM_PWM_FREQ+1) | 0x0 |

Table 365: TIMER_GPIO3_CONF_REG (0x50010234)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO3_CONF | Select one of the 32 GPIOs as IN3, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 366: TIMER_GPIO4_CONF_REG (0x50010238)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO4_CONF | Select one of the 32 GPIOs as IN4, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 367: TIMER_CAPTURE_GPIO3_REG (0x5001023C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO3 | Gives the Capture time for event on GPIO3 | 0x0 |

Table 368: TIMER_CAPTURE_GPIO4_REG (0x50010240)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO4 | Gives the Capture time for event on GPIO4 | 0x0 |

Table 369: TIMER_CLEAR_GPIO_EVENT_REG (0x50010244)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------|--|-------|
| 3 | R0/W | TIM_CLEAR_GPIO4_EVENT | 1 = Clear GPIO4 event. Return always 0 | 0x0 |
| 2 | R0/W | TIM_CLEAR_GPIO3_EVENT | 1 = Clear GPIO3 event. Return always 0 | 0x0 |
| 1 | R0/W | TIM_CLEAR_GPIO2_EVENT | 1 = Clear GPIO2 event. Return always 0 | 0x0 |
| 0 | R0/W | TIM_CLEAR_GPIO1_EVENT | 1 = Clear GPIO1 event. Return always 0 | 0x0 |

Table 370: TIMER_CLEAR_IRQ_REG (0x50010248)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---------------------------------|-------|
| 0 | R0/W | TIM_CLEAR_IRQ | Write any value clear interrupt | 0x0 |

Table 371: TIMER2_CTRL_REG (0x50010300)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:9 | - | - | | 0x0 |
| 8 | R/W | TIM_CLK_EN | Timer clock enable 1 = clock enabled | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------------|--|-------|
| | | | 0 = clock disabled | |
| 7 | R/W | TIM_SYS_CLK_EN | Select clock 1 = Timer uses the DIVN clock 0 = Timer uses the Ip clock | 0x0 |
| 6 | R/W | TIM_FREE_RUN_MODE_EN | Valid when timer counts up, if it is '1' timer does not zero when reaches to reload value. it becomes zero only when it reaches the max value. | 0x0 |
| 5 | R/W | TIM_IRQ_EN | Interrupt mask 1 = timer IRQ is unmasked 0 = timer IRQ is masked | 0x0 |
| 4 | R/W | TIM_IN2_EVENT_FALL_ALL_EN | Event input 2 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 3 | R/W | TIM_IN1_EVENT_FALL_ALL_EN | Event input 1 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 2 | R/W | TIM_COUNT_DIRECTION_EN | Timer count direction 1 = down 0 = up | 0x0 |
| 1 | R/W | TIM_ONESHOT_MODE_EN | Timer mode 1 = One shot enabled 0 = Counter enabled | 0x0 |
| 0 | R/W | TIM_EN | Timer enable 1 = On 0 = Off | 0x0 |

Table 372: TIMER2_TIMER_VAL_REG (0x50010304)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|-------------------------------|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_TIMER_VALUE | Gives the current timer value | 0x0 |

Table 373: TIMER2_STATUS_REG (0x50010308)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|--|-------|
| 31:4 | - | - | | 0x0 |
| 3:2 | R | TIM_ONESHOT_PHASE | OneShot phase 0 = Wait for event 1 = Delay phase 2 = Start Shot 3 = Shot phase | 0x0 |
| 1 | R | TIM_IN2_STATE | Gives the logic level of the IN1 | 0x0 |
| 0 | R | TIM_IN1_STATE | Gives the logic level of the IN2 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 374: TIMER2_GPIO1_CONF_REG (0x5001030C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO1_CONF | Select one of the 32 GPIOs as IN1, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 375: TIMER2_GPIO2_CONF_REG (0x50010310)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO2_CONF | Select one of the 32 GPIOs as IN2, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 376: TIMER2_RELOAD_REG (0x50010314)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R/W | TIM_RELOAD | Reload or max value in timer mode, Delay phase duration in oneshot mode. Actual delay is the register value plus synchronization time (3 clock cycles) | 0x0 |

Table 377: TIMER2_SHOTWIDTH_REG (0x50010318)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|-------------------------------------|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R/W | TIM_SHOTWIDTH | Shot phase duration in oneshot mode | 0x0 |

Table 378: TIMER2_PRESCALER_REG (0x5001031C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R/W | TIM_PRESCALER | Defines the timer count frequency. CLOCK frequency / (TIM_PRESCALER+1) | 0x0 |

Table 379: TIMER2_CAPTURE_GPIO1_REG (0x50010320)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO1 | Gives the Capture time for event on GPIO1 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 380: TIMER2_CAPTURE_GPIO2_REG (0x50010324)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO2 | Gives the Capture time for event on GPIO2 | 0x0 |

Table 381: TIMER2_PRESCALER_VAL_REG (0x50010328)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|---|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R | TIM_PRESCALER_VAL | Gives the current prescaler counter value | 0x0 |

Table 382: TIMER2_PWM_FREQ_REG (0x5001032C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|---|-------|
| 31:16 | - | - | | 0x0 |
| 15:0 | R/W | TIM_PWM_FREQ | Defines the PWM frequency. Timer clock frequency / (TIM_PWM_FREQ+1) Timer clock is clock after prescaler | 0x0 |

Table 383: TIMER2_PWM_DC_REG (0x50010330)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:16 | - | - | | 0x0 |
| 15:0 | R/W | TIM_PWM_DC | Defines the PWM duty cycle. TIM_PWM_DC / (TIM_PWM_FREQ+1) | 0x0 |

Table 384: TIMER2_CLEAR_IRQ_REG (0x50010334)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---------------------------------|-------|
| 0 | R0/W | TIM_CLEAR_IRQ | Write any value clear interrupt | 0x0 |

Table 385: TIMER3_CTRL_REG (0x50040A00)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:9 | - | - | | 0x0 |
| 8 | R/W | TIM_CLK_EN | Timer clock enable 1 = clock enabled 0 = clock disabled | 0x0 |
| 7 | R/W | TIM_SYS_CLK_EN | Select clock 1 = Timer uses the DIVN clock 0 = Timer uses the Ip clock | 0x0 |
| 6 | R/W | TIM_FREE_RUN_M | Valid when timer counts up, if it is '1' timer does not | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------------|--|-------|
| | | ODE_EN | zero when reaches to reload value. it becomes zero only when it reaches the max value. | |
| 5 | R/W | TIM_IRQ_EN | Interrupt mask 1 = timer IRQ is unmasked 0 = timer IRQ is masked | 0x0 |
| 4 | R/W | TIM_IN2_EVENT_F_ALL_EN | Event input 2 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 3 | R/W | TIM_IN1_EVENT_F_ALL_EN | Event input 1 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 2 | R/W | TIM_COUNT_DOWN_EN | Timer count direction 1 = down 0 = up | 0x0 |
| 1 | R/W | - | | 0x0 |
| 0 | R/W | TIM_EN | Timer enable 1 = On 0 = Off | 0x0 |

Table 386: TIMER3_TIMER_VAL_REG (0x50040A04)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|-------------------------------|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_TIMER_VALUE | Gives the current timer value | 0x0 |

Table 387: TIMER3_STATUS_REG (0x50040A08)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|--|-------|
| 31:4 | - | - | | 0x0 |
| 3:2 | R | TIM_ONESHOT_PHASE | OneShot phase 0 = Wait for event 1 = Delay phase 2 = Start Shot 3 = Shot phase | 0x0 |
| 1 | R | TIM_IN2_STATE | Gives the logic level of the IN1 | 0x0 |
| 0 | R | TIM_IN1_STATE | Gives the logic level of the IN2 | 0x0 |

Table 388: TIMER3_GPIO1_CONF_REG (0x50040A0C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO1_CONF | Select one of the 32 GPIOs as IN1, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 389: TIMER3_GPIO2_CONF_REG (0x50040A10)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO2_CONF | Select one of the 32 GPIOs as IN2, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 390: TIMER3_RELOAD_REG (0x50040A14)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R/W | TIM_RELOAD | Reload or max value in timer mode. Actual delay is the register value plus synchronization time (3 clock cycles) | 0x0 |

Table 391: TIMER3_PRESCALER_REG (0x50040A1C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R/W | TIM_PRESCALER | Defines the timer count frequency. CLOCK frequency / (TIM_PRESCALER+1) | 0x0 |

Table 392: TIMER3_CAPTURE_GPIO1_REG (0x50040A20)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO1 | Gives the Capture time for event on GPIO1 | 0x0 |

Table 393: TIMER3_CAPTURE_GPIO2_REG (0x50040A24)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO2 | Gives the Capture time for event on GPIO2 | 0x0 |

Table 394: TIMER3_PRESCALER_VAL_REG (0x50040A28)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|---|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R | TIM_PRESCALER_VAL | Gives the current prescaler counter value | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 395: TIMER3_PWM_FREQ_REG (0x50040A2C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|---|-------|
| 31:16 | - | - | | 0x0 |
| 15:0 | R/W | TIM_PWM_FREQ | Defines the PWM frequency. Timer clock frequency / (TIM_PWM_FREQ+1) Timer clock is clock after prescaler | 0x0 |

Table 396: TIMER3_PWM_DC_REG (0x50040A30)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:16 | - | - | | 0x0 |
| 15:0 | R/W | TIM_PWM_DC | Defines the PWM duty cycle. TIM_PWM_DC / (TIM_PWM_FREQ+1) | 0x0 |

Table 397: TIMER3_CLEAR_IRQ_REG (0x50040A34)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---------------------------------|-------|
| 0 | R0/W | TIM_CLEAR_IRQ | Write any value clear interrupt | 0x0 |

Table 398: TIMER4_CTRL_REG (0x50040B00)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------------|--|-------|
| 31:9 | - | - | | 0x0 |
| 8 | R/W | TIM_CLK_EN | Timer clock enable 1 = clock enabled 0 = clock disabled | 0x0 |
| 7 | R/W | TIM_SYS_CLK_EN | Select clock 1 = Timer uses the DIVN clock 0 = Timer uses the lp clock | 0x0 |
| 6 | R/W | TIM_FREE_RUN_MODE_EN | Valid when timer counts up, if it is '1' timer does not zero when reaches to reload value. it becomes zero only when it reaches the max value. | 0x0 |
| 5 | R/W | TIM_IRQ_EN | Interrupt mask 1 = timer IRQ is unmasked 0 = timer IRQ is masked | 0x0 |
| 4 | R/W | TIM_IN2_EVENT_FALL_EN | Event input 2 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 3 | R/W | TIM_IN1_EVENT_FALL_EN | Event input 1 edge type 1 = falling edge 0 = rising edge | 0x0 |
| 2 | R/W | TIM_COUNT_DOWN_EN | Timer count direction 1 = down 0 = up | 0x0 |
| 1 | R/W | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-----------------------------------|-------|
| 0 | R/W | TIM_EN | Timer enable 1 = On 0 = Off | 0x0 |

Table 399: TIMER4_TIMER_VAL_REG (0x50040B04)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|-------------------------------|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_TIMER_VALUE | Gives the current timer value | 0x0 |

Table 400: TIMER4_STATUS_REG (0x50040B08)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|--|-------|
| 31:4 | - | - | | 0x0 |
| 3:2 | R | TIM_ONESHOT_PH_ASE | OneShot phase 0 = Wait for event 1 = Delay phase 2 = Start Shot 3 = Shot phase | 0x0 |
| 1 | R | TIM_IN2_STATE | Gives the logic level of the IN1 | 0x0 |
| 0 | R | TIM_IN1_STATE | Gives the logic level of the IN2 | 0x0 |

Table 401: TIMER4_GPIO1_CONF_REG (0x50040B0C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO1_CONF | Select one of the 32 GPIOs as IN1, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 402: TIMER4_GPIO2_CONF_REG (0x50040B10)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5:0 | R/W | TIM_GPIO2_CONF | Select one of the 32 GPIOs as IN2, Valid value 0-32. 1 for the first gpio, 32 for the last gpio. 0 Disable input | 0x0 |

Table 403: TIMER4_RELOAD_REG (0x50040B14)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R/W | TIM_RELOAD | Reload or max value in timer mode. Actual delay is | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | the register value plus synchronization time (3 clock cycles) | |

Table 404: TIMER4_PRESCALER_REG (0x50040B1C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R/W | TIM_PRESCALER | Defines the timer count frequency. CLOCK frequency / (TIM_PRESCALER+1) | 0x0 |

Table 405: TIMER4_CAPTURE_GPIO1_REG (0x50040B20)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO1 | Gives the Capture time for event on GPIO1 | 0x0 |

Table 406: TIMER4_CAPTURE_GPIO2_REG (0x50040B24)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 31:24 | - | - | | 0x0 |
| 23:0 | R | TIM_CAPTURE_GPIO2 | Gives the Capture time for event on GPIO2 | 0x0 |

Table 407: TIMER4_PRESCALER_VAL_REG (0x50040B28)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|---|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R | TIM_PRESCALER_VAL | Gives the current prescaler counter value | 0x0 |

Table 408: TIMER4_PWM_FREQ_REG (0x50040B2C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|---|-------|
| 31:16 | - | - | | 0x0 |
| 15:0 | R/W | TIM_PWM_FREQ | Defines the PWM frequency. Timer clock frequency / (TIM_PWM_FREQ+1) Timer clock is clock after prescaler | 0x0 |

Table 409: TIMER4_PWM_DC_REG (0x50040B30)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|-------------|-------|
| 31:16 | - | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 15:0 | R/W | TIM_PWM_DC | Defines the PWM duty cycle. TIM_PWM_DC / (TIM_PWM_FREQ+1) | 0x0 |

Table 410: TIMER4_CLEAR_IRQ_REG (0x50040B34)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---------------------------------|-------|
| 0 | R0/W | TIM_CLEAR_IRQ | Write any value clear interrupt | 0x0 |

42.15 True Random Number Generator Controller Registers

Table 411: Register map TRNG

| Address | Register | Description |
|------------|------------------|--------------------------|
| 0x50040C00 | TRNG_CTRL_REG | TRNG control register |
| 0x50040C04 | TRNG_FIFOLVL_REG | TRNG FIFO level register |
| 0x50040C08 | TRNG_VER_REG | TRNG Version register |

Table 412: TRNG_CTRL_REG (0x50040C00)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:2 | - | - | | 0x0 |
| 1 | - | - | | 0x0 |
| 0 | R/W | TRNG_ENABLE | 0: Disable the TRNG 1: Enable the TRNG this signal is ignored when the FIFO is full | 0x0 |

Table 413: TRNG_FIFOLVL_REG (0x50040C04)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5 | R | TRNG_FIFOFULL | 1:FIFO full indication. This bit is cleared if the FIFO is read. | 0x0 |
| 4:0 | R | TRNG_FIFOLVL | Number of 32 bit words of random data in the FIFO (max 31) until the FIFO is full. When it is 0 and TRNG_FIFOFULL is 1, it means the FIFO is full. | 0x0 |

Table 414: TRNG_VER_REG (0x50040C08)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------|----------------------|-------|
| 31:24 | R | TRNG_MAJ | Major version number | 0x0 |
| 23:16 | R | TRNG_MIN | Minor version number | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---------------------|-------|
| 15:0 | R | TRNG SVN | SVN revision number | 0x103 |

42.16 UART Registers

Table 415: Register map UART

| Address | Register | Description |
|------------|----------------------|---|
| 0x50020000 | UART_RBR_THR_DLL_REG | Receive Buffer Register |
| 0x50020004 | UART_IER_DLH_REG | Interrupt Enable Register |
| 0x50020008 | UART_IIR_FCR_REG | Interrupt Identification Register/FIFO Control Register |
| 0x5002000C | UART_LCR_REG | Line Control Register |
| 0x50020010 | UART_MCR_REG | Modem Control Register |
| 0x50020014 | UART_LSR_REG | Line Status Register |
| 0x5002001C | UART_SCR_REG | Scratchpad Register |
| 0x50020030 | UART_SRBR_STHR0_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020034 | UART_SRBR_STHR1_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020038 | UART_SRBR_STHR2_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002003C | UART_SRBR_STHR3_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020040 | UART_SRBR_STHR4_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020044 | UART_SRBR_STHR5_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020048 | UART_SRBR_STHR6_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002004C | UART_SRBR_STHR7_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020050 | UART_SRBR_STHR8_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020054 | UART_SRBR_STHR9_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020058 | UART_SRBR_STHR10_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002005C | UART_SRBR_STHR11_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020060 | UART_SRBR_STHR12_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020064 | UART_SRBR_STHR13_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020068 | UART_SRBR_STHR14_REG | Shadow Receive/Transmit Buffer Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|------------------------|---|
| 0x5002006C | UART_SRBR_STHR15_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002007C | UART_USR_REG | UART Status register. |
| 0x50020080 | UART_TFL_REG | Transmit FIFO Level |
| 0x50020084 | UART_RFL_REG | Receive FIFO Level. |
| 0x50020088 | UART_SRR_REG | Software Reset Register. |
| 0x50020090 | UART_SBCR_REG | Shadow Break Control Register |
| 0x50020094 | UART_SDMAM_REG | Shadow DMA Mode |
| 0x50020098 | UART_SFE_REG | Shadow FIFO Enable |
| 0x5002009C | UART_SRT_REG | Shadow RCVR Trigger |
| 0x500200A0 | UART_STET_REG | Shadow TX Empty Trigger |
| 0x500200A4 | UART_HTX_REG | Halt TX |
| 0x500200A8 | UART_DMASA_REG | DMA Software Acknowledge |
| 0x500200C0 | UART_DLF_REG | Divisor Latch Fraction Register |
| 0x500200F8 | UART_UCV_REG | Component Version |
| 0x500200FC | UART_CTR_REG | Component Type Register |
| 0x50020100 | UART2_RBR_THR_DL_L_REG | Receive Buffer Register |
| 0x50020104 | UART2_IER_DLH_REG | Interrupt Enable Register |
| 0x50020108 | UART2_IIR_FCR_REG | Interrupt Identification Register/FIFO Control Register |
| 0x5002010C | UART2_LCR_REG | Line Control Register |
| 0x50020110 | UART2_MCR_REG | Modem Control Register |
| 0x50020114 | UART2_LSR_REG | Line Status Register |
| 0x50020118 | UART2_MSR_REG | Modem Status Register |
| 0x5002011C | UART2_SCR_REG | Scratchpad Register |
| 0x50020130 | UART2_SRBR_STHR0_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020134 | UART2_SRBR_STHR1_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020138 | UART2_SRBR_STHR2_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002013C | UART2_SRBR_STHR3_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020140 | UART2_SRBR_STHR4_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020144 | UART2_SRBR_STHR5_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020148 | UART2_SRBR_STHR6_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002014C | UART2_SRBR_STHR7_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020150 | UART2_SRBR_STHR8_REG | Shadow Receive/Transmit Buffer Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|------------------------|---|
| 0x50020154 | UART2_SRBR_STHR9_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020158 | UART2_SRBR_STHR10_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002015C | UART2_SRBR_STHR11_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020160 | UART2_SRBR_STHR12_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020164 | UART2_SRBR_STHR13_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020168 | UART2_SRBR_STHR14_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002016C | UART2_SRBR_STHR15_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002017C | UART2_USR_REG | UART Status register. |
| 0x50020180 | UART2_TFL_REG | Transmit FIFO Level |
| 0x50020184 | UART2_RFL_REG | Receive FIFO Level. |
| 0x50020188 | UART2_SRR_REG | Software Reset Register. |
| 0x5002018C | UART2_SRTS_REG | Shadow Request to Send |
| 0x50020190 | UART2_SBCR_REG | Shadow Break Control Register |
| 0x50020194 | UART2_SDMAM_REG | Shadow DMA Mode |
| 0x50020198 | UART2_SFE_REG | Shadow FIFO Enable |
| 0x5002019C | UART2_SRT_REG | Shadow RCVR Trigger |
| 0x500201A0 | UART2_STET_REG | Shadow TX Empty Trigger |
| 0x500201A4 | UART2_HTX_REG | Halt TX |
| 0x500201A8 | UART2_DMASA_REG | DMA Software Acknowledge |
| 0x500201C0 | UART2_DLF_REG | Divisor Latch Fraction Register |
| 0x500201C4 | UART2_RAR_REG | Receive Address Register |
| 0x500201C8 | UART2_TAR_REG | Transmit Address Register |
| 0x500201CC | UART2_LCR_EXT | Line Extended Control Register |
| 0x500201F8 | UART2_UCV_REG | Component Version |
| 0x500201FC | UART2_CTR_REG | Component Type Register |
| 0x50020200 | UART3_RBR_THR_DL_L_REG | Receive Buffer Register |
| 0x50020204 | UART3_IER_DLH_REG | Interrupt Enable Register |
| 0x50020208 | UART3_IIR_FCR_REG | Interrupt Identification Register/FIFO Control Register |
| 0x5002020C | UART3_LCR_REG | Line Control Register |
| 0x50020210 | UART3_MCR_REG | Modem Control Register |
| 0x50020214 | UART3_LSR_REG | Line Status Register |
| 0x50020218 | UART3_MSR_REG | Modem Status Register |
| 0x5002021C | UART3_CONFIG_REG | ISO7816 Config Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|-----------------------|---|
| 0x50020230 | UART3_SRBR_STHR0_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020234 | UART3_SRBR_STHR1_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020238 | UART3_SRBR_STHR2_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002023C | UART3_SRBR_STHR3_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020240 | UART3_SRBR_STHR4_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020244 | UART3_SRBR_STHR5_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020248 | UART3_SRBR_STHR6_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002024C | UART3_SRBR_STHR7_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020250 | UART3_SRBR_STHR8_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020254 | UART3_SRBR_STHR9_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020258 | UART3_SRBR_STHR10_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002025C | UART3_SRBR_STHR11_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020260 | UART3_SRBR_STHR12_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020264 | UART3_SRBR_STHR13_REG | Shadow Receive/Transmit Buffer Register |
| 0x50020268 | UART3_SRBR_STHR14_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002026C | UART3_SRBR_STHR15_REG | Shadow Receive/Transmit Buffer Register |
| 0x5002027C | UART3_USR_REG | UART Status register. |
| 0x50020280 | UART3_TFL_REG | Transmit FIFO Level |
| 0x50020284 | UART3_RFL_REG | Receive FIFO Level. |
| 0x50020288 | UART3_SRR_REG | Software Reset Register. |
| 0x5002028C | UART3_SRTS_REG | Shadow Request to Send |
| 0x50020290 | UART3_SBCR_REG | Shadow Break Control Register |
| 0x50020294 | UART3_SDMAM_REG | Shadow DMA Mode |
| 0x50020298 | UART3_SFE_REG | Shadow FIFO Enable |
| 0x5002029C | UART3_SRT_REG | Shadow RCVR Trigger |
| 0x500202A0 | UART3_STET_REG | Shadow TX Empty Trigger |
| 0x500202A4 | UART3_HTX_REG | Halt TX |
| 0x500202A8 | UART3_DMASA_REG | DMA Software Acknowledge |
| 0x500202C0 | UART3_DLF_REG | Divisor Latch Fraction Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|----------------------|---------------------------------------|
| 0x500202C4 | UART3_RAR_REG | Receive Address Register |
| 0x500202C8 | UART3_TAR_REG | Transmit Address Register |
| 0x500202CC | UART3_LCR_EXT | Line Extended Control Register |
| 0x500202E0 | UART3_CTRL_REG | ISO7816 Control Register |
| 0x500202E4 | UART3_TIMER_REG | ISO7816 Timer Register |
| 0x500202E8 | UART3_ERR_CTRL_REG | ISO7816 Error Signal Control Register |
| 0x500202EC | UART3_IRQ_STATUS_REG | ISO7816 Interrupt Status Register |
| 0x500202F8 | UART3_UCV_REG | Component Version |
| 0x500202FC | UART3_CTR_REG | Component Type Register |

Table 416: UART_RBR_THR_DLL_REG (0x50020000)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | RBR_THR_DLL | <p>Receive Buffer Register: (RBR).</p> <p>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.</p> <p>Transmit Holding Register: (THR)</p> <p>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.</p> <p>Divisor Latch (Low): (DLL)</p> <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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:</p> $\text{baud rate} = (\text{serial clock freq}) / (16 * \text{divisor})$ <p>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, once the DLL is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to pass before transmitting or receiving data.</p> <p>Divisor Latch (High): (DLH) (Note: This register is placed in UART_IER_DLH_REG with offset 0x4)</p> <p>Upper 8-bits 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. The output baud rate is equal to the serial clock frequency divided by sixteen times the value of the baud rate divisor, as follows:</p> $\text{baud rate} = (\text{serial clock freq}) / (16 * \text{divisor}).$ <p>Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications occur. Also, once the DLH is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to pass before transmitting or receiving data.</p> | |

Table 417: UART_IER_DLH_REG (0x50020004)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | PTIME_DLH7 | <p>Interrupt Enable Register: PTIME, Programmable THRE Interrupt Mode Enable. This is used to enable/disable the generation of THRE Interrupt. 0 = disabled 1 = enabled</p> <p>Divisor Latch (High): Bit[7] of the 8 bit DLH register.</p> | 0x0 |
| 6:5 | R/W | DLH6_5 | Divisor Latch (High): Bit[6:5] of the 8 bit DLH register | 0x0 |
| 4 | R/W | ELCOLR_DLH4 | <p>Interrupt Enable Register: (read only) ELCOLR, this bit controls the method for clearing the status in the LSR register. This is applicable only for Overrun Error, Parity Error, Framing Error, and Break Interrupt status bits.</p> <p>Always 0 = LSR status bits are cleared either on reading Rx FIFO (RBR Read) or On reading LSR register.</p> <p>Divisor Latch (High): Bit[4] of the 8 bit DLH register</p> | 0x0 |
| 3 | R/W | EDSSI_DLH3 | <p>Interrupt Enable Register: reserved</p> <p>Divisor Latch (High): Bit[3] of the 8 bit DLH register</p> | 0x0 |
| 2 | R/W | ELSI_DLH2 | 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 = | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| | | | enabled Divisor Latch (High): Bit[2] of the 8 bit DLH 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): Bit[1] of the 8 bit DLH 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): Bit[0] of the 8 bit DLH register. | 0x0 |

Table 418: UART_IIR_FCR_REG (0x50020008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | IIR_FCR | <p>On Read Interrupt Identification Register :</p> <p>Bits[7:6], FIFO's Enabled (or FIFOSE): This is used to indicate whether the FIFO's are enabled or disabled.</p> <p>00 = disabled. 11 = enabled.</p> <p>Bits[5:4],Reserved</p> <p>Bits[3:0], Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types:</p> <ul style="list-style-type: none"> 0001 = no interrupt pending. 0010 = THR empty. 0100 = received data available. 0110 = receiver line status. 0111 = busy detect. 1100 = character timeout. <p>On Write FIFO Control Register</p> <p>Bits[7:6], 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</p> <p>Bits[5:4], 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</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>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</p> <p>Bit[3], DMA Mode (or DMAM): This determines the DMA signalling mode used for the <code>dma_tx_req_n</code> and <code>dma_rx_req_n</code> output signals. 0 = mode 0 1 = mode 1</p> <p>Bit[2], 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.</p> <p>Bit[1], 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.</p> <p>Bit[0], 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.</p> | |

Table 419: UART_LCR_REG (0x5002000C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | UART_DLAB | <p>Divisor Latch Access Bit.</p> <p>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.</p> <p>This bit must be cleared after initial baud rate setup in order to access other registers.</p> | 0x0 |
| 6 | R/W | UART_BC | <p>Break Control Bit.</p> <p>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.</p> | 0x0 |
| 5 | - | - | | 0x0 |
| 4 | R/W | UART_EPS | <p>Even Parity Select. Writeable only when UART is not busy (USR[0] is zero).</p> <p>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.</p> | 0x0 |
| 3 | R/W | UART_PEN | <p>Parity Enable. Writeable only when UART is not busy (USR[0] is zero)</p> <p>This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively.</p> <p>0 = parity disabled 1 = parity enabled</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| 2 | R/W | UART_STOP | <p>Number of stop bits.</p> <p>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.</p> <p>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.</p> <p>0 = 1 stop bit 1 = 1.5 stop bits when DLS (LCR[1:0]) is zero, else 2 stop bit</p> | 0x0 |
| 1:0 | R/W | UART_DLS | <p>Data Length Select.</p> <p>This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bit that may be selected areas follows:</p> <p>00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits</p> | 0x0 |

Table 420: UART_MCR_REG (0x50020010)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|---|-------|
| 31:7 | - | - | | 0x0 |
| 6 | R/W | - | | 0x0 |
| 5 | R/W | - | | 0x0 |
| 4 | R/W | UART_LB | <p>LoopBack Bit.</p> <p>This is used to put the UART into a diagnostic mode for test purposes.</p> <p>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.</p> <p>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.</p> | 0x0 |
| 3 | R/W | - | | 0x0 |
| 2 | R/W | - | | 0x0 |
| 1 | R/W | - | | 0x0 |
| 0 | R/W | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 421: UART_LSR_REG (0x50020014)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | UART_RFE | <p>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.</p> <p>0 = no error in RX FIFO 1 = error in RX FIFO</p> <p>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.</p> | 0x0 |
| 6 | R | UART_TEMT | <p>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 and the Transmitter Shift Register are both empty.</p> | 0x1 |
| 5 | R | UART_THRE | <p>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.</p> <p>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.</p> | 0x1 |
| 4 | R | UART_BI | <p>Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data.</p> <p>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.</p> <p>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.</p> <p>Reading the LSR clears the BI bit. In the non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read.</p> | 0x0 |
| 3 | R | UART_FE | <p>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.</p> <p>In the FIFO mode, since 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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|---|-------|
| | | | <p>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 i.e. 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]).</p> <p>0 = no framing error 1 = framing error Reading the LSR clears the FE bit.</p> | |
| 2 | R | UART_PE | <p>Parity Error bit.</p> <p>This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set.</p> <p>In the FIFO mode, since 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.</p> <p>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]).</p> <p>0 = no parity error 1 = parity error Reading the LSR clears the PE bit.</p> | 0x0 |
| 1 | R | UART_OE | <p>Overrun error bit.</p> <p>This is used to indicate the occurrence of an overrun error.</p> <p>This occurs if a new data character was received before the previous data was read.</p> <p>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.</p> <p>0 = no overrun error 1 = overrun error Reading the LSR clears the OE bit.</p> | 0x0 |
| 0 | R | UART_DR | <p>Data Ready bit.</p> <p>This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 422: UART_SCR_REG (0x5002001C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 31:8 | - | - | | 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 423: UART_SRBR_STHR0_REG (0x50020030)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 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 424: UART_SRBR_STHR1_REG (0x50020034)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 425: UART_SRBR_STHR2_REG (0x50020038)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>Shadow Transmit Holding Register</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 426: UART_SRBR_STHR3_REG (0x5002003C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | configuration. Any attempt to write data when the FIFO is full results in the write data being lost. | |

Table 427: UART_SRBR_STHR4_REG (0x50020040)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 428: UART_SRBR_STHR5_REG (0x50020044)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 429: UART_SRBR_STHR6_REG (0x50020048)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 430: UART_SRBR_STHR7_REG (0x5002004C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 431: UART_SRBR_STHR8_REG (0x50020050)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 432: UART_SRBR_STHR9_REG (0x50020054)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 433: UART_SRBR_STHR10_REG (0x50020058)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 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 434: UART_SRBR_STHR11_REG (0x5002005C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 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 435: UART_SRBR_STHR12_REG (0x50020060)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 436: UART_SRBR_STHR13_REG (0x50020064)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>Shadow Transmit Holding Register</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 437: UART_SRBR_STHR14_REG (0x50020068)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | configuration. Any attempt to write data when the FIFO is full results in the write data being lost. | |

Table 438: UART_SRBR_STHR15_REG (0x5002006C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 439: UART_USR_REG (0x5002007C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4 | R | UART_RFF | <p>Receive FIFO Full.</p> <p>This is used to indicate that the receive FIFO is completely full.</p> <p>0 = Receive FIFO not full 1 = Receive FIFO Full</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| | | | This bit is cleared when the RX FIFO is no longer full. | |
| 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 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 uart is idle or inactive. 0 = uart is idle or inactive 1 = 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 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 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 440: UART_TFL_REG (0x50020080)

| 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 441: UART_RFL_REG (0x50020084)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------------|---|-------|
| 4:0 | R | UART_RECEIVE_FIFO_LEVEL | Receive FIFO Level. This indicates the number of data entries in the receive FIFO. | 0x0 |

Table 442: UART_SRR_REG (0x50020088)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:3 | - | - | | 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 (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 443: UART_SBCR_REG (0x50020090)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------------|--|-------|
| 31:1 | - | - | | 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 performing 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 444: UART_SDMAM_REG (0x50020094)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_SHADOW_DMA_MODE | <p>Shadow DMA Mode.</p> <p>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.</p> <p>0 = mode 0 1 = mode 1</p> | 0x0 |

Table 445: UART_SFE_REG (0x50020098)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_SHADOW_FIFO_ENABLE | <p>Shadow FIFO Enable.</p> <p>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.</p> | 0x0 |

Table 446: UART_SRT_REG (0x5002009C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------------|--|-------|
| 31:2 | - | - | | 0x0 |
| 1:0 | R/W | UART_SHADOW_RCVR_TRIGGER | <p>Shadow RCVR Trigger.</p> <p>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.</p> <p>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:</p> <ul style="list-style-type: none"> 00 = 1 character in the FIFO 01 = FIFO $\frac{1}{4}$ full 10 = FIFO $\frac{1}{2}$ full 11 = FIFO 2 less than full | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 447: UART_STET_REG (0x500200A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------------------|--|-------|
| 31:2 | - | - | | 0x0 |
| 1:0 | R/W | UART_SHADOW_T X_EMPTY_TRIGGER | <p>Shadow TX Empty Trigger.</p> <p>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.</p> <p>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:</p> <ul style="list-style-type: none"> 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO ¼ full 11 = FIFO ½ full | 0x0 |

Table 448: UART_HTX_REG (0x500200A4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_HALT_TX | <p>This register is used to halt transmissions, so that the transmit FIFO can be filled by the master when FIFOs are implemented and enabled.</p> <p>0 = Halt TX disabled</p> <p>1 = Halt TX enabled</p> <p>Note, if FIFOs are not enabled, the setting of the halt TX register has no effect on operation.</p> | 0x0 |

Table 449: UART_DMASA_REG (0x500200A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | W | UART_DMASA | <p>This register is used 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_app_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.</p> | 0x0 |

Table 450: UART_DLF_REG (0x500200C0)

| 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 451: UART_UCV_REG (0x500200F8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|-------------------|----------------|
| 31:0 | R | UART_UCV | Component Version | 0x34303 12A |

Table 452: UART_CTR_REG (0x500200FC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|-------------------------|----------------|
| 31:0 | R | UART_CTR | Component Type Register | 0x44570 110 |

Table 453: UART2_RBR_THR_DLL_REG (0x50020100)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|---|-------|
| 8 | R/W | RBR_THR_9BIT | When 9BIT_DATA_EN, On read :Receive Buffer bit 8 - On write Transmit Buffer bit 8 when LCR_EXT[3]=1 | 0x0 |
| 7:0 | R/W | RBR_THR_DLL | <p>Receive Buffer Register: (RBR).</p> <p>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.</p> <p>Transmit Holding Register: (THR)</p> <p>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.</p> <p>Divisor Latch (Low): (DLL)</p> <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>(sclk) frequency divided by sixteen times the value of the baud rate divisor, as follows:</p> $\text{baud rate} = (\text{serial clock freq}) / (16 * \text{divisor})$ <p>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, once the DLL is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to pass before transmitting or receiving data.</p> <p>Divisor Latch (High): (DLH) (Note: This register is placed in UART_IER_DLH_REG with offset 0x4)</p> <p>Upper 8-bits 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. The output baud rate is equal to the serial clock frequency divided by sixteen times the value of the baud rate divisor, as follows:</p> $\text{baud rate} = (\text{serial clock freq}) / (16 * \text{divisor}).$ <p>Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications occur. Also, once the DLH is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to pass before transmitting or receiving data.</p> | |

Table 454: UART2_IER_DLH_REG (0x50020104)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | PTIME_DLH7 | <p>Interrupt Enable Register: PTIME, Programmable THRE Interrupt Mode Enable. This is used to enable/disable the generation of THRE Interrupt. 0 = disabled 1 = enabled</p> <p>Divisor Latch (High): Bit[7] of the 8 bit DLH register.</p> | 0x0 |
| 6:5 | R/W | DLH6_5 | Divisor Latch (High): Bit[6:5] of the 8 bit DLH register | 0x0 |
| 4 | R/W | ELCOLR_DLH4 | <p>Interrupt Enable Register: ELCOLR (read only), this bit controls the method for clearing the status in the LSR register. This is applicable only for Overrun Error, Parity Error, Framing Error, and Break Interrupt status bits.</p> <p>0 = LSR status bits are cleared either on reading Rx FIFO (RBR Read) or On reading LSR register.</p> <p>Divisor Latch (High): Bit[4] of the 8 bit DLH register</p> | 0x0 |
| 3 | R/W | EDSSI_DLH3 | <p>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</p> <p>Divisor Latch (High): Bit[3] of the 8 bit DLH register</p> | 0x0 |
| 2 | R/W | ELSI_DLH2 | 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 = | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| | | | enabled Divisor Latch (High): Bit[2] of the 8 bit DLH 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): Bit[1] of the 8 bit DLH 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): Bit[0] of the 8 bit DLH register. | 0x0 |

Table 455: UART2_IIR_FCR_REG (0x50020108)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | IIR_FCR | <p>On Read Interrupt Identification Register : Bits[7:6], FIFO's Enabled (or FIFOSE): This is used to indicate whether the FIFO's are enabled or disabled. 00 = disabled. 11 = enabled.</p> <p>Bits[5:4],Reserved</p> <p>Bits[3:0], 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.</p> <p>On Write FIFO Control Register</p> <p>Bits[7:6], 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</p> <p>Bits[5:4], 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</p> <p>Bit[3], 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</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>Bit[2], 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.</p> <p>Bit[1], 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.</p> <p>Bit[0], 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.</p> | |

Table 456: UART2_LCR_REG (0x5002010C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | UART_DLAB | <p>Divisor Latch Access Bit.</p> <p>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.</p> <p>This bit must be cleared after initial baud rate setup in order to access other registers.</p> | 0x0 |
| 6 | R/W | UART_BC | <p>Break Control Bit.</p> <p>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.</p> | 0x0 |
| 5 | R/W | UART_SP | <p>Stick Parity. (writeable only when UART is not busy USR[0] is 0); otherwise always writable and always readable. This bit is used to force parity value.</p> <p>When PEN, EPS and Stick Parity are set to 1, the parity bit is transmitted and checked as logic 0. If PEN and Stick Parity are set to 1 and EPS is a logic 0, then parity bit is transmitted and checked as a logic 1. If this bit is set to 0, Stick Parity is disabled.</p> | 0x0 |
| 4 | R/W | UART_EPS | <p>Even Parity Select. Writeable only when UART is not busy (USR[0] is zero).</p> <p>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.</p> | 0x0 |
| 3 | R/W | UART_PEN | <p>Parity Enable. Writeable only when UART is not busy (USR[0] is zero)</p> <p>This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively.</p> <p>0 = parity disabled 1 = parity enabled</p> | 0x0 |
| 2 | R/W | UART_STOP | Number of stop bits. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| | | | <p>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.</p> <p>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.</p> <p>0 = 1 stop bit 1 = 1.5 stop bits when DLS (LCR[1:0]) is zero, else 2 stop bit</p> | |
| 1:0 | R/W | UART_DLS | <p>Data Length Select.</p> <p>This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bit that may be selected areas follows:</p> <p>00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits</p> | 0x0 |

Table 457: UART2_MCR_REG (0x50020110)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6 | R/W | - | | 0x0 |
| 5 | R/W | UART_AFCE | <p>Auto Flow Control Enable.</p> <p>When FIFOs are enabled and the Auto Flow Control Enable (AFCE) bit is set, Auto Flow Control features are enabled as described in "Auto Flow Control".</p> <p>0 = Auto Flow Control Mode disabled 1 = Auto Flow Control Mode enabled</p> | 0x0 |
| 4 | R/W | UART_LB | <p>LoopBack Bit.</p> <p>This is used to put the UART into a diagnostic mode for test purposes.</p> <p>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) are looped back to the inputs, internally.</p> | 0x0 |
| 3 | R/W | - | | 0x0 |
| 2 | R/W | - | | 0x0 |
| 1 | R/W | UART_RTS | <p>Request to Send.</p> <p>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.</p> <p>When Auto RTS Flow Control is not enabled</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | (MCR[5] set to zero), the rts_n signal is set low by programming MCR[1] (RTS) to a high. In Auto Flow Control, 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. | |
| 0 | R/W | - | | 0x0 |

Table 458: UART2_LSR_REG (0x50020114)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 8 | R | UART_ADDR_RCV_D | <p>Address Received Bit.</p> <p>If 9Bit data mode (LCR_EXT[0]=1) is enabled, this bit is used to indicate the 9th bit of the receive data is set to 1. This bit can also be used to indicate whether the incoming character is address or data.</p> <p>1 = Indicates the character is address.</p> <p>0 = Indicates the character is data.</p> <p>In the FIFO mode, since the 9th bit is associated with a character received, it is revealed when the character with the 9th bit set to 1 is at the top of the FIFO.</p> <p>Reading the LSR clears the 9BIT.</p> <p>Note: User needs to ensure that interrupt gets cleared (reading LSR register) before the next address byte arrives. If there is a delay in clearing the interrupt, then Software will not be able to distinguish between multiple address related interrupt.</p> | 0x0 |
| 7 | R | UART_RFE | <p>Receiver FIFO Error bit.</p> <p>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.</p> <p>0 = no error in RX FIFO</p> <p>1 = error in RX FIFO</p> <p>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.</p> | 0x0 |
| 6 | R | UART_TEMT | <p>Transmitter Empty bit.</p> <p>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 and the Transmitter Shift Register are both empty.</p> | 0x1 |
| 5 | R | UART_THRE | <p>Transmit Holding Register Empty bit.</p> <p>If THRE mode is disabled (IER[7] set to zero) and regardless of FIFO's being implemented/enabled or</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|---|-------|
| | | | <p>not, this bit indicates that the THR or TX FIFO is empty.</p> <p>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.</p> | |
| 4 | R | UART_BI | <p>Break Interrupt bit.</p> <p>This is used to indicate the detection of a break sequence on the serial input data.</p> <p>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.</p> <p>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.</p> <p>Reading the LSR clears the BI bit. In the non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read.</p> | 0x0 |
| 3 | R | UART_FE | <p>Framing Error bit.</p> <p>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.</p> <p>In the FIFO mode, since 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.</p> <p>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 i.e. 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]).</p> <p>0 = no framing error 1 = framing error</p> <p>Reading the LSR clears the FE bit.</p> | 0x0 |
| 2 | R | UART_PE | <p>Parity Error bit.</p> <p>This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set.</p> <p>In the FIFO mode, since 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.</p> <p>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]).</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| 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 459: UART2_MSR_REG (0x50020118)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:5 | - | - | | 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 | - | - | | 0x0 |
| 0 | R | UART_DCTS | Delta Clear to Send. This is used to indicate that the modem control line cts_n has changed since the last time the MSR was read. 0 = no change on cts_n since last read of MSR 1 = change on cts_n since last read of MSR Reading the MSR clears the DCTS bit. In Loopback | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>Mode (MCR[4] = 1), DCTS reflects changes on MCR[1] (RTS).</p> <p>Note, if the DCTS bit is not set and the cts_n signal is asserted (low) and a reset occurs (software or otherwise), then the DCTS bit is set when the reset is removed if the cts_n signal remains asserted.</p> | |

Table 460: UART2_SCR_REG (0x5002011C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 31:8 | - | - | | 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 461: UART2_SRBR_STHR0_REG (0x50020130)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | FIFO is full results in the write data being lost. | |

Table 462: UART2_SRBR_STHR1_REG (0x50020134)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 463: UART2_SRBR_STHR2_REG (0x50020138)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 464: UART2_SRBR_STHR3_REG (0x5002013C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 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 465: UART2_SRBR_STHR4_REG (0x50020140)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 466: UART2_SRBR_STHR5_REG (0x50020144)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 467: UART2_SRBR_STHR6_REG (0x50020148)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 468: UART2_SRBR_STHR7_REG (0x5002014C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 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 469: UART2_SRBR_STHR8_REG (0x50020150)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 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 470: UART2_SRBR_STHR9_REG (0x50020154)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 471: UART2_SRBR_STHR10_REG (0x50020158)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>Shadow Transmit Holding Register</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 472: UART2_SRBR_STHR11_REG (0x5002015C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | configuration. Any attempt to write data when the FIFO is full results in the write data being lost. | |

Table 473: UART2_SRBR_STHR12_REG (0x50020160)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 474: UART2_SRBR_STHR13_REG (0x50020164)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 475: UART2_SRBR_STHR14_REG (0x50020168)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 476: UART2_SRBR_STHR15_REG (0x5002016C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 477: UART2_USR_REG (0x5002017C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4 | R | UART_RFF | <p>Receive FIFO Full. This is used to indicate that the receive FIFO is completely full.</p> <p>0 = Receive FIFO not full 1 = Receive FIFO Full</p> <p>This bit is cleared when the RX FIFO is no longer full.</p> | 0x0 |
| 3 | R | UART_RFNE | <p>Receive FIFO Not Empty. This is used to indicate that the receive FIFO contains one or more entries.</p> <p>0 = Receive FIFO is empty 1 = Receive FIFO is not empty</p> <p>This bit is cleared when the RX FIFO is empty.</p> | 0x0 |
| 2 | R | UART_TFE | <p>Transmit FIFO Empty. This is used to indicate that the transmit FIFO is completely empty.</p> <p>0 = Transmit FIFO is not empty 1 = Transmit FIFO is empty</p> <p>This bit is cleared when the TX FIFO is no longer empty.</p> | 0x1 |
| 1 | R | UART_TFNF | <p>Transmit FIFO Not Full. This is used to indicate that the transmit FIFO is not full.</p> <p>0 = Transmit FIFO is full 1 = Transmit FIFO is not full</p> <p>This bit is cleared when the TX FIFO is full.</p> | 0x1 |
| 0 | R | UART_BUSY | <p>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.</p> | 0x0 |

Table 478: UART2_TFL_REG (0x50020180)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------------|---|-------|
| 4:0 | R | UART_TRANSMIT_FIFO_LEVEL | <p>Transmit FIFO Level. This indicates the number of data entries in the transmit FIFO.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 479: UART2_RFL_REG (0x50020184)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------------|---|-------|
| 4:0 | R | UART_RECEIVE_FIFO_LEVEL | Receive FIFO Level. This indicates the number of data entries in the receive FIFO. | 0x0 |

Table 480: UART2_SRR_REG (0x50020188)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:3 | - | - | | 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 (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 481: UART2_SRTS_REG (0x5002018C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------------------|--|-------|
| 31:1 | - | - | | 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 perform 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, (active MCR[5] = 1) and | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 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. | |

Table 482: **UART2_SBCR_REG** (0x50020190)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------------|--|-------|
| 31:1 | - | - | | 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 performing 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 |

Table 483: **UART2_SDMAM_REG** (0x50020194)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|--|-------|
| 31:1 | - | - | | 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 484: **UART2_SFE_REG** (0x50020198)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------------|--|-------|
| 31:1 | - | - | | 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | controller portion of FIFOs are reset. | |

Table 485: UART2_SRT_REG (0x5002019C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------------------|---|-------|
| 31:2 | - | - | | 0x0 |
| 1:0 | R/W | UART_SHADOW_R CVR_TRIGGER | <p>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.</p> <p>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:</p> <ul style="list-style-type: none"> 00 = 1 character in the FIFO 01 = FIFO ¼ full 10 = FIFO ½ full 11 = FIFO 2 less than full | 0x0 |

Table 486: UART2_STET_REG (0x500201A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------------------|---|-------|
| 31:2 | - | - | | 0x0 |
| 1:0 | R/W | UART_SHADOW_T X_EMPTY_TRIGGER | <p>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.</p> <p>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:</p> <ul style="list-style-type: none"> 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO ¼ full 11 = FIFO ½ full | 0x0 |

Table 487: UART2_HTX_REG (0x500201A4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_HALT_TX | This register is used to halt transmissions, so that the transmit FIFO can be filled by the master when | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | FIFOs are implemented and enabled. 0 = Halt TX disabled 1 = Halt TX enabled Note, if FIFOs are not enabled, the setting of the halt TX register has no effect on operation. | |

Table 488: UART2_DMASA_REG (0x500201A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | W | UART_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 489: UART2_DLF_REG (0x500201C0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:4 | - | - | | 0x0 |
| 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 490: UART2_RAR_REG (0x500201C4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| 7:0 | R/W | UART_RAR | This is an address matching register during receive mode. If the 9-th bit is set in the incoming character then the remaining 8-bits will be checked against this register value. If the match happens then subsequent characters with 9-th bit set to 0 will be treated as data byte until the next address byte is received. Note: - This register is applicable only when 'ADDR_MATCH'(LCR_EXT[1] and 'DLS_E' (LCR_EXT[0]) bits are set to 1. RAR should be programmed only when UART is not busy. | 0x0 |

Table 491: UART2_TAR_REG (0x500201C8)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| 7:0 | R/W | UART_TAR | This is an address matching register during transmit mode. If DLS_E (LCR_EXT[0]) bit is enabled, then uart will send the 9-bit character with 9-th bit set to | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>1 and remaining 8-bit address will be sent from this register provided 'SEND_ADDR' (LCR_EXT[2]) bit is set to 1.</p> <p>Note:</p> <ul style="list-style-type: none"> - This register is used only to send the address. The normal data should be sent by programming THR register. - Once the address is started to send on the DW_apb_uart serial lane, then 'SEND_ADDR' bit will be auto-cleared by the hardware. | |

Table 492: UART2_LCR_EXT (0x500201CC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|--|-------|
| 3 | R/W | UART_TRANSMIT_MODE | <p>Transmit mode control bit. This bit is used to control the type of transmit mode during 9-bit data transfers.</p> <p>1 = In this mode of operation, Transmit Holding Register (THR) and Shadow Transmit Holding Register (STHR) are 9-bit wide. The user needs to ensure that the THR/STHR register is written correctly for address/data.</p> <p>Address: 9th bit is set to 1, Data : 9th bit is set to 0.</p> <p>Note: Transmit address register (TAR) is not applicable in this mode of operation.</p> <p>0 = In this mode of operation, Transmit Holding Register (THR) and Shadow Transmit Holding register (STHR) are 8-bit wide. The user needs to program the address into Transmit Address Register (TAR) and data into the THR/STHR register. SEND_ADDR bit is used as a control knob to indicate the uart on when to send the address.</p> | 0x0 |
| 2 | R/W | UART_SEND_ADDR | <p>Send address control bit. This bit is used as a control knob for the user to determine when to send the address during transmit mode.</p> <p>1 = 9-bit character will be transmitted with 9-th bit set to 1 and the remaining 8-bits will match to what is being programmed in 'Transmit Address Register'.</p> <p>0 = 9-bit character will be transmitted with 9-th bit set to 0 and the remaining 8-bits will be taken from the TXFIFO which is programmed through 8-bit wide THR/STHR register.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This bit is auto-cleared by the hardware, after sending out the address character. User is not expected to program this bit to 0. 2. This field is applicable only when DLS_E bit is set to 1 and TRANSMIT_MODE is set to 0. | 0x0 |
| 1 | R/W | UART_ADDR_MATCH | <p>Address Match Mode. This bit is used to enable the address match feature during receive.</p> <p>1 = Address match mode; uart will wait until the incoming character with 9-th bit set to 1. And further checks to see if the address matches with what is</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| | | | <p>programmed in 'Receive Address Match Register'. If match is found, then subsequent characters will be treated as valid data and DW_apb_uart starts receiving data.</p> <p>0 = Normal mode; DW_apb_uart will start to receive the data and 9-bit character will be formed and written into the receive RXFIFO. User is responsible to read the data and differentiate b/n address and data.</p> <p>Note: This field is applicable only when DLS_E is set to 1.</p> | |
| 0 | R/W | UART_DLS_E | Extension for DLS. This bit is used to enable 9-bit data for transmit and receive transfers. | 0x0 |

Table 493: UART2_UCV_REG (0x500201F8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|-------------------|----------------|
| 31:0 | R | UART_UCV | Component Version | 0x34303 12A |

Table 494: UART2_CTR_REG (0x500201FC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|-------------------------|----------------|
| 31:0 | R | UART_CTR | Component Type Register | 0x44570 110 |

Table 495: UART3_RBR_THR_DLL_REG (0x50020200)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|---|-------|
| 8 | R/W | RBR_THR_9BIT | When 9BIT_DATA_EN, On read :Receive Buffer bit 8 - On write Transmit Buffer bit 8 when LCR_EXT[3]=1 | 0x0 |
| 7:0 | R/W | RBR_THR_DLL | <p>Receive Buffer Register: (RBR).</p> <p>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.</p> <p>Transmit Holding Register: (THR)</p> <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>Divisor Latch (Low): (DLL)</p> <p>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:</p> $\text{baud rate} = (\text{serial clock freq}) / (16 * \text{divisor})$ <p>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, once the DLL is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to pass before transmitting or receiving data.</p> <p>Divisor Latch (High): (DLH) (Note: This register is placed in UART_IER_DLH_REG with offset 0x4)</p> <p>Upper 8-bits 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. The output baud rate is equal to the serial clock frequency divided by sixteen times the value of the baud rate divisor, as follows:</p> $\text{baud rate} = (\text{serial clock freq}) / (16 * \text{divisor}).$ <p>Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications occur. Also, once the DLH is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to pass before transmitting or receiving data.</p> | |

Table 496: UART3_IER_DLH_REG (0x50020204)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | PTIME_DLH7 | <p>Interrupt Enable Register: PTIME, Programmable THRE Interrupt Mode Enable. This is used to enable/disable the generation of THRE Interrupt. 0 = disabled 1 = enabled</p> <p>Divisor Latch (High): Bit[7] of the 8 bit DLH register.</p> | 0x0 |
| 6:5 | R/W | DLH6_5 | Divisor Latch (High): Bit[6:5] of the 8 bit DLH register | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| 4 | R/W | ELCOLR_DLH4 | <p>Interrupt Enable Register: ELCOLR (read only), this bit controls the method for clearing the status in the LSR register. This is applicable only for Overrun Error, Parity Error, Framing Error, and Break Interrupt status bits.</p> <p>0 = LSR status bits are cleared either on reading Rx FIFO (RBR Read) or On reading LSR register.</p> <p>Divisor Latch (High): Bit[4] of the 8 bit DLH register</p> | 0x0 |
| 3 | R/W | EDSSI_DLH3 | <p>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</p> <p>Divisor Latch (High): Bit[3] of the 8 bit DLH register</p> | 0x0 |
| 2 | R/W | ELSI_DLH2 | <p>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</p> <p>Divisor Latch (High): Bit[2] of the 8 bit DLH register.</p> | 0x0 |
| 1 | R/W | ETBEI_DLH1 | <p>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</p> <p>Divisor Latch (High): Bit[1] of the 8 bit DLH register.</p> | 0x0 |
| 0 | R/W | ERBFI_DLH0 | <p>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</p> <p>Divisor Latch (High): Bit[0] of the 8 bit DLH register.</p> | 0x0 |

Table 497: UART3_IIR_FCR_REG (0x50020208)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | IIR_FCR | <p>On Read Interrupt Identification Register :</p> <p>Bits[7:6], FIFO's Enabled (or FIFOSE): This is used to indicate whether the FIFO's are enabled or disabled. 00 = disabled. 11 = enabled.</p> <p>Bits[5:4],Reserved</p> <p>Bits[3:0], 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.</p> <p>On Write FIFO Control Register</p> <p>Bits[7:6], 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</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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</p> <p>Bits[5:4], 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</p> <p>Bit[3], 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</p> <p>Bit[2], 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.</p> <p>Bit[1], 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.</p> <p>Bit[0], 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.</p> | |

Table 498: UART3_LCR_REG (0x5002020C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | UART_DLAB | <p>Divisor Latch Access Bit. 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 initial baud rate setup in order to access other registers.</p> | 0x0 |
| 6 | R/W | UART_BC | <p>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.</p> | 0x0 |
| 5 | R/W | UART_SP | <p>Stick Parity. (writeable only when UART is not busy USR[0] is 0); otherwise always writable and always readable. This bit is used to force parity value. When PEN, EPS and Stick Parity are set to 1, the parity bit is transmitted and checked as logic 0. If</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| | | | PEN and Stick Parity are set to 1 and EPS is a logic 0, then parity bit is transmitted and checked as a logic 1. If this bit is set to 0, Stick Parity is disabled. | |
| 4 | R/W | UART_EPS | <p>Even Parity Select. Writeable only when UART is not busy (USR[0] is zero).</p> <p>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.</p> | 0x0 |
| 3 | R/W | UART_PEN | <p>Parity Enable. Writeable only when UART is not busy (USR[0] is zero)</p> <p>This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively.</p> <p>0 = parity disabled 1 = parity enabled</p> | 0x0 |
| 2 | R/W | UART_STOP | <p>Number of stop bits.</p> <p>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.</p> <p>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.</p> <p>0 = 1 stop bit 1 = 1.5 stop bits when DLS (LCR[1:0]) is zero, else 2 stop bit</p> | 0x0 |
| 1:0 | R/W | UART_DLS | <p>Data Length Select.</p> <p>This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bit that may be selected areas follows:</p> <p>00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits</p> | 0x0 |

Table 499: UART3_MCR_REG (0x50020210)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6 | R/W | - | | 0x0 |
| 5 | R/W | UART_AFCE | <p>Auto Flow Control Enable.</p> <p>When FIFOs are enabled and the Auto Flow Control Enable (AFCE) bit is set, Auto Flow Control features are enabled as described in "Auto Flow Control".</p> <p>0 = Auto Flow Control Mode disabled 1 = Auto Flow Control Mode enabled</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| 4 | R/W | UART_LB | <p>LoopBack Bit.</p> <p>This is used to put the UART into a diagnostic mode for test purposes.</p> <p>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) are looped back to the inputs, internally.</p> | 0x0 |
| 3 | R/W | - | | 0x0 |
| 2 | R/W | - | | 0x0 |
| 1 | R/W | UART_RTS | <p>Request to Send.</p> <p>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.</p> <p>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, 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.</p> <p>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.</p> | 0x0 |
| 0 | R/W | - | | 0x0 |

Table 500: UART3_LSR_REG (0x50020214)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 8 | R | UART_ADDR_RCV_D | <p>Address Received Bit.</p> <p>If 9Bit data mode (LCR_EXT[0]=1) is enabled, this bit is used to indicate the 9th bit of the receive data is set to 1. This bit can also be used to indicate whether the incoming character is address or data.</p> <p>1 = Indicates the character is address.</p> <p>0 = Indicates the character is data.</p> <p>In the FIFO mode, since the 9th bit is associated with a character received, it is revealed when the character with the 9th bit set to 1 is at the top of the FIFO.</p> <p>Reading the LSR clears the 9BIT.</p> <p>Note: User needs to ensure that interrupt gets cleared (reading LSR register) before the next address byte arrives. If there is a delay in clearing the interrupt, then Software will not be able to distinguish between multiple address related interrupt.</p> | 0x0 |
| 7 | R | UART_RFE | <p>Receiver FIFO Error bit.</p> <p>This bit is only relevant when FIFOs are enabled</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | <p>(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.</p> <p>0 = no error in RX FIFO 1 = error in RX FIFO</p> <p>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.</p> | |
| 6 | R | UART_TEMT | <p>Transmitter Empty bit.</p> <p>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 and the Transmitter Shift Register are both empty.</p> | 0x1 |
| 5 | R | UART_THRE | <p>Transmit Holding Register Empty bit.</p> <p>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.</p> <p>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.</p> | 0x1 |
| 4 | R | UART_BI | <p>Break Interrupt bit.</p> <p>This is used to indicate the detection of a break sequence on the serial input data.</p> <p>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.</p> <p>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.</p> <p>Reading the LSR clears the BI bit. In the non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read.</p> | 0x0 |
| 3 | R | UART_FE | <p>Framing Error bit.</p> <p>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.</p> <p>In the FIFO mode, since 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.</p> <p>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 i.e. data,</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|--|-------|
| | | | <p>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]).</p> <p>0 = no framing error 1 = framing error</p> <p>Reading the LSR clears the FE bit.</p> | |
| 2 | R | UART_PE | <p>Parity Error bit.</p> <p>This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set.</p> <p>In the FIFO mode, since 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.</p> <p>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]).</p> <p>0 = no parity error 1 = parity error</p> <p>Reading the LSR clears the PE bit.</p> | 0x0 |
| 1 | R | UART_OE | <p>Overrun error bit.</p> <p>This is used to indicate the occurrence of an overrun error.</p> <p>This occurs if a new data character was received before the previous data was read.</p> <p>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.</p> <p>0 = no overrun error 1 = overrun error</p> <p>Reading the LSR clears the OE bit.</p> | 0x0 |
| 0 | R | UART_DR | <p>Data Ready bit.</p> <p>This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO.</p> <p>0 = no data ready 1 = data ready</p> <p>This bit is cleared when the RBR is read in non-FIFO mode, or when the receiver FIFO is empty, in FIFO mode.</p> | 0x0 |

Table 501: UART3_MSR_REG (0x50020218)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|----------------|-------|
| 31:5 | - | - | | 0x0 |
| 4 | R | UART_CTS | Clear to Send. | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | <p>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.</p> <p>0 = cts_n input is de-asserted (logic 1) 1 = cts_n input is asserted (logic 0)</p> <p>In Loopback Mode (MCR[4] = 1), CTS is the same as MCR[1] (RTS).</p> | |
| 3:1 | - | - | | 0x0 |
| 0 | R | UART_DCTS | <p>Delta Clear to Send.</p> <p>This is used to indicate that the modem control line cts_n has changed since the last time the MSR was read.</p> <p>0 = no change on cts_n since last read of MSR 1 = change on cts_n since last read of MSR</p> <p>Reading the MSR clears the DCTS bit. In Loopback Mode (MCR[4] = 1), DCTS reflects changes on MCR[1] (RTS).</p> <p>Note, if the DCTS bit is not set and the cts_n signal is asserted (low) and a reset occurs (software or otherwise), then the DCTS bit is set when the reset is removed if the cts_n signal remains asserted.</p> | 0x0 |

Table 502: UART3_CONFIG_REG (0x5002021C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:3 | R/W | ISO7816_SCRATCH_PAD | This register is for programmers to use as a temporary storage space. It has no defined purpose in the UART Ctrl. | 0x0 |
| 2 | R/W | ISO7816_ENABLE | 0 : Normal Uart 1 : ISO7816 Enabled | 0x0 |
| 1 | R/W | ISO7816_ERR_SIG_EN | 0 : Error Signal feature disabled 1 : Error Signal feature enabled | 0x0 |
| 0 | R/W | ISO7816_CONVENTION | 0 : Direct convention 1 : Inverse convention | 0x0 |

Table 503: UART3_SRBR_STHR0_REG (0x50020230)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 504: UART3_SRBR_STHR1_REG (0x50020234)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>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.</p> | |

Table 505: UART3_SRBR_STHR2_REG (0x50020238)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> | 0x0 |

Table 506: UART3_SRBR_STHR3_REG (0x5002023C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 507: UART3_SRBR_STHR4_REG (0x50020240)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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,</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>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.</p> | |

Table 508: UART3_SRBR_STHR5_REG (0x50020244)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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,</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 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 509: UART3_SRBR_STHR6_REG (0x50020248)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 510: UART3_SRBR_STHR7_REG (0x5002024C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 511: UART3_SRBR_STHR8_REG (0x50020250)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 512: UART3_SRBR_STHR9_REG (0x50020254)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 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 513: UART3_SRBR_STHR10_REG (0x50020258)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 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 514: UART3_SRBR_STHR11_REG (0x5002025C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | Shadow Receive Buffer Register x: This is a shadow register for the RBR and has been | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 515: UART3_SRBR_STHR12_REG (0x50020260)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>Shadow Transmit Holding Register</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> | |

Table 516: UART3_SRBR_STHR13_REG (0x50020264)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | configuration. Any attempt to write data when the FIFO is full results in the write data being lost. | |

Table 517: UART3_SRBR_STHR14_REG (0x50020268)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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.</p> <p>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.</p> | 0x0 |

Table 518: UART3_SRBR_STHR15_REG (0x5002026C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R/W | SRBR_STHRx | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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.</p> | |

Table 519: UART3_USR_REG (0x5002027C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4 | R | UART_RFF | <p>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.</p> | 0x0 |
| 3 | R | UART_RFNE | <p>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.</p> | 0x0 |
| 2 | R | UART_TFE | <p>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</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | empty. | |
| 1 | R | UART_TFNF | <p>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.</p> | 0x1 |
| 0 | R | UART_BUSY | <p>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.</p> | 0x0 |

Table 520: UART3_TFL_REG (0x50020280)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------------|---|-------|
| 4:0 | R | UART_TRANSMIT_FIFO_LEVEL | <p>Transmit FIFO Level. This indicates the number of data entries in the transmit FIFO.</p> | 0x0 |

Table 521: UART3_RFL_REG (0x50020284)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------------|---|-------|
| 4:0 | R | UART_RECEIVE_FIFO_LEVEL | <p>Receive FIFO Level. This indicates the number of data entries in the receive FIFO.</p> | 0x0 |

Table 522: UART3_SRR_REG (0x50020288)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:3 | - | - | | 0x0 |
| 2 | W | UART_XFR | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| | | | necessary to clear this bit. | |
| 1 | W | UART_RFR | <p>RCVR FIFO Reset.</p> <p>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.</p> <p>Note that this bit is 'self-clearing'. It is not necessary to clear this bit.</p> | 0x0 |
| 0 | W | UART_UR | <p>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.</p> | 0x0 |

Table 523: UART3_SRTS_REG (0x5002028C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_SHADOW_REQUEST_TO_SEND | <p>Shadow Request to Send.</p> <p>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.</p> <p>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.</p> <p>In Auto Flow Control, (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).</p> <p>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.</p> | 0x0 |

Table 524: UART3_SBCR_REG (0x50020290)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_SHADOW_BREAK_CONTROL | <p>Shadow Break Control Bit.</p> <p>This is a shadow register for the Break bit (LCR[6]), this can be used to remove the burden of having to performing a read modify write on the LCR. This is used to cause a break condition to be transmitted to the receiving device.</p> <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | forced low until the Break bit is cleared. | |

Table 525: UART3_SDMAM_REG (0x50020294)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_SHADOW_DMA_MODE | <p>Shadow DMA Mode.</p> <p>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.</p> <p>0 = mode 0 1 = mode 1</p> | 0x0 |

Table 526: UART3_SFE_REG (0x50020298)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_SHADOW_FIFO_ENABLE | <p>Shadow FIFO Enable.</p> <p>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.</p> | 0x0 |

Table 527: UART3_SRT_REG (0x5002029C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------------|--|-------|
| 31:2 | - | - | | 0x0 |
| 1:0 | R/W | UART_SHADOW_RCVR_TRIGGER | <p>Shadow RCVR Trigger.</p> <p>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.</p> <p>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:</p> <p>00 = 1 character in the FIFO 01 = FIFO 1/4 full</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 10 = FIFO ½ full 11 = FIFO 2 less than full | |

Table 528: UART3_STET_REG (0x500202A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------------------|--|-------|
| 31:2 | - | - | | 0x0 |
| 1:0 | R/W | UART_SHADOW_T X_EMPTY_TRIGGER | <p>Shadow TX Empty Trigger.</p> <p>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.</p> <p>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:</p> <ul style="list-style-type: none"> 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO ¼ full 11 = FIFO ½ full | 0x0 |

Table 529: UART3_HTX_REG (0x500202A4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R/W | UART_HALTED_TX | <p>This register is used to halt transmissions, so that the transmit FIFO can be filled by the master when FIFOs are implemented and enabled.</p> <p>0 = Halt TX disabled 1 = Halt TX enabled</p> <p>Note, if FIFOs are not enabled, the setting of the halt TX register has no effect on operation.</p> | 0x0 |

Table 530: UART3_DMASA_REG (0x500202A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | W | UART_DMASA | <p>This register is used 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_app_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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 531: UART3_DLF_REG (0x500202C0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:4 | - | - | | 0x0 |
| 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 532: UART3_RAR_REG (0x500202C4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| 7:0 | R/W | UART_RAR | <p>This is an address matching register during receive mode. If the 9-th bit is set in the incoming character then the remaining 8-bits will be checked against this register value. If the match happens then subsequent characters with 9-th bit set to 0 will be treated as data byte until the next address byte is received.</p> <p>Note:</p> <ul style="list-style-type: none"> - This register is applicable only when 'ADDR_MATCH'(LCR_EXT[1] and 'DLS_E' (LCR_EXT[0]) bits are set to 1. <p>RAR should be programmed only when UART is not busy.</p> | 0x0 |

Table 533: UART3_TAR_REG (0x500202C8)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| 7:0 | R/W | UART_TAR | <p>This is an address matching register during transmit mode. If DLS_E (LCR_EXT[0]) bit is enabled, then uart will send the 9-bit character with 9-th bit set to 1 and remaining 8-bit address will be sent from this register provided 'SEND_ADDR' (LCR_EXT[2]) bit is set to 1.</p> <p>Note:</p> <ul style="list-style-type: none"> - This register is used only to send the address. The normal data should be sent by programming THR register. - Once the address is started to send on the DW_apb_uart serial lane, then 'SEND_ADDR' bit will be auto-cleared by the hardware. | 0x0 |

Table 534: UART3_LCR_EXT (0x500202CC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|--|-------|
| 3 | R/W | UART_TRANSMIT_MODE | <p>Transmit mode control bit. This bit is used to control the type of transmit mode during 9-bit data transfers.</p> <p>1 = In this mode of operation, Transmit Holding Register (THR) and Shadow Transmit Holding Register (STHR) are 9-bit wide. The user needs to ensure that the THR/STHR register is written correctly for address/data.</p> <p>Address: 9th bit is set to 1,</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| | | | <p>Data : 9th bit is set to 0.</p> <p>Note: Transmit address register (TAR) is not applicable in this mode of operation.</p> <p>0 = In this mode of operation, Transmit Holding Register (THR) and Shadow Transmit Holding register (STHR) are 8-bit wide. The user needs to program the address into Transmit Address Register (TAR) and data into the THR/STHR register. SEND_ADDR bit is used as a control knob to indicate the uart on when to send the address.</p> | |
| 2 | R/W | UART_SEND_ADDR | <p>Send address control bit. This bit is used as a control knob for the user to determine when to send the address during transmit mode.</p> <p>1 = 9-bit character will be transmitted with 9-th bit set to 1 and the remaining 8-bits will match to what is being programmed in 'Transmit Address Register'.</p> <p>0 = 9-bit character will be transmitted with 9-th bit set to 0 and the remaining 8-bits will be taken from the TXFIFO which is programmed through 8-bit wide THR/STHR register.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This bit is auto-cleared by the hardware, after sending out the address character. User is not expected to program this bit to 0. 2. This field is applicable only when DLS_E bit is set to 1 and TRANSMIT_MODE is set to 0. | 0x0 |
| 1 | R/W | UART_ADDR_MATCH | <p>Address Match Mode. This bit is used to enable the address match feature during receive.</p> <p>1 = Address match mode; uart will wait until the incoming character with 9-th bit set to 1. And further checks to see if the address matches with what is programmed in 'Receive Address Match Register'. If match is found, then sub-sequent characters will be treated as valid data and DW_apb_uart starts receiving data.</p> <p>0 = Normal mode; DW_apb_uart will start to receive the data and 9-bit character will be formed and written into the receive RXFIFO. User is responsible to read the data and differentiate b/n address and data.</p> <p>Note: This field is applicable only when DLS_E is set to 1.</p> | 0x0 |
| 0 | R/W | UART_DLS_E | Extension for DLS. This bit is used to enable 9-bit data for transmit and receive transfers. | 0x0 |

Table 535: UART3_CTRL_REG (0x500202E0)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|---|-------|
| 31:12 | - | - | | 0x0 |
| 11 | R/W | ISO7816_AUTO_GT | 0 : uart sends when tx data is available 1 : uart sends new character after guard time | 0x0 |
| 10 | R/W | ISO7816_ERR_TX_ | 0 : ERR_TX_VALUE IRQ is masked | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------------|---|-------|
| | | VALUE_IRQMASK | 1 : ERR_TX_VALUE IRQ is enabled | |
| 9 | R/W | ISO7816_ERR_TX_TIME_IRQMASK | 0 : ERR_TX_TIME IRQ is masked 1 : ERR_TX_TIME IRQ is enabled | 0x0 |
| 8 | R/W | ISO7816_TIM_EXPIRED_IRQMASK | 0 : timer expired IRQ is masked 1 : timer expired IRQ is enabled | 0x0 |
| 7 | R | ISO7816_CLK_STATUS | 0 : iso7816 clock is stopped 1 : iso7816 clock is running | 0x0 |
| 6 | R/W | ISO7816_CLK_LEVEL | 0 : iso7816 clock level low when stopped 1 : iso7816 clock level high when stopped | 0x0 |
| 5 | R/W | ISO7816_CLK_EN | 0 : iso7816 clock disabled 1 : iso7816 clock enabled | 0x0 |
| 4:0 | R/W | ISO7816_CLK_DIV | ISO7816 clk freq = sclk/(2*(ISO7816_CLK_DIV+1)) | 0x0 |

Table 536: UART3_TIMER_REG (0x500202E4)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 31:18 | - | - | | 0x0 |
| 17 | R/W | ISO7816_TIM_MODE | 0 : Timer will count up to max value then stops. Timer has to be disabled and enabled again to restart. Timer is clocked with the ISO7816 clock 1 : Timer will count guard time. ISO7816_TIM_MAX has to be 16*GuardTime-1 | 0x0 |
| 16 | R/W | ISO7816_TIM_EN | 0 : Timer is disabled 1 : Timer is enabled | 0x0 |
| 15:0 | R/W | ISO7816_TIM_MAX | On write : timer will count from 0 to ISO7816_TIM_MAX On read : gives the current timer value | 0x0 |

Table 537: UART3_ERR_CTRL_REG (0x500202E8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------------|---|-------|
| 31:9 | - | - | | 0x0 |
| 8:4 | R/W | ISO7816_ERR_PULSE_WIDTH | When Error Signal feature is enable and receive mode, it gives the width of the error signal in 1/16etu | 0x10 |
| 3:0 | R/W | ISO7816_ERR_PULSE_OFFSET | When Error Signal feature is enable and receive mode, it gives the offset of the error signal in | 0xE |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------------------|-------|
| | | | 1/16etu from the 9.6etu | |

Table 538: UART3_IRQ_STATUS_REG (0x500202EC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------------|--|-------|
| 31:3 | - | - | | 0x0 |
| 2 | R/W | ISO7816_ERR_TX_VALUE_IRQ | On read 1 : If error signal is enabled and in transmit mode, module generates IRQ when receiver does not receive correctly the character On Write 1 : Clear IRQ | 0x0 |
| 1 | R/W | ISO7816_ERR_TX_TIME_IRQ | On read 1 : If error signal is enabled and in transmit mode, module generates IRQ when it checks the error signal On Write 1 : Clear IRQ | 0x0 |
| 0 | R | ISO7816_TIM_EXPIRED_IRQ | On read 1 : when Timer is expired. Timer has to be disabled to clear the IRQ. When sclk is lower than pclk then this bit has to be checked if it's cleared before return form the IRQ Handler | 0x0 |

Table 539: UART3_UCV_REG (0x500202F8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|-------------------|----------------|
| 31:0 | R | UART_UCV | Component Version | 0x34303 12A |

Table 540: UART3_CTR_REG (0x500202FC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|-------------------------|----------------|
| 31:0 | R | UART_CTR | Component Type Register | 0x44570 110 |

42.17 USB Controller Registers

Table 541: Register map USB

| Address | Register | Description |
|------------|---------------|------------------------|
| 0x50040000 | USB_MCTRL_REG | Main Control Register) |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|-----------------|---|
| 0x50040004 | USB_XCVDIAG_REG | Transceiver diagnostic Register (for test purpose only) |
| 0x50040008 | USB_TCR_REG | Transceiver configuration Register |
| 0x5004000C | USB_UTR_REG | USB test Register (for test purpose only) |
| 0x50040010 | USB_FAR_REG | Function Address Register |
| 0x50040014 | USB_NFSR_REG | Node Functional State Register |
| 0x50040018 | USB_MAEV_REG | Main Event Register |
| 0x5004001C | USB_MAMSK_REG | Main Mask Register |
| 0x50040020 | USB_ALTEV_REG | Alternate Event Register |
| 0x50040024 | USB_ALTMSK_REG | Alternate Mask Register |
| 0x50040028 | USB_TXEV_REG | Transmit Event Register |
| 0x5004002C | USB_TXMSK_REG | Transmit Mask Register |
| 0x50040030 | USB_RXEV_REG | Receive Event Register |
| 0x50040034 | USB_RXMSK_REG | Receive Mask Register |
| 0x50040038 | USB_NAKEV_REG | NAK Event Register |
| 0x5004003C | USB_NAKMSK_REG | NAK Mask Register |
| 0x50040040 | USB_FWEV_REG | FIFO Warning Event Register |
| 0x50040044 | USB_FWMSK_REG | FIFO Warning Mask Register |
| 0x50040048 | USB_FNH_REG | Frame Number High Byte Register |
| 0x5004004C | USB_FNL_REG | Frame Number Low Byte Register |
| 0x5004007C | USB_UX20CDR_REG | Transceiver 2.0 Configuration and Diagnostics Register(for test purpose only) |
| 0x50040080 | USB_EPC0_REG | Endpoint Control 0 Register |
| 0x50040084 | USB_TXD0_REG | Transmit Data 0 Register |
| 0x50040088 | USB_TXS0_REG | Transmit Status 0 Register |
| 0x5004008C | USB_TXC0_REG | Transmit command 0 Register |
| 0x50040090 | USB_EP0_NAK_REG | EP0 INNAK and OUTNAK Register |
| 0x50040094 | USB_RXD0_REG | Receive Data 0 Register |
| 0x50040098 | USB_RXS0_REG | Receive Status 0 Register |
| 0x5004009C | USB_RXC0_REG | Receive Command 0 Register |
| 0x500400A0 | USB_EPC1_REG | Endpoint Control Register 1 |
| 0x500400A4 | USB_TXD1_REG | Transmit Data Register 1 |
| 0x500400A8 | USB_TXS1_REG | Transmit Status Register 1 |
| 0x500400AC | USB_TXC1_REG | Transmit Command Register 1 |
| 0x500400B0 | USB_EPC2_REG | Endpoint Control Register 2 |
| 0x500400B4 | USB_RXD1_REG | Receive Data Register,1 |
| 0x500400B8 | USB_RXS1_REG | Receive Status Register 1 |
| 0x500400BC | USB_RXC1_REG | Receive Command Register 1 |
| 0x500400C0 | USB_EPC3_REG | Endpoint Control Register 3 |
| 0x500400C4 | USB_TXD2_REG | Transmit Data Register 2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|-----------------------|------------------------------|
| 0x500400C8 | USB_TXS2_REG | Transmit Status Register 2 |
| 0x500400CC | USB_TXC2_REG | Transmit Command Register 2 |
| 0x500400D0 | USB_EPC4_REG | Endpoint Control Register 4 |
| 0x500400D4 | USB_RXD2_REG | Receive Data Register 2 |
| 0x500400D8 | USB_RXS2_REG | Receive Status Register 2 |
| 0x500400DC | USB_RXC2_REG | Receive Command Register 2 |
| 0x500400E0 | USB_EPC5_REG | Endpoint Control Register 5 |
| 0x500400E4 | USB_TXD3_REG | Transmit Data Register 3 |
| 0x500400E8 | USB_TXS3_REG | Transmit Status Register 3 |
| 0x500400EC | USB_TXC3_REG | Transmit Command Register 3 |
| 0x500400F0 | USB_EPC6_REG | Endpoint Control Register 6 |
| 0x500400F4 | USB_RXD3_REG | Receive Data Register 3 |
| 0x500400F8 | USB_RXS3_REG | Receive Status Register 3 |
| 0x500400FC | USB_RXC3_REG | Receive Command Register 3 |
| 0x500401A0 | USB_DMA_CTRL_REG | USB DMA control register |
| 0x500401A8 | USB_CHARGER_CTR_L_REG | USB Charger Control Register |
| 0x500401AC | USB_CHARGER_STA_T_REG | USB Charger Status Register |

Table 542: USB_MCTRL_REG (0x50040000)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------|--|-------|
| 31:15 | - | - | | 0x0 |
| 4 | R/W | LSMODE | Low Speed Mode This bit enables USB 1.5 Mbit/s low speed and swaps D+ and D- pull-up resistors. Changing speed may only be done if USBEN is set to 0. Also D+ and D- rise and fall times are adjusted according to the USB specification. | 0x0 |
| 3 | R/W | USB_NAT | Node Attached This bit indicates that this node is ready to be detected as attached to USB. When cleared to 0 the transceiver forces SE0 on the USB port to prevent the hub (to which this node is connected to) from detecting an attach event. After reset or when the USB node is disabled, this bit is cleared to 0 to give the device time before it must respond to commands. After this bit has been set to 1, the device no longer drives the USB and should be ready to receive Reset signalling from the hub. Note: This bit can only be set if USBEN is '1' | 0x0 |
| 2 | - | - | | 0x0 |
| 1 | R/W | USB_DBG | Debug Mode. When this bit is set, the following registers are writable: Main Event (MAEV), Alternate Event | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | (ALTEV), NAK Event (NAKEV), Transmit Status and Receive Status. Setting the DBG bit forces the node into a locked state. The node states can be read out of the transceiver diagnostic register (XCVDIAG) at location 0xFF6802 by setting the DIAG bit in the Test Control register (UTR). Note: The operation of CoR bits is not effected by entering Debug mode) Note: This bit can only be set if USBEN is '1' | |
| 0 | R/W | USBEN | USB EnableSetting this bit to 1 enables the Full/Low Speed USB node. If the USBEN bit is cleared to 0, the USB is disabled and the 48 MHz clock within the USB node is stopped. In addition, all USB registers are set to their reset state. Note that the transceiver forces SE0 on the bus to prevent the hub to detect the USB node, when it is disabled (not attached). The USBEN bit is cleared to 0 after reset | 0x0 |

Table 543: USB_XCVDIAG_REG (0x50040004)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | USB_VPIN | With Bit0 = 1 this bit shows the level of the USB_Dp receive data from transceiver; i.e. D+ <= VSE. | 0x0 |
| 6 | R | USB_VMIN | With Bit0 = 1 this bit shows the level USB_Dm receive data from transceiver; i.e. D- <= VSE. | 0x0 |
| 5 | R | USB_RCV | With Bit0 = 1 this bit shows the differential level of the receive comparator. | 0x0 |
| 4 | - | - | | 0x0 |
| 3 | R/W | USB_XCV_TXEN | With Bit0 = 1, this bit enables test Bits 2,1. Must be kept to '0' for normal operation | 0x0 |
| 2 | R/W | USB_XCV_TXn | With Bit3,0 = 1, this bit sets USB_Dm to a high level, independent of LSMODE selection | 0x0 |
| 1 | R/W | USB_XCV_TXp | With Bit3,0 = 1, this bit sets USB_Dp to a high level, independent of LSMODE selection | 0x0 |
| 0 | R/W | USB_XCV_TEST | Enable USB_XCVDIAG_REG 0: Normal operation, test bits disabled 1: Enable test bits 7,6,5,3,2,1 | 0x0 |

Table 544: USB_TCR_REG (0x50040008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:5 | R/W | USB_VADJ | Reference Voltage/ Threshold voltage AdjustControls the single-ended receiver threshold. Shall not be modified unless instructed by Dialog Semiconductor Only enabled if USB_UTR_REG[7] = 1 | 0x4 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| 4:0 | R/W | USB_CADJ | Transmitter Current Adjust Controls the driver edge rate control current. Shall not be modified unless instructed by Dialog Semiconductor Only enabled if USB_UTR_REG[7] = 1 | 0x10 |

Table 545: USB_UTR_REG (0x5004000C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_DIAG | Diagnostic enable '0': Normal operational. '1': Access to the USB_XCVDIAG_REG and USB_TCR_REG enabled. For diagnostic purposes only | 0x0 |
| 6 | R/W | USB_NCRC | No CRC16 When this bit is set to 1, all packets transmitted by the Full/Low Speed USB node are sent without a trailing CRC16. Receive operations are unaffected. This mode is used to check that CRC errors can be detected by other nodes. For diagnostic purposes only | 0x0 |
| 5 | R/W | USB_SF | Short Frame Enables the Frame timer to lock and track short, non-compliant USB frame sizes. The Short Frame bit should not be set during normal operation. For test purposes only | 0x0 |
| 4:0 | R/W | USB_UTR_RES | Reserved. Must be kept to '0' | 0x0 |

Table 546: USB_FAR_REG (0x50040010)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_AD_EN | Address Enable When set to 1, USB address field bits 6-0 are used in address comparison When cleared to 0, the device does not respond to any token on the USB bus. Note: If the DEF bit in the Endpoint Control 0 register is set, Endpoint 0 responds to the default address. | 0x0 |
| 6:0 | R/W | USB_AD | Address This field holds the 7-bit function address used to transmit and receive all tokens addressed to this device. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 547: USB_NFSR_REG (0x50040014)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|--|-------|
| 31:2 | - | - | | 0x0 |
| 1:0 | R/W | USB_NFS | <p>The Node Functional State Register reports and controls the current functional state of the USB node.</p> <p>00: NodeReset. This is the USB Reset state. This is entered upon a module reset or by software upon detection of a USB Reset. Upon entry, all endpoint pipes are disabled. DEF in the Endpoint Control 0 (EPC0) register and AD_EN in the Function Address (FAR) register should be cleared by software on entry to this state. On exit, DEF should be reset so the device responds to the default address.</p> <p>01: NodeResume In this state, resume signalling is generated. This state should be entered by firmware to initiate a remote wake-up sequence by the device. The node must remain in this state for at least 1 ms and no more than 15 ms.</p> <p>10: NodeOperational This is the normal operational state. In this state the node is configured for operation on the USB bus.</p> <p>11: NodeSuspend Suspend state should be entered by firmware on detection of a Suspend event while in Operational state. While in Suspend state, the transceivers operate in their low-power suspend mode. All endpoint controllers and the bits TX_EN, LAST and RX_EN are reset, while all other internal states are frozen. On detection of bus activity, the RESUME bit in the ALTEV register is set. In response, software can cause entry to NodeOperational state.</p> | 0x0 |

Table 548: USB_MAEV_REG (0x50040018)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 31:12 | - | - | | 0x0 |
| 11 | R/W | USB_CH_EV | <p>USB Charger event This bit is set if one of the bits in USB_CHARGER_STAT_REG[2-0] change. This bit is cleared to 0 when if USB_CHARGER_STAT_REG is read.</p> | 0x0 |
| 10 | R/W | USB_EP0_NAK | <p>Endpoint 0 NAK Event This bit is an OR of EP0_NAK_REG[EP0_OUTNAK] and EP0_NAK_REG[EP0_INNAK] bits. USB_EP0_NAK is cleared to 0 when EP0_NAK_REG is read.</p> | 0x0 |
| 9 | R/W | USB_EP0_RX | <p>Endpoint 0 Receive Event This bit is a copy of the RXS0[RX_LAST] and is cleared to 0 when this RXS0 register is read. Note: Since Endpoint 0 implements a store and forward principle, an overrun condition for FIFO0 cannot occur</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| 8 | R/W | USB_EP0_TX | <p>Endpoint 0 Transmit Event This bit is a copy of the TXS0[TX_DONE] bit and is cleared to 0 when the TXS0 register is read. Note: Since Endpoint 0 implements a store and forward principle, an underrun condition for FIFO0 cannot occur.</p> | 0x0 |
| 7 | R/W | USB_INTR | <p>Master Interrupt Enable This bit is hardwired to 0 in the Main Event (MAEV) register; bit 7 in the Main Mask (MAMSK) register is the Master Interrupt Enable.</p> | 0x0 |
| 6 | R/W | USB_RX_EV | <p>Receive Event This bit is set to 1 if any of the unmasked bits in the Receive Event (RXEV) register is set to 1. It indicates that a SETUP or OUT transaction has been completed. This bit is cleared to 0 when all of the RX_LAST bits in each Receive Status (RXSn) register and all RXOVRRN bits in the RXEV register are cleared to 0.</p> | 0x0 |
| 5 | R/W | USB_ULD | <p>Unlocked/Locked Detected This bit is set to 1, when the frame timer has either entered unlocked condition from a locked condition, or has re-entered a locked condition from an unlocked condition as determined by the UL bit in the Frame Number (FNH or FNL) register. This bit is cleared to 0 when the register is read.</p> | 0x0 |
| 4 | R/W | USB_NAK | <p>Negative Acknowledge Event This bit indicates that one of the unmasked NAK Event (NAKEV) register bits has been set to 1. This bit is cleared to 0 when the NAKEV register is read.</p> | 0x0 |
| 3 | R/W | USB_FRAME | <p>Frame Event This bit is set to 1, if the frame counter is updated with a new value. This can be due to the receipt of a valid SOF packet on the USB or to an artificial update if the frame counter was unlocked or a frame was missed. This bit is cleared to 0 when the register is read.</p> | 0x0 |
| 2 | R/W | USB_TX_EV | <p>Transmit Event This bit is set to 1, if any of the unmasked bits in the Transmit Event (TXEV) register (TXFIFO or TXUNDRNn) is set to 1. Therefore, it indicates that an IN transaction has been completed. This bit is cleared to 0 when all the TX_DONE bits and the TXUNDRN bits in each Transmit Status (TXSn) register are cleared to 0.</p> | 0x0 |
| 1 | R/W | USB_ALT | <p>Alternate Event This bit indicates that one of the unmasked ALTEV register bits has been set to 1. This bit is cleared to 0 by reading the ALTEV register.</p> | 0x0 |
| 0 | R/W | USB_WARN | <p>Warning Event This bit indicates that one of the unmasked bits in the FIFO Warning Event (FWEV) register has been set to 1. This bit is cleared to 0 by reading the FWEV register.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 549: USB_MAMSK_REG (0x5004001C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---|-------|
| 31:12 | - | - | | 0x0 |
| 11 | R/W | USB_M_CH_EV | The Main Mask Register masks out events reported in the MAEV registers. A bit set to 1, enables the interrupts for the respective event in the MAEV register. If the corresponding bit is cleared to 0, interrupt generation for this event is disabled. Same Bit Definition as MAEV Register | 0x0 |
| 10 | R/W | USB_M_EP0_NAK | Same Bit Definition as MAEV Register | 0x0 |
| 9 | R/W | USB_M_EP0_RX | Same Bit Definition as MAEV Register | 0x0 |
| 8 | R/W | USB_M_EP0_TX | Same Bit Definition as MAEV Register | 0x0 |
| 7 | R/W | USB_M_INTR | Same Bit Definition as MAEV Register | 0x0 |
| 6 | R/W | USB_M_RX_EV | Same Bit Definition as MAEV Register | 0x0 |
| 5 | R/W | USB_M_ULD | Same Bit Definition as MAEV Register | 0x0 |
| 4 | R/W | USB_M_NAK | Same Bit Definition as MAEV Register | 0x0 |
| 3 | R/W | USB_M_FRAME | Same Bit Definition as MAEV Register | 0x0 |
| 2 | R/W | USB_M_TX_EV | Same Bit Definition as MAEV Register | 0x0 |
| 1 | R/W | USB_M_ALT | Same Bit Definition as MAEV Register | 0x0 |
| 0 | R/W | USB_M_WARN | Same Bit Definition as MAEV Register | 0x0 |

Table 550: USB_ALTEV_REG (0x50040020)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_RESUME | Resume Resume signalling is detected on the USB when the device is in Suspend state (NFS in the NFSR register is set to SUSPEND), and a non IDLE signal is present on the USB, indicating that this device should begin its wake-up sequence and enter Operational state. This bit is cleared when the register is read. | 0x0 |
| 6 | R/W | USB_RESET | Reset This bit is set to 1, when 2.5 us of SEO have been detected on the upstream port. In response, the functional state should be reset (NFS in the NFSR register is set to RESET), where it must remain for at least 100 us. The functional state can then return to Operational state. This bit is cleared when the register is read | 0x0 |
| 5 | R/W | USB_SD5 | Suspend Detect 5 ms This bit is set to 1 after 5 ms of IDLE have been detected on the upstream port, indicating that this device is permitted to perform a remote wake-up operation. The resume may be initiated under firmware control by writing the resume value to the NFSR register. This bit is cleared when the register is read. | 0x0 |
| 4 | R/W | USB_SD3 | Suspend Detect 3 ms | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|--|-------|
| | | | This bit is set to 1 after 3 ms of IDLE have been detected on the upstream port, indicating that the device should be suspended. The suspend occurs under firmware control by writing the suspend value to the Node Functional State (NFSR) register. This bit is cleared when the register is read. | |
| 3 | R/W | USB_EOP | End of Packet A valid EOP sequence was been detected on the USB. It is used when this device has initiated a Remote wake-up sequence to indicate that the Resume sequence has been acknowledged and completed by the host. This bit is cleared when the register is read. | 0x0 |
| 2 | - | - | | 0x0 |
| 1:0 | - | - | | 0x0 |

Table 551: USB_ALTMSK_REG (0x50040024)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_M_RESUME | A bit set to 1 in this register enables automatic setting of the ALT bit in the MAEV register when the respective event in the ALTEV register occurs. Otherwise, setting MAEV.ALTD bit is disabled. Same Bit Definition as ALTEV Register | 0x0 |
| 6 | R/W | USB_M_RESET | Same Bit Definition as ALTEV Register | 0x0 |
| 5 | R/W | USB_M_SD5 | Same Bit Definition as ALTEV Register | 0x0 |
| 4 | R/W | USB_M_SD3 | Same Bit Definition as ALTEV Register | 0x0 |
| 3 | R/W | USB_M_EOP | Same Bit Definition as ALTEV Register | 0x0 |
| 2 | - | - | | 0x0 |
| 1:0 | - | - | | 0x0 |

Table 552: USB_TXEV_REG (0x50040028)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R | USB_TXUDRRN31 | Transmit Underrun n: 3:1 The bit n is a copy of the respective TX_URUN bit from the corresponding Transmit Status register (TXSn). Whenever any of the Transmit FIFOs underflows, the respective TXUDRRN bit is set to 1. These bits are cleared to 0 when the corresponding Transmit Status register is read | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R | USB_TXFIFO31 | Transmit FIFO n: 3:1 The bit n is a copy of the TX_DONE bit from the corresponding Transmit Status register (TXSn). A bit is set to 1 when the IN transaction for the corresponding transmit endpoint n has been | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | completed. These bits are cleared to 0 when the corresponding TXSn register is read. | |

Table 553: USB_TXMSK_REG (0x5004002C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|---|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R/W | USB_M_TXUDRRN31 | The Transmit Mask Register is used to select the bits of the TXEV registers, which causes the TX_EV bit in the MAEV register to be set to 1. When a bit is set to 1 and the corresponding bit in the TXEV register is set to 1, the TX_EV bit in the MAEV register is set to 1. When cleared to 0, the corresponding bit in the TXEV register does not cause TX_EV to be set to 1. Same Bit Definition as TXEV Register | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R/W | USB_M_TXFIFO31 | Same Bit Definition as TXEV Register | 0x0 |

Table 554: USB_RXEV_REG (0x50040030)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R | USB_RXOVERRN31 | Receive Overrun n: 3:1 The bit n is set to 1 in the event of an overrun condition in the corresponding receive FIFO n. They are cleared to 0 when the register is read. The firmware must check the respective RX_ERR bits that packets received for the other receive endpoints (EP2, EP4 and EP6,) are not corrupted by errors, as these endpoints support data streaming (packets which are longer than the actual FIFO depth). | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R | USB_RXFIFO31 | Receive FIFO n: 3:1 The bit n is set to 1 whenever either RX_ERR or RX_LAST in the respective Receive Status register (RXSn) is set to 1. Reading the corresponding RXSn register automatically clears these bits. The CoR function is disabled, when the Freeze signal is asserted. The USB node discards all packets for Endpoint 0 received with errors. This is necessary in case of retransmission due to media errors, ensuring that a good copy of a SETUP packet is captured. Otherwise, the FIFO may potentially be tied up, holding corrupted data and unable to receive a retransmission of the same packet. If data streaming is used for the receive endpoints (EP2, EP4 and EP6, EP8) the firmware must check the respective RX_ERR bits to ensure the packets received are not corrupted by errors. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 555: USB_RXMSK_REG (0x50040034)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R/W | USB_M_RXOVRN31 | The Receive Mask Register is used to select the bits of the RXEV registers, which causes the RX_EV bit in the MAEV register to be set to 1. When set to 1 and the corresponding bit in the RXEV register is set to 1, RX_EV bit in the MAEV register is set to 1. When cleared to 0, the corresponding bit in the RXEV register does not cause RX_EV to be set to 1. Same Bit Definition as RXEV Register | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R/W | USB_M_RXFIFO31 | Same Bit Definition as RXEV Register | 0x0 |

Table 556: USB_NAKEV_REG (0x50040038)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R | USB_OUT31 | OUT n: 3:1 The bit n is set to 1 when a NAK handshake is generated for an enabled address/endpoint combination (AD_EN in the FAR register is set to 1 and EP_EN in the EPCx register is set to 1) in response to an OUT token. This bit is not set if NAK is generated as result of an overrun condition. It is cleared when the register is read. | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R | USB_IN31 | IN n: 3:1 The bit n is set to 1 when a NAK handshake is generated for an enabled address/endpoint combination (AD_EN in the Function Address, FAR, register is set to 1 and EP_EN in the Endpoint Control, EPCx, register is set to 1) in response to an IN token. This bit is cleared when the register is read. | 0x0 |

Table 557: USB_NAKMSK_REG (0x5004003C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R/W | USB_M_OUT31 | When set and the corresponding bit in the NAKEV register is set, the NAK bit in the MAEV register is set. When cleared, the corresponding bit in the NAKEV register does not cause NAK to be set. Same Bit Definition as NAKEV Register | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R/W | USB_M_IN31 | Same Bit Definition as NAKEV Register | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 558: USB_FWEV_REG (0x50040040)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R | USB_RXWARN31 | Receive Warning n: 3:1 The bit n is set to 1 when the respective receive endpoint FIFO reaches the warning limit, as specified by the RFWL bits of the respective EPCx register. This bit is cleared when the warning condition is cleared by either reading data from the FIFO or when the FIFO is flushed. | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R | USB_TXWARN31 | Transmit Warning n: 3:1 The bit n is set to 1 when the respective transmit endpoint FIFO reaches the warning limit, as specified by the TFWL bits of the respective TXCn register, and transmission from the respective endpoint is enabled. This bit is cleared when the warning condition is cleared by either writing new data to the FIFO when the FIFO is flushed, or when transmission is done, as indicated by the TX_DONE bit in the TXSn register. | 0x0 |

Table 559: USB_FWMSK_REG (0x50040044)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6:4 | R/W | USB_M_RXWARN31 | The FIFO Warning Mask Register selects, which FWEV bits are reported in the MAEV register. A bit set to 1 and the corresponding bit in the FWEV register is set 1, causes the WARN bit in the MAEV register to be set to 1. When cleared to 0, the corresponding bit in the FWEV register does not cause WARN to be set to 1. Same Bit Definition as FWEV Register | 0x0 |
| 3 | - | - | | 0x0 |
| 2:0 | R/W | USB_M_TXWARN31 | The FIFO Warning Mask Register selects, which FWEV bits are reported in the MAEV register. A bit set to 1 and the corresponding bit in the FWEV register is set 1, causes the WARN bit in the MAEV register to be set to 1. When cleared to 0, the corresponding bit in the FWEV register does not cause WARN to be set to 1. Same Bit Definition as FWEV Register | 0x0 |

Table 560: USB_FNH_REG (0x50040048)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | USB_MF | Missed SOF Flag This flag is set to 1, when the frame number in a valid received SOF does not match the expected next value, or when an SOF is not received within 12060 bit times. This bit is set by the hardware and | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| | | | is cleared by reading the FNH register. | |
| 6 | R | USB_UL | Unlock Flag This bit indicates that at least two frames were received without an expected frame number, or that no valid SOF was received within 12060 bit times. If this bit is set, the frame number from the next valid SOF packet is loaded in FN. This bit is set by the hardware and is cleared by reading the FNH register. | 0x1 |
| 5 | R | USB_RFC | Reset Frame Count Writing a 1 to this bit resets the frame number to 00016, after which this bit clears itself to 0 again. This bit always reads 0. | 0x0 |
| 4:3 | - | - | | 0x0 |
| 2:0 | R | USB_FN_10_8 | Frame Number This 3-bit field contains the three most significant bits (MSB) of the current frame number, received in the last SOF packet. If a valid frame number is not received within 12060 bit times (Frame Length Maximum, FLMAX, with tolerance) of the previous change, the frame number is incremented artificially. If two successive frames are missed or are incorrect, the current FN is frozen and loaded with the next frame number from a valid SOF packet. If the frame number low byte was read by firmware before reading the FNH register, the user actually reads the contents of a buffer register which holds the value of the three frame number bits of this register when the low byte was read. Therefore, the correct sequence to read the frame number is: FNL, FNH. Read operations to the FNH register, without first reading the Frame Number Low Byte (FNL) register directly, read the actual value of the three MSBs of the frame number. | 0x0 |

Table 561: USB_FNL_REG (0x5004004C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R | USB_FN | The Frame Number Low Byte Register holds the low byte of the frame number. To ensure consistency, reading this low byte causes the three frame number bits in the FNH register to be locked until this register is read. The correct sequence to read the frame number is: FNL, FNH. | 0x0 |

Table 562: USB_UX20CDR_REG (0x5004007C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|-------------|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | RPU_TEST7 | Test bit | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|--|-------|
| 6 | R/W | RPU_TEST_SW2 | 0: Closes SW2 switch to reduced pull-up resistor connected to the USB_Dp and USB_Dm. 1: Opens SW2 switch resistor connected to the USB_Dp and USB_Dm (independent of the VBus state). | 0x0 |
| 5 | R/W | RPU_TEST_SW1 | 0: Enable the pull-up resistor on USB_Dp (SW1 closed) 1: Disable the pull-up resistor on USB_Dp (SW1 open) (Independent of the VBus state). | 0x0 |
| 4 | R/W | RPU_TEST_EN | Pull-Up Resistor Test Enable 0: Normal operation 1: Enables the test features controlled by RPU_TEST_SW1, RPU_TEST_SW1DM and RPU_TEST_SW2 | 0x0 |
| 3 | - | - | | 0x0 |
| 2 | R/W | RPU_TEST_SW1DM | 0: Enable the pull-up resistor on USB_Dm (SW1DM closed) 1: Disable the pull-up resistor on USB_Dm (SW1DM open) (Independent of the VBus state). | 0x0 |
| 1 | R/W | RPU_RCDELAY | Test bit, must be kept 0 | 0x0 |
| 0 | R/W | RPU_SSPPROTN | Test bit, must be kept 0 | 0x0 |

Table 563: USB_EPC0_REG (0x50040080)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_STALL | <p>Stall</p> <p>Setting this bit to 1 causes the chip to generate STALL handshakes under the following conditions:</p> <ul style="list-style-type: none"> - The transmit FIFO is enabled and an IN token is received. - The receive FIFO is enabled and an OUT token is received. <p>Note: A SETUP token does not cause a STALL handshake to be generated when this bit is set.</p> <p>Upon transmitting the STALL handshake, the RX_LAST and the TX_DONE bits in the respective Receive/Transmit Status registers are set to 1.</p> | 0x0 |
| 6 | R/W | USB_DEF | <p>Default Address</p> <p>When set to 1, the device responds to the default address regardless of the contents of FAR6-0/EP03-0 fields. When an IN packet is transmitted for the endpoint, the DEF bit is automatically cleared to 0.</p> <p>This bit aids in the transition from default address to assigned address. The transition from the default address 00000000000b to an address assigned during bus enumeration may not occur in the middle of the SET_ADDRESS control sequence. This is necessary to complete the control sequence. However, the address must change immediately after this sequence finishes in order to avoid errors</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>when another control sequence immediately follows the SET_ADDRESS command.</p> <p>On USB reset, the firmware has 10 ms for set-up, and should write 8016 to the FAR register and 0016 to the EPC0 register. On receipt of a SET_ADDRESS command, the firmware must write 4016 to the EPC0 register and (8016 or <assigned_function_address>) to the FAR register. It must then queue a zero length IN packet to complete the status phase of the SET_ADDRESS control sequence.</p> | |
| 5:4 | - | - | | 0x0 |
| 3:0 | R | USB_EP | <p>Endpoint Address</p> <p>This field holds the 4-bit Endpoint address. For Endpoint 0, these bits are hardwired to 0000b. Writing a 1 to any of the EP bits is ignored.</p> | 0x0 |

Table 564: USB_TXD0_REG (0x50040084)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | W | USB_TXFD | <p>Transmit FIFO Data Byte</p> <p>The firmware is expected to write only the packet payload data. The PID and CRC16 are created automatically.</p> | 0x0 |

Table 565: USB_TXS0_REG (0x50040088)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6 | R | USB_ACK_STAT | <p>Acknowledge Status</p> <p>This bit indicates the status, as received from the host, of the ACK for the packet previously sent. This bit is to be interpreted when TX_DONE is set to 1. It is set to 1, when an ACK is received; otherwise, it remains cleared. This bit is also cleared to 0, when this register is read.</p> | 0x0 |
| 5 | R | USB_TX_DONE | <p>Transmission Done</p> <p>When set to 1, this bit indicates that a packet has completed transmission. It is cleared to 0, when this register is read.</p> | 0x0 |
| 4:0 | R | USB_TCOUNT | <p>Transmission Count</p> <p>This 5-bit field indicates the number of empty bytes available in the FIFO. This field is never larger than 8 for Endpoint 0.</p> | 0x8 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 566: USB_TXC0_REG (0x5004008C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4 | R/W | USB_IGN_IN | Ignore IN Tokens When this bit is set to 1, the endpoint will ignore any IN tokens directed to its configured address. | 0x0 |
| 3 | R/W | USB_FLUSH | Flush FIFO Writing a 1 to this bit flushes all data from the control endpoint FIFOs, resets the endpoint to Idle state, clears the FIFO read and write pointer, and then clears itself. If the endpoint is currently using the FIFO0 to transfer data on USB, flushing is delayed until after the transfer is done. It is equivalent to the FLUSH bit in the RXC0 register. | 0x0 |
| 2 | R/W | USB_TOGGLE_TX0 | Toggle This bit specifies the PID used when transmitting the packet. A value of 0 causes a DATA0 PID to be generated, while a value of 1 causes a DATA1 PID to be generated. This bit is not altered by the hardware. | 0x0 |
| 1 | - | - | | 0x0 |
| 0 | R/W | USB_TX_EN | Transmission Enable This bit enables data transmission from the FIFO. It is cleared to 0 by hardware after transmitting a single packet, or a STALL handshake, in response to an IN token. It must be set to 1 by firmware to start packet transmission. The RX_EN bit in the Receive Command 0 (RXC0) register takes precedence over this bit; i.e. if RX_EN is set, TX_EN bit is ignored until RX_EN is reset. Zero length packets are indicated by setting this bit without writing any data to the FIFO. | 0x0 |

Table 567: USB_EP0_NAK_REG (0x50040090)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:2 | - | - | | 0x0 |
| 1 | R | USB_EP0_OUTNAK | End point 0 OUT NAK This bit n is set to 1 when a NAK handshake is generated for an enabled address/endpoint combination (AD_EN in the FAR register is set to 1) in response to an OUT token. This bit is not set if NAK is generated as result of an overrun condition. It is cleared when the register is read. | 0x0 |
| 0 | R | USB_EP0_INNAK | End point 0 IN NAK This bit is set to 1 when a NAK handshake is generated for an enabled address/endpoint combination (AD_EN in the FAR register is set to 1) in response to an IN token. This bit is cleared when the register is read. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 568: USB_RXD0_REG (0x50040094)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R | USB_RXFD | Receive FIFO Data Byte The firmware should expect to read only the packet payload data. The PID and CRC16 are removed from the incoming data stream automatically. In TEST mode this register allow read/write access. | 0x0 |

Table 569: USB_RXS0_REG (0x50040098)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | - | - | | 0x0 |
| 6 | R | USB_SETUP | Setup This bit indicates that the setup packet has been received. This bit is unchanged for zero length packets. It is cleared to 0 when this register is read. | 0x0 |
| 5 | R | USB_TOGGLE_RX0 | Toggle This bit specified the PID used when receiving the packet. A value of 0 indicates that the last successfully received packet had a DATA0 PID, while a value of 1 indicates that this packet had a DATA1 PID. This bit is unchanged for zero length packets. It is cleared to 0 when this register is read. | 0x0 |
| 4 | R | USB_RX_LAST | Receive Last Bytes This bit indicates that an ACK was sent upon completion of a successful receive operation. This bit is unchanged for zero length packets. It is cleared to 0 when this register is read. | 0x0 |
| 3:0 | R | USB_RCOUNT | Receive Count This 4-bit field contains the number of bytes presently in the RX FIFO. This number is never larger than 8 for Endpoint 0. | 0x0 |

Table 570: USB_RXC0_REG (0x5004009C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:6 | - | - | | 0x0 |
| 5 | - | - | | 0x0 |
| 4 | - | - | | 0x0 |
| 3 | R/W | USB_FLUSH | Flush Writing a 1 to this bit flushes all data from the control endpoint FIFOs, resets the endpoint to Idle state, clears the FIFO read and write pointer, and then clears itself. If the endpoint is currently using FIFO0 to transfer data on USB, flushing is delayed until after the transfer is done. This bit is cleared to 0 on reset. This bit is equivalent to FLUSH in the TXC0 register. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| 2 | R/W | USB_IGN_SETUP | Ignore SETUP Tokens When this bit is set to 1, the endpoint ignores any SETUP tokens directed to its configured address. | 0x0 |
| 1 | R/W | USB_IGN_OUT | Ignore OUT Tokens When this bit is set to 1, the endpoint ignores any OUT tokens directed to its configured address. | 0x0 |
| 0 | R/W | USB_RX_EN | Receive Enable OUT packet reception is disabled after every data packet is received, or when a STALL handshake is returned in response to an OUT token. A 1 must be written to this bit to re-enable data reception. Reception of SETUP packets is always enabled. In the case of back-to-back SETUP packets (for a given endpoint) where a valid SETUP packet is received with no other intervening non-SETUP tokens, the Endpoint Controller discards the new SETUP packet and returns an ACK handshake. If any other reasons prevent the Endpoint Controller from accepting the SETUP packet, it must not generate a handshake. This allows recovery from a condition where the ACK of the first SETUP token was lost by the host. | 0x0 |

Table 571: USB_EPC1_REG (0x500400A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_STALL | Stall Setting this bit to 1 causes the chip to generate STALL handshakes under the following conditions: The transmit FIFO is enabled and an IN token is received. The receive FIFO is enabled and an OUT token is received. Setting this bit to 1 does not generate a STALL handshake in response to a SETUP token | 0x0 |
| 6 | - | - | | 0x0 |
| 5 | R/W | USB_ISO | Isochronous When this bit is set to 1, the endpoint is isochronous. This implies that no NAK is sent if the endpoint is not ready but enabled; i.e. If an IN token is received and no data is available in the FIFO to transmit, or if an OUT token is received and the FIFO is full since there is no USB handshake for isochronous transfers. | 0x0 |
| 4 | R/W | USB_EP_EN | Endpoint Enable When this bit is set to 1, the EP[3:0] field is used in address comparison, together with the AD[6:0] field in the FAR register. When cleared to 0, the endpoint does not respond to any token on the USB bus. | 0x0 |
| 3:0 | R/W | USB_EP | Endpoint Address This 4-bit field holds the endpoint address. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 572: USB_TXD1_REG (0x500400A4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | W | USB_TXFIFO | <p>Transmit FIFO Data Byte</p> <p>The firmware is expected to write only the packet payload data. PID and CRC16 are inserted automatically in the transmit data stream.</p> <p>In TEST mode this register allow read/write access via the core bus.</p> | 0x0 |

Table 573: USB_TXS1_REG (0x500400A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | USB_TX_URUN | <p>Transmit FIFO Underrun</p> <p>This bit is set to 1, if the transmit FIFO becomes empty during a transmission, and no new data is written to the FIFO. If so, the Media Access Controller (MAC) forces a bit stuff error followed by an EOP. This bit is cleared to 0, when this register is read.</p> | 0x0 |
| 6 | R | USB_ACK_STAT | <p>Acknowledge Status</p> <p>This bit is interpreted when TX_DONE is set. Its function differs depending on whether ISO (ISO in the EPCx register is set) or non-ISO operation (ISO is reset) is used.</p> <p>For non-ISO operation, this bit indicates the acknowledge status (from the host) about the ACK for the previously sent packet. This bit itself is set to 1, when an ACK is received; otherwise, it is cleared to 0.</p> <p>For ISO operation, this bit is set if a frame number LSB match (see IGN_ISOMSK bit in the USB_TXCx_REG) occurs, and data was sent in response to an IN token. Otherwise, this bit is cleared to 0, the FIFO is flushed and TX_DONE is set.</p> <p>This bit is also cleared to 0, when this register is read.</p> | 0x0 |
| 5 | R | USB_TX_DONE | <p>Transmission Done</p> <p>When set to 1, this bit indicates that the endpoint responded to a USB packet. Three conditions can cause this bit to be set:</p> <p>A data packet completed transmission in response to an IN token with non-ISO operation.</p> <p>The endpoint sent a STALL handshake in response to an IN token</p> <p>A scheduled ISO frame was transmitted or discarded.</p> <p>This bit is cleared to 0 when this register is read.</p> | 0x0 |
| 4:0 | R | USB_TCOUNT | <p>Transmission Count</p> <p>This 5-bit field holds the number of empty bytes</p> | 0x1F |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | available in the FIFO. If this number is greater than 31, a value of 31 is actually reported. | |

Table 574: USB_TXC1_REG (0x500400AC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_IGN_ISOMSK | <p>Ignore ISO Mask</p> <p>This bit has an effect only if the endpoint is set to be isochronous. If set to 1, this bit disables locking of specific frame numbers with the alternate function of the TOGGLE bit. Thus data is transmitted upon reception of the next IN token. If cleared to 0, data is only transmitted when FNLO matches TOGGLE. This bit is cleared to 0 after reset.</p> | 0x0 |
| 6:5 | R/W | USB_TFWL | <p>Transmit FIFO Warning Limit</p> <p>These bits specify how many more bytes can be transmitted from the respective FIFO before an underrun condition occurs. If the number of bytes remaining in the FIFO is equal to or less than the selected warning limit, the TXWARN bit in the FWEV register is set. To avoid interrupts caused by setting this bit while the FIFO is being filled before a transmission begins, TXWARN is only set when transmission from the endpoint is enabled (TX_ENn in the TXCn register is set).</p> <p>TFWL[1:0] :</p> <ul style="list-style-type: none"> 00: TFWL disabled 01: Less than 5 bytes remaining in FIFO 10: Less than 9 bytes remaining in FIFO 11: Less than 17 bytes remaining in FIFO | 0x0 |
| 4 | R/W | USB_RFF | <p>Refill FIFO</p> <p>Setting the LAST bit to 1 automatically saves the Transmit Read Pointer (TXRP) to a buffer. When the RFF bit is set to 1, the buffered TXRP is reloaded into the TXRP. This allows the user to repeat the last transaction if no ACK was received from the host. If the MAC is currently using the FIFO to transmit, TXRP is reloaded only after the transmission is complete. After reload, this bit is cleared to 0 by hardware.</p> | 0x0 |
| 3 | R/W | USB_FLUSH | <p>Flush FIFO</p> <p>Writing a 1 to this bit flushes all data from the corresponding transmit FIFO, resets the endpoint to Idle state, and clears both the FIFO read and write pointers. If the MAC is currently using the FIFO to transmit, data is flushed after the transmission is complete. After data flushing, this bit is cleared to 0 by hardware.</p> | 0x0 |
| 2 | R/W | USB_TOGGLE_TX | <p>Toggle</p> <p>The function of this bit differs depending on whether ISO (ISO bit in the EPCn register is set to 1) or non-ISO operation (ISO bit is cleared to 0) is used.</p> <p>For non-ISO operation, it specifies the PID used</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | <p>when transmitting the packet. A value of 0 causes a DATA0 PID to be generated, while a value of 1 causes a DATA1 PID to be generated.</p> <p>For ISO operation, this bit and the LSB of the frame counter (FNL0) act as a mask for the TX_EN bit to allow pre-queuing of packets to specific frame numbers; i.e. transmission is enabled only if bit 0 in the FNL register is set to TOGGLE. If an IN token is not received while this condition is true, the contents of the FIFO are flushed with the next SOF. If the endpoint is set to ISO, data is always transferred with a DATA0 PID.</p> | |
| 1 | R/W | USB_LAST | <p>Last Byte</p> <p>Setting this bit to 1 indicates that the entire packet has been written into the FIFO. This is used especially for streaming data to the FIFO while the actual transmission occurs. If the LAST bit is not set to 1 and the transmit FIFO becomes empty during a transmission, a stuff error followed by an EOP is forced on the bus. Zero length packets are indicated by setting this bit without writing any data to the FIFO.</p> <p>The transmit state machine transmits the payload data, CRC16 and the EOP signal before clearing this bit.</p> | 0x0 |
| 0 | R/W | USB_TX_EN | <p>Transmission Enable</p> <p>This bit enables data transmission from the FIFO. It is cleared to 0 by hardware after transmitting a single packet or after a STALL handshake in response to an IN token. It must be set to 1 by firmware to start packet transmission.</p> | 0x0 |

Table 575: USB_EPC2_REG (0x500400B0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_STALL | <p>Stall</p> <p>Setting this bit to 1 causes the chip to generate STALL handshakes under the following conditions:</p> <p>The transmit FIFO is enabled and an IN token is received.</p> <p>The receive FIFO is enabled and an OUT token is received.</p> <p>Setting this bit to 1 does not generate a STALL handshake in response to a SETUP token</p> | 0x0 |
| 6 | - | - | | 0x0 |
| 5 | R/W | USB_ISO | <p>Isochronous</p> <p>When this bit is set to 1, the endpoint is isochronous. This implies that no NAK is sent if the endpoint is not ready but enabled; i.e. If an IN token is received and no data is available in the FIFO to transmit, or if an OUT token is received and the FIFO is full since there is no USB handshake for isochronous transfers.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| 4 | R/W | USB_EP_EN | Endpoint Enable When this bit is set to 1, the EP[3:0] field is used in address comparison, together with the AD[6:0] field in the FAR register. When cleared to 0, the endpoint does not respond to any token on the USB bus. | 0x0 |
| 3:0 | R/W | USB_EP | Endpoint Address This 4-bit field holds the endpoint address. | 0x0 |

Table 576: USB_RXD1_REG (0x500400B4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R | USB_RXFD | Receive FIFO Data Byte The firmware should expect to read only the packet payload data. The PID and CRC16 are terminated by the receive state machine. In TEST mode this register allow read/write access via the core bus. | 0x0 |

Table 577: USB_RXS1_REG (0x500400B8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 14:8 | R | USB_RXCOUNT | it contains the number of bytes presently in the endpoint receive FIFO (range 0..64) | 0x0 |
| 7 | R | USB_RX_ERR | Receive Error When set to 1, this bit indicates a media error, such as bit-stuffing or CRC. If this bit is set to 1, the firmware must flush the respective FIFO. | 0x0 |
| 6 | R | USB_SETUP | Setup This bit indicates that the setup packet has been received. It is cleared when this register is read. | 0x0 |
| 5 | R | USB_TOGGLE_RX | Toggle The function of this bit differs depending on whether ISO (ISO in the EPCn register is set) or non-ISO operation (ISO is reset) is used. For non-ISO operation, a value of 0 indicates that the last successfully received packet had a DATA0 PID, while a value of 1 indicates that this packet had a DATA1 PID. For ISO operation, this bit reflects the LSB of the frame number (FNL0) after a packet was successfully received for this endpoint. This bit is reset to 0 by reading the RXSn register. | 0x0 |
| 4 | R | USB_RX_LAST | Receive Last This bit indicates that an ACK was sent upon completion of a successful receive operation. This bit is cleared to 0 when this register is read. | 0x0 |
| 3:0 | R | USB_RCOUNT | Receive Counter This 4-bit field contains the number of bytes | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | presently in the endpoint receive FIFO. If this number is greater than 15, a value of 15 is actually reported. | |

Table 578: USB_RXC1_REG (0x500400BC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6:5 | R/W | USB_RFWL | <p>Receive FIFO Warning Limit These bits specify how many more bytes can be received to the respective FIFO before an overrun condition occurs. If the number of empty bytes remaining in the FIFO is equal to or less than the selected warning limit, the RXWARN bit in the FWEV register is set to 1. RFWL[1:0] :</p> <ul style="list-style-type: none"> 00: RFWL disabled 01: Less than 5 bytes remaining in FIFO 10: Less than 9 bytes remaining in FIFO 11: Less than 17 bytes remaining in FIFO | 0x0 |
| 4 | - | - | | 0x0 |
| 3 | R/W | USB_FLUSH | <p>Flush FIFO Writing a 1 to this bit flushes all data from the corresponding receive FIFO, resets the endpoint to Idle state, and resets both the FIFO read and write pointers. If the MAC is currently using the FIFO to receive data, flushing is delayed until after receiving is completed.</p> | 0x0 |
| 2 | R/W | USB_IGN_SETUP | <p>Ignore SETUP Tokens When this bit is set to 1, the endpoint ignores any SETUP tokens directed to its configured address.</p> | 0x0 |
| 1 | - | - | | 0x0 |
| 0 | R/W | USB_RX_EN | <p>Receive Enable OUT packet cannot be received after every data packet is received, or when a STALL handshake is returned in response to an OUT token. This bit must be written with a 1 to re-enable data reception. SETUP packets can always be received. In the case of back-to-back SETUP packets (for a given endpoint) where a valid SETUP packet has been received with no other intervening non-SETUP tokens, the receive state machine discards the new SETUP packet and returns an ACK handshake. If, for any other reason, the receive state machine cannot accept the SETUP packet, no HANDSHAKE should be generated.</p> | 0x0 |

Table 579: USB_EPC3_REG (0x500400C0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|-------------|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_STALL | Stall | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| | | | <p>Setting this bit to 1 causes the chip to generate STALL handshakes under the following conditions:</p> <p>The transmit FIFO is enabled and an IN token is received.</p> <p>The receive FIFO is enabled and an OUT token is received.</p> <p>Setting this bit to 1 does not generate a STALL handshake in response to a SETUP token</p> | |
| 6 | - | - | | 0x0 |
| 5 | R/W | USB_ISO | <p>Isochronous</p> <p>When this bit is set to 1, the endpoint is isochronous. This implies that no NAK is sent if the endpoint is not ready but enabled; i.e. If an IN token is received and no data is available in the FIFO to transmit, or if an OUT token is received and the FIFO is full since there is no USB handshake for isochronous transfers.</p> | 0x0 |
| 4 | R/W | USB_EP_EN | <p>Endpoint Enable</p> <p>When this bit is set to 1, the EP[3:0] field is used in address comparison, together with the AD[6:0] field in the FAR register. When cleared to 0, the endpoint does not respond to any token on the USB bus.</p> | 0x0 |
| 3:0 | R/W | USB_EP | <p>Endpoint Address</p> <p>This 4-bit field holds the endpoint address.</p> | 0x0 |

Table 580: USB_TXD2_REG (0x500400C4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | W | USB_TXFD | <p>Transmit FIFO Data Byte</p> <p>The firmware is expected to write only the packet payload data. PID and CRC16 are inserted automatically in the transmit data stream.</p> <p>In TEST mode this register allow read/write access via the core bus.</p> | 0x0 |

Table 581: USB_TXS2_REG (0x500400C8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | USB_TX_URUN | <p>Transmit FIFO Underrun</p> <p>This bit is set to 1, if the transmit FIFO becomes empty during a transmission, and no new data is written to the FIFO. If so, the Media Access Controller (MAC) forces a bit stuff error followed by an EOP. This bit is cleared to 0, when this register is read.</p> | 0x0 |
| 6 | R | USB_ACK_STAT | <p>Acknowledge Status</p> <p>This bit is interpreted when TX_DONE is set. Its function differs depending on whether ISO (ISO in</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| | | | <p>the EPCx register is set) or non-ISO operation (ISO is reset) is used.</p> <p>For non-ISO operation, this bit indicates the acknowledge status (from the host) about the ACK for the previously sent packet. This bit itself is set to 1, when an ACK is received; otherwise, it is cleared to 0.</p> <p>For ISO operation, this bit is set if a frame number LSB match (see IGN_ISOMSK bit in the USB_TXCx_REG) occurs, and data was sent in response to an IN token. Otherwise, this bit is cleared to 0, the FIFO is flushed and TX_DONE is set.</p> <p>This bit is also cleared to 0, when this register is read.</p> | |
| 5 | R | USB_TX_DONE | <p>Transmission Done</p> <p>When set to 1, this bit indicates that the endpoint responded to a USB packet. Three conditions can cause this bit to be set:</p> <ul style="list-style-type: none"> A data packet completed transmission in response to an IN token with non-ISO operation. The endpoint sent a STALL handshake in response to an IN token A scheduled ISO frame was transmitted or discarded. <p>This bit is cleared to 0 when this register is read.</p> | 0x0 |
| 4:0 | R | USB_TCOUNT | <p>Transmission Count</p> <p>This 5-bit field holds the number of empty bytes available in the FIFO. If this number is greater than 31, a value of 31 is actually reported.</p> | 0x1F |

Table 582: USB_TXC2_REG (0x500400CC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_IGN_ISOMSK | <p>Ignore ISO Mask</p> <p>This bit has an effect only if the endpoint is set to be isochronous. If set to 1, this bit disables locking of specific frame numbers with the alternate function of the TOGGLE bit. Thus data is transmitted upon reception of the next IN token. If cleared to 0, data is only transmitted when FNL0 matches TOGGLE. This bit is cleared to 0 after reset.</p> | 0x0 |
| 6:5 | R/W | USB_TFWL | <p>Transmit FIFO Warning Limit</p> <p>These bits specify how many more bytes can be transmitted from the respective FIFO before an underrun condition occurs. If the number of bytes remaining in the FIFO is equal to or less than the selected warning limit, the TXWARN bit in the FWEV register is set. To avoid interrupts caused by setting this bit while the FIFO is being filled before a transmission begins, TXWARN is only set when transmission from the endpoint is enabled (TX_EN in the TXCn register is set).</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| | | | TFWL[1:0] : 00: TFWL disabled 01: Less than 5 bytes remaining in FIFO 10: Less than 9 bytes remaining in FIFO 11: Less than 17 bytes remaining in FIFO | |
| 4 | R/W | USB_RFF | Refill FIFO Setting the LAST bit to 1 automatically saves the Transmit Read Pointer (TXRP) to a buffer. When the RFF bit is set to 1, the buffered TXRP is reloaded into the TXRP. This allows the user to repeat the last transaction if no ACK was received from the host. If the MAC is currently using the FIFO to transmit, TXRP is reloaded only after the transmission is complete. After reload, this bit is cleared to 0 by hardware. | 0x0 |
| 3 | R/W | USB_FLUSH | Flush FIFO Writing a 1 to this bit flushes all data from the corresponding transmit FIFO, resets the endpoint to Idle state, and clears both the FIFO read and write pointers. If the MAC is currently using the FIFO to transmit, data is flushed after the transmission is complete. After data flushing, this bit is cleared to 0 by hardware. | 0x0 |
| 2 | R/W | USB_TOGGLE_TX | Toggle The function of this bit differs depending on whether ISO (ISO bit in the EPCn register is set to 1) or non-ISO operation (ISO bit is cleared to 0) is used. For non-ISO operation, it specifies the PID used when transmitting the packet. A value of 0 causes a DATA0 PID to be generated, while a value of 1 causes a DATA1 PID to be generated. For ISO operation, this bit and the LSB of the frame counter (FNL0) act as a mask for the TX_EN bit to allow pre-queuing of packets to specific frame numbers; i.e. transmission is enabled only if bit 0 in the FNL register is set to TOGGLE. If an IN token is not received while this condition is true, the contents of the FIFO are flushed with the next SOF. If the endpoint is set to ISO, data is always transferred with a DATA0 PID. | 0x0 |
| 1 | R/W | USB_LAST | Last Byte Setting this bit to 1 indicates that the entire packet has been written into the FIFO. This is used especially for streaming data to the FIFO while the actual transmission occurs. If the LAST bit is not set to 1 and the transmit FIFO becomes empty during a transmission, a stuff error followed by an EOP is forced on the bus. Zero length packets are indicated by setting this bit without writing any data to the FIFO. The transmit state machine transmits the payload data, CRC16 and the EOP signal before clearing this bit. | 0x0 |
| 0 | R/W | USB_TX_EN | Transmission Enable This bit enables data transmission from the FIFO. It is cleared to 0 by hardware after transmitting a single packet or after a STALL handshake in | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | response to an IN token. It must be set to 1 by firmware to start packet transmission. | |

Table 583: USB_EPC4_REG (0x500400D0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_STALL | Stall Setting this bit to 1 causes the chip to generate STALL handshakes under the following conditions: The transmit FIFO is enabled and an IN token is received. The receive FIFO is enabled and an OUT token is received. Setting this bit to 1 does not generate a STALL handshake in response to a SETUP token | 0x0 |
| 6 | - | - | | 0x0 |
| 5 | R/W | USB_ISO | Isochronous When this bit is set to 1, the endpoint is isochronous. This implies that no NAK is sent if the endpoint is not ready but enabled; i.e. If an IN token is received and no data is available in the FIFO to transmit, or if an OUT token is received and the FIFO is full since there is no USB handshake for isochronous transfers. | 0x0 |
| 4 | R/W | USB_EP_EN | Endpoint Enable When this bit is set to 1, the EP[3:0] field is used in address comparison, together with the AD[6:0] field in the FAR register. When cleared to 0, the endpoint does not respond to any token on the USB bus. | 0x0 |
| 3:0 | R/W | USB_EP | Endpoint Address This 4-bit field holds the endpoint address. | 0x0 |

Table 584: USB_RXD2_REG (0x500400D4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R | USB_RXFD | Receive FIFO Data Byte The firmware should expect to read only the packet payload data. The PID and CRC16 are terminated by the receive state machine. In TEST mode this register allow read/write access via the core bus. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 585: USB_RXS2_REG (0x500400D8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 14:8 | R | USB_RXCOUNT | it contains the number of bytes presently in the endpoint receive FIFO (range 0..64) | 0x0 |
| 7 | R | USB_RX_ERR | Receive Error When set to 1, this bit indicates a media error, such as bit-stuffing or CRC. If this bit is set to 1, the firmware must flush the respective FIFO. | 0x0 |
| 6 | R | USB_SETUP | Setup This bit indicates that the setup packet has been received. It is cleared when this register is read. | 0x0 |
| 5 | R | USB_TOGGLE_RX | Toggle The function of this bit differs depending on whether ISO (ISO in the EPCn register is set) or non-ISO operation (ISO is reset) is used. For non-ISO operation, a value of 0 indicates that the last successfully received packet had a DATA0 PID, while a value of 1 indicates that this packet had a DATA1 PID. For ISO operation, this bit reflects the LSB of the frame number (FNL0) after a packet was successfully received for this endpoint. This bit is reset to 0 by reading the RXSn register. | 0x0 |
| 4 | R | USB_RX_LAST | Receive Last This bit indicates that an ACK was sent upon completion of a successful receive operation. This bit is cleared to 0 when this register is read. | 0x0 |
| 3:0 | R | USB_RCOUNT | Receive Counter This 4-bit field contains the number of bytes presently in the endpoint receive FIFO. If this number is greater than 15, a value of 15 is actually reported. | 0x0 |

Table 586: USB_RXC2_REG (0x500400DC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6:5 | R/W | USB_RFWL | Receive FIFO Warning Limit These bits specify how many more bytes can be received to the respective FIFO before an overrun condition occurs. If the number of empty bytes remaining in the FIFO is equal to or less than the selected warning limit, the RXWARN bit in the FWEV register is set to 1.RFWL[1:0] : 00: RFWL disabled 01: Less than 5 bytes remaining in FIFO 10: Less than 9 bytes remaining in FIFO 11: Less than 17 bytes remaining in FIFO | 0x0 |
| 4 | - | - | | 0x0 |
| 3 | R/W | USB_FLUSH | Flush FIFO Writing a 1 to this bit flushes all data from the corresponding receive FIFO, resets the endpoint to | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| | | | Idle state, and resets both the FIFO read and write pointers. If the MAC is currently using the FIFO to receive data, flushing is delayed until after receiving is completed. | |
| 2 | R/W | USB_IGN_SETUP | Ignore SETUP Tokens When this bit is set to 1, the endpoint ignores any SETUP tokens directed to its configured address. | 0x0 |
| 1 | - | - | | 0x0 |
| 0 | R/W | USB_RX_EN | Receive Enable OUT packet cannot be received after every data packet is received, or when a STALL handshake is returned in response to an OUT token. This bit must be written with a 1 to re-enable data reception. SETUP packets can always be received. In the case of back-to-back SETUP packets (for a given endpoint) where a valid SETUP packet has been received with no other intervening non-SETUP tokens, the receive state machine discards the new SETUP packet and returns an ACK handshake. If, for any other reason, the receive state machine cannot accept the SETUP packet, no HANDSHAKE should be generated. | 0x0 |

Table 587: USB_EPC5_REG (0x500400E0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_STALL | Stall Setting this bit to 1 causes the chip to generate STALL handshakes under the following conditions: The transmit FIFO is enabled and an IN token is received. The receive FIFO is enabled and an OUT token is received. Setting this bit to 1 does not generate a STALL handshake in response to a SETUP token | 0x0 |
| 6 | - | - | | 0x0 |
| 5 | R/W | USB_ISO | Isochronous When this bit is set to 1, the endpoint is isochronous. This implies that no NAK is sent if the endpoint is not ready but enabled; i.e. If an IN token is received and no data is available in the FIFO to transmit, or if an OUT token is received and the FIFO is full since there is no USB handshake for isochronous transfers. | 0x0 |
| 4 | R/W | USB_EP_EN | Endpoint Enable When this bit is set to 1, the EP[3:0] field is used in address comparison, together with the AD[6:0] field in the FAR register. When cleared to 0, the endpoint does not respond to any token on the USB bus. | 0x0 |
| 3:0 | R/W | USB_EP | Endpoint Address This 4-bit field holds the endpoint address. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 588: USB_TXD3_REG (0x500400E4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | W | USB_TXF D | <p>Transmit FIFO Data Byte</p> <p>The firmware is expected to write only the packet payload data. PID and CRC16 are inserted automatically in the transmit data stream.</p> <p>In TEST mode this register allow read/write access via the core bus.</p> | 0x0 |

Table 589: USB_TXS3_REG (0x500400E8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R | USB_TX_URUN | <p>Transmit FIFO Underrun</p> <p>This bit is set to 1, if the transmit FIFO becomes empty during a transmission, and no new data is written to the FIFO. If so, the Media Access Controller (MAC) forces a bit stuff error followed by an EOP. This bit is cleared to 0, when this register is read.</p> | 0x0 |
| 6 | R | USB_ACK_STAT | <p>Acknowledge Status</p> <p>This bit is interpreted when TX_DONE is set. Its function differs depending on whether ISO (ISO in the EPCx register is set) or non-ISO operation (ISO is reset) is used.</p> <p>For non-ISO operation, this bit indicates the acknowledge status (from the host) about the ACK for the previously sent packet. This bit itself is set to 1, when an ACK is received; otherwise, it is cleared to 0.</p> <p>For ISO operation, this bit is set if a frame number LSB match (see IGN_ISOMSK bit in the USB_TXCx_REG) occurs, and data was sent in response to an IN token. Otherwise, this bit is cleared to 0, the FIFO is flushed and TX_DONE is set.</p> <p>This bit is also cleared to 0, when this register is read.</p> | 0x0 |
| 5 | R | USB_TX_DONE | <p>Transmission Done</p> <p>When set to 1, this bit indicates that the endpoint responded to a USB packet. Three conditions can cause this bit to be set:</p> <p>A data packet completed transmission in response to an IN token with non-ISO operation.</p> <p>The endpoint sent a STALL handshake in response to an IN token</p> <p>A scheduled ISO frame was transmitted or discarded.</p> <p>This bit is cleared to 0 when this register is read.</p> | 0x0 |
| 4:0 | R | USB_TCOUNT | <p>Transmission Count</p> <p>This 5-bit field holds the number of empty bytes</p> | 0x1F |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | available in the FIFO. If this number is greater than 31, a value of 31 is actually reported. | |

Table 590: USB_TXC3_REG (0x500400EC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_IGN_ISOMSK | <p>Ignore ISO Mask</p> <p>This bit has an effect only if the endpoint is set to be isochronous. If set to 1, this bit disables locking of specific frame numbers with the alternate function of the TOGGLE bit. Thus data is transmitted upon reception of the next IN token. If cleared to 0, data is only transmitted when FNLO matches TOGGLE. This bit is cleared to 0 after reset.</p> | 0x0 |
| 6:5 | R/W | USB_TFWL | <p>Transmit FIFO Warning Limit</p> <p>These bits specify how many more bytes can be transmitted from the respective FIFO before an underrun condition occurs. If the number of bytes remaining in the FIFO is equal to or less than the selected warning limit, the TXWARN bit in the FWEV register is set. To avoid interrupts caused by setting this bit while the FIFO is being filled before a transmission begins, TXWARN is only set when transmission from the endpoint is enabled (TX_ENn in the TXCn register is set).</p> <p>TFWL[1:0] :</p> <ul style="list-style-type: none"> 00: TFWL disabled 01: Less than 5 bytes remaining in FIFO 10: Less than 9 bytes remaining in FIFO 11: Less than 17 bytes remaining in FIFO | 0x0 |
| 4 | R/W | USB_RFF | <p>Refill FIFO</p> <p>Setting the LAST bit to 1 automatically saves the Transmit Read Pointer (TXRP) to a buffer. When the RFF bit is set to 1, the buffered TXRP is reloaded into the TXRP. This allows the user to repeat the last transaction if no ACK was received from the host. If the MAC is currently using the FIFO to transmit, TXRP is reloaded only after the transmission is complete. After reload, this bit is cleared to 0 by hardware.</p> | 0x0 |
| 3 | R/W | USB_FLUSH | <p>Flush FIFO</p> <p>Writing a 1 to this bit flushes all data from the corresponding transmit FIFO, resets the endpoint to Idle state, and clears both the FIFO read and write pointers. If the MAC is currently using the FIFO to transmit, data is flushed after the transmission is complete. After data flushing, this bit is cleared to 0 by hardware.</p> | 0x0 |
| 2 | R/W | USB_TOGGLE_TX | <p>Toggle</p> <p>The function of this bit differs depending on whether ISO (ISO bit in the EPCn register is set to 1) or non-ISO operation (ISO bit is cleared to 0) is used.</p> <p>For non-ISO operation, it specifies the PID used</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | <p>when transmitting the packet. A value of 0 causes a DATA0 PID to be generated, while a value of 1 causes a DATA1 PID to be generated.</p> <p>For ISO operation, this bit and the LSB of the frame counter (FNL0) act as a mask for the TX_EN bit to allow pre-queuing of packets to specific frame numbers; i.e. transmission is enabled only if bit 0 in the FNL register is set to TOGGLE. If an IN token is not received while this condition is true, the contents of the FIFO are flushed with the next SOF. If the endpoint is set to ISO, data is always transferred with a DATA0 PID.</p> | |
| 1 | R/W | USB_LAST | <p>Last Byte</p> <p>Setting this bit to 1 indicates that the entire packet has been written into the FIFO. This is used especially for streaming data to the FIFO while the actual transmission occurs. If the LAST bit is not set to 1 and the transmit FIFO becomes empty during a transmission, a stuff error followed by an EOP is forced on the bus. Zero length packets are indicated by setting this bit without writing any data to the FIFO.</p> <p>The transmit state machine transmits the payload data, CRC16 and the EOP signal before clearing this bit.</p> | 0x0 |
| 0 | R/W | USB_TX_EN | <p>Transmission Enable</p> <p>This bit enables data transmission from the FIFO. It is cleared to 0 by hardware after transmitting a single packet or after a STALL handshake in response to an IN token. It must be set to 1 by firmware to start packet transmission.</p> | 0x0 |

Table 591: USB_EPC6_REG (0x500400F0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7 | R/W | USB_STALL | <p>Stall</p> <p>Setting this bit to 1 causes the chip to generate STALL handshakes under the following conditions:</p> <p>The transmit FIFO is enabled and an IN token is received.</p> <p>The receive FIFO is enabled and an OUT token is received.</p> <p>Setting this bit to 1 does not generate a STALL handshake in response to a SETUP token</p> | 0x0 |
| 6 | - | - | | 0x0 |
| 5 | R/W | USB_ISO | <p>Isochronous</p> <p>When this bit is set to 1, the endpoint is isochronous. This implies that no NAK is sent if the endpoint is not ready but enabled; i.e. If an IN token is received and no data is available in the FIFO to transmit, or if an OUT token is received and the FIFO is full since there is no USB handshake for isochronous transfers.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| 4 | R/W | USB_EP_EN | Endpoint Enable When this bit is set to 1, the EP[3:0] field is used in address comparison, together with the AD[6:0] field in the FAR register. When cleared to 0, the endpoint does not respond to any token on the USB bus. | 0x0 |
| 3:0 | R/W | USB_EP | Endpoint Address This 4-bit field holds the endpoint address. | 0x0 |

Table 592: USB_RXD3_REG (0x500400F4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:8 | - | - | | 0x0 |
| 7:0 | R | USB_RXFD | Receive FIFO Data Byte The firmware should expect to read only the packet payload data. The PID and CRC16 are terminated by the receive state machine. In TEST mode this register allow read/write access via the core bus. | 0x0 |

Table 593: USB_RXS3_REG (0x500400F8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 14:8 | R | USB_RXCOUNT | it contains the number of bytes presently in the endpoint receive FIFO (range 0..64) | 0x0 |
| 7 | R | USB_RX_ERR | Receive Error When set to 1, this bit indicates a media error, such as bit-stuffing or CRC. If this bit is set to 1, the firmware must flush the respective FIFO. | 0x0 |
| 6 | R | USB_SETUP | Setup This bit indicates that the setup packet has been received. It is cleared when this register is read. | 0x0 |
| 5 | R | USB_TOGGLE_RX | Toggle The function of this bit differs depending on whether ISO (ISO in the EPCn register is set) or non-ISO operation (ISO is reset) is used. For non-ISO operation, a value of 0 indicates that the last successfully received packet had a DATA0 PID, while a value of 1 indicates that this packet had a DATA1 PID. For ISO operation, this bit reflects the LSB of the frame number (FNL0) after a packet was successfully received for this endpoint. This bit is reset to 0 by reading the RXSn register. | 0x0 |
| 4 | R | USB_RX_LAST | Receive Last This bit indicates that an ACK was sent upon completion of a successful receive operation. This bit is cleared to 0 when this register is read. | 0x0 |
| 3:0 | R | USB_RCOUNT | Receive Counter This 4-bit field contains the number of bytes | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | presently in the endpoint receive FIFO. If this number is greater than 15, a value of 15 is actually reported. | |

Table 594: USB_RXC3_REG (0x500400FC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:7 | - | - | | 0x0 |
| 6:5 | R/W | USB_RFWL | <p>Receive FIFO Warning Limit These bits specify how many more bytes can be received to the respective FIFO before an overrun condition occurs. If the number of empty bytes remaining in the FIFO is equal to or less than the selected warning limit, the RXWARN bit in the FWEV register is set to 1. RFWL[1:0] :</p> <ul style="list-style-type: none"> 00: RFWL disabled 01: Less than 5 bytes remaining in FIFO 10: Less than 9 bytes remaining in FIFO 11: Less than 17 bytes remaining in FIFO | 0x0 |
| 4 | - | - | | 0x0 |
| 3 | R/W | USB_FLUSH | <p>Flush FIFO Writing a 1 to this bit flushes all data from the corresponding receive FIFO, resets the endpoint to Idle state, and resets both the FIFO read and write pointers. If the MAC is currently using the FIFO to receive data, flushing is delayed until after receiving is completed.</p> | 0x0 |
| 2 | R/W | USB_IGN_SETUP | <p>Ignore SETUP Tokens When this bit is set to 1, the endpoint ignores any SETUP tokens directed to its configured address.</p> | 0x0 |
| 1 | - | - | | 0x0 |
| 0 | R/W | USB_RX_EN | <p>Receive Enable OUT packet cannot be received after every data packet is received, or when a STALL handshake is returned in response to an OUT token. This bit must be written with a 1 to re-enable data reception. SETUP packets can always be received. In the case of back-to-back SETUP packets (for a given endpoint) where a valid SETUP packet has been received with no other intervening non-SETUP tokens, the receive state machine discards the new SETUP packet and returns an ACK handshake. If, for any other reason, the receive state machine cannot accept the SETUP packet, no HANDSHAKE should be generated.</p> | 0x0 |

Table 595: USB_DMA_CTRL_REG (0x500401A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:7 | - | - | | 0x0 |
| 6 | R/W | USB_DMA_EN | 0 = USB DMA control off. (Normal operation) | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| | | | 1 = USB_DMA on. DMA channels 0 and 1 are connected by USB Endpoint according bits USB_DMA_TX and USB_DMA_RX | |
| 5:3 | R/W | USB_DMA_TX | 000 = DMA channels 1 is connected Tx USB Endpoint 1 001 = DMA channels 1 is connected Tx USB Endpoint 3 010 = DMA channels 1 is connected Tx USB Endpoint 5 100, 1xx = Reserved | 0x0 |
| 2:0 | R/W | USB_DMA_RX | 000 = DMA channels 0 is connected Rx USB Endpoint 2 001 = DMA channels 0 is connected Rx USB Endpoint 4 010 = DMA channels 0 is connected Rx USB Endpoint 6 100, 1xx = Reserved | 0x0 |

Table 596: USB_CHARGER_CTRL_REG (0x500401A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 31:6 | - | - | | 0x0 |
| 5 | R/W | IDM_SINK_ON | 0 = Disable 1 = Enable the Idm_sink to USBm | 0x0 |
| 4 | R/W | IDP_SINK_ON | 0 = Disable 1 = Enable the Idp_sink to USBp | 0x0 |
| 3 | R/W | VDM_SRC_ON | 0 = Disable 1 = Enable Vdm_src to USBm and USB_DCP_DET status bit. | 0x0 |
| 2 | R/W | VDP_SRC_ON | 0 = Disable 1 = Enable the Vdp_src to USB_CHG_DET status bit. | 0x0 |
| 1 | R/W | IDP_SRC_ON | 0 = Disable 1 = Enable the Idp_src and Rdm_dwn. | 0x0 |
| 0 | R/W | USB_CHARGE_ON | 0 = Disable USB charger detect circuit. 1 = Enable USB charger detect circuit. | 0x0 |

Table 597: USB_CHARGER_STAT_REG (0x500401AC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|------------------------------------|-------|
| 31:6 | R | - | | 0x0 |
| 5 | R | USB_DM_VAL2 | 0 = USBm < 2.3V 1 = USBm > 2.5V | 0x0 |
| 4 | R | USB_DP_VAL2 | 0: USBp < 2.3V 1: USBp > 2.5V | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| 3 | R | USB_DM_VAL | 0 = USBm < 0.8V 1 = USBm > 1.5V (PS2 or Proprietary Charger) | 0x0 |
| 2 | R | USB_DP_VAL | 0 = USBp < 0.8V 1 = USBp > 1.5V | 0x0 |
| 1 | R | USB_CHG_DET | 0 = Standard downstream or nothing connected. 1 = Charging Downstream Port (CDP) or Dedicated Charging. | 0x0 |
| 0 | R | USB_DCP_DET | 0 = Charging downstream port is detected. 1 = Dedicated charger is detected. Control bit VDM_SRC_ON must be set to validate this status bit. Note: This register shows the actual status. | 0x0 |

42.18 Silicon Version Registers

Table 598: Register map Version

| Address | Register | Description |
|------------|-------------------|----------------------------------|
| 0x50040200 | CHIP_ID1_REG | Chip identification register 1. |
| 0x50040204 | CHIP_ID2_REG | Chip identification register 2. |
| 0x50040208 | CHIP_ID3_REG | Chip identification register 3. |
| 0x5004020C | CHIP_ID4_REG | Chip identification register 4. |
| 0x50040210 | CHIP_SWC_REG | Software compatibility register. |
| 0x50040214 | CHIP_REVISION_REG | Chip revision register. |
| 0x500402F8 | CHIP_TEST1_REG | Chip test register 1. |
| 0x500402FC | CHIP_TEST2_REG | Chip test register 2. |

Table 599: CHIP_ID1_REG (0x50040200)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:8 | R | - | | 0x0 |
| 7:0 | R | CHIP_ID1 | First character of device type "2522" in ASCII. | 0x32 |

Table 600: CHIP_ID2_REG (0x50040204)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:8 | R | - | | 0x0 |
| 7:0 | R | CHIP_ID2 | Second character of device type "2522" in ASCII. | 0x35 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 601: CHIP_ID3_REG (0x50040208)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:8 | R | - | | 0x0 |
| 7:0 | R | CHIP_ID3 | Third character of device type "2522" in ASCII. | 0x32 |

Table 602: CHIP_ID4_REG (0x5004020C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:8 | R | - | | 0x0 |
| 7:0 | R | CHIP_ID4 | Fourth character of device type "2522" in ASCII. | 0x32 |

Table 603: CHIP_SWC_REG (0x50040210)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 31:4 | R | - | | 0x0 |
| 3:0 | R | CHIP_SWC | SoftWare Compatibility code. Integer (default = 0) which is incremented if a silicon change has impact on the CPU Firmware. Can be used by software developers to write silicon revision dependent code. | 0x0 |

Table 604: CHIP_REVISION_REG (0x50040214)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:8 | R | - | | 0x0 |
| 7:0 | R | CHIP_REVISION | Chip version, corresponds with type number in ASCII. 0x41 = 'A', 0x42 = 'B' | 0x41 |

Table 605: CHIP_TEST1_REG (0x500402F8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|--|-------|
| 31:8 | R | - | | 0x0 |
| 7:0 | R | CHIP_LAYOUT_REVISION | Chip layout revision, corresponds with type number in ASCII. 0x41 = 'A', 0x42 = 'B' | 0x42 |

Table 606: CHIP_TEST2_REG (0x500402FC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|--------------------------|-------|
| 31:4 | R | - | | 0x0 |
| 3:0 | R | CHIP_METAL_OPTION | Chip metal option value. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

42.19 Wake-Up Controller Registers

Table 607: Register map WakeUp

| Address | Register | Description |
|------------|----------------------|---|
| 0x50000100 | WKUP_CTRL_REG | Control register for the wakeup counter |
| 0x50000108 | WKUP_RESET_IRQ_REG | Reset wakeup interrupt |
| 0x50000114 | WKUP_SELECT_P0_REG | select which inputs from P0 port can trigger wkup counter |
| 0x50000118 | WKUP_SELECT_P1_REG | select which inputs from P1 port can trigger wkup counter |
| 0x50000128 | WKUP_POL_P0_REG | select the sensitivity polarity for each P0 input |
| 0x5000012C | WKUP_POL_P1_REG | select the sensitivity polarity for each P1 input |
| 0x5000013C | WKUP_STATUS_P0_REG | Event status register for P0 |
| 0x50000140 | WKUP_STATUS_P1_REG | Event status register for P1 |
| 0x50000148 | WKUP_CLEAR_P0_REG | Clear event register for P0 |
| 0x5000014C | WKUP_CLEAR_P1_REG | Clear event register for P1 |
| 0x50000154 | WKUP_SEL_GPIO_P0_REG | select which inputs from P0 port can trigger interrupt |
| 0x50000158 | WKUP_SEL_GPIO_P1_REG | select which inputs from P1 port can trigger interrupt |

Table 608: WKUP_CTRL_REG (0x50000100)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------|---|-------|
| 15:8 | - | - | | 0x0 |
| 7 | R/W | WKUP_ENABLE IRQ | 0: no interrupt will be enabled 1: if you have an event an IRQ will be generated | 0x0 |
| 6 | R/W | WKUP_SFT_KEYHIT | 0 = no effect 1 = emulate key hit. First make this bit 0 before any new key hit can be sensed. | 0x0 |
| 5:0 | R/W | WKUP_DEB_VALUE | Wakeup debounce time. If set to 0, no debouncing will be done. Debounce time: N*1 ms. N =1..63 | 0x0 |

Table 609: WKUP_RESET_IRQ_REG (0x50000108)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 15:0 | W | WKUP_IRQ_RST | writing any value to this register will reset the interrupt. reading always returns 0. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 610: WKUP_SELECT_P0_REG (0x50000114)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:0 | R/W | WKUP_SELECT_P0 | 0: input P0_xx is not enabled for wakeup event 1: input P0_xx is enabled for wakeup event | 0x0 |

Table 611: WKUP_SELECT_P1_REG (0x50000118)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 22:0 | R/W | WKUP_SELECT_P1 | 0: input P1_xx is not enabled for wakeup event 1: input P1_xx is enabled for wakeup event | 0x0 |

Table 612: WKUP_POL_P0_REG (0x50000128)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:0 | R/W | WKUP_POL_P0 | 0: enabled input P0_xx will give an event if that input goes high 1: enabled input P0_xx will give an event if that input goes low | 0x0 |

Table 613: WKUP_POL_P1_REG (0x5000012C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 22:0 | R/W | WKUP_POL_P1 | 0: enabled input P1_xx will give an event if that input goes high 1: enabled input P1_xx will give an event if that input goes low | 0x0 |

Table 614: WKUP_STATUS_P0_REG (0x5000013C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:0 | R | WKUP_STAT_P0 | Contains the latched value of any toggle of the GPIOs Port P0. WKUP_STAT_P0[0] -> P0_00. | 0x0 |

Table 615: WKUP_STATUS_P1_REG (0x50000140)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 22:0 | R | WKUP_STAT_P1 | Contains the latched value of any toggle of the GPIOs Port P1. WKUP_STAT_P1[0] -> P1_00. | 0x0 |

Table 616: WKUP_CLEAR_P0_REG (0x50000148)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 31:0 | W | WKUP_CLEAR_P0 | Clear latched value of the GPIOs P0 when corresponding bit is 1 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 617: WKUP_CLEAR_P1_REG (0x5000014C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 22:0 | W | WKUP_CLEAR_P1 | Clear latched value of the GPIOs P1 when corresponding bit is 1 | 0x0 |

Table 618: WKUP_SEL_GPIO_P0_REG (0x50000154)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|--|-------|
| 31:0 | R/W | WKUP_SEL_GPIO_P0 | 0: input P0_xx is not enabled for GPIO interrupt 1: input P0_xx is enabled for GPIO interrupt | 0x0 |

Table 619: WKUP_SEL_GPIO_P1_REG (0x50000158)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|--|-------|
| 22:0 | R/W | WKUP_SEL_GPIO_P1 | 0: input P1_xx is not enabled for GPIO interrupt 1: input P1_xx is enabled for GPIO interrupt | 0x0 |

42.20 Watchdog Controller Registers

Table 620: Register map WDOG

| Address | Register | Description |
|------------|-------------------|----------------------------|
| 0x50000700 | WATCHDOG_REG | Watchdog timer register. |
| 0x50000704 | WATCHDOG_CTRL_REG | Watchdog control register. |

Table 621: WATCHDOG_REG (0x50000700)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|--|--------|
| 31:14 | R0/W | WDOG_WEN | Bit [31:14] = 0 = Write enable for Watchdog timer else Write disable. This filter prevents unintentional presetting the watchdog with a SW run-away. | 0x0 |
| 13 | R/W | WDOG_VAL_NEG | 0 = Watchdog timer value is positive. 1 = Watchdog timer value is negative. | 0x0 |
| 12:0 | R/W | WDOG_VAL | <u>Write:</u> Watchdog timer reload value. Note that all bits [31-14] must be 0 to reload this register. <u>Read:</u> Actual Watchdog timer value. Decremented by 1 every ~10 msec (RC32K) or ~29 msec (RCX), i.e. the Watchdog timer clock tick. Bit 13 indicates a negative counter value. 2, 1, 0, 3FFF ₁₆ , 3FFE 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 = 3FF0 ₁₆ -> WDOG reset -> reload 1FFF ₁₆ If WATCHDOG_CTRL_REG[NMI_RST] = 1 then | 0x1FFF |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>if WDOG_VAL <= 0 -> WDOG reset -> reload 1FFF₁₆</p> <p>Note 1: The programmed value WDOG_VAL is updated in the (independent) Watchdog timer at the 2nd next RC32K or RCX clock tick.</p> <p>Note 2: Select RC32K or RCX with CLK_RCX_REG[RCX_ENABLE]. The RC32K is selected by default.</p> <p>Note 3: If WATCHDOG_CTRL_REG[NMI_RST] = 0, the time between the NMI generation and the WDOG reset generation is 15 Watchdog timer clock ticks.</p> | |

Table 622: WATCHDOG_CTRL_REG (0x50000704)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:4 | R | - | | 0x0 |
| 3 | R | WRITE_BUSY | <p>0 = A new WATCHDOG_REG[WDOG_VAL] can be written.</p> <p>1 = No new WATCHDOG_REG[WDOG_VAL] can be written.</p> <p>Note: It takes some time before the programmed WDOG_VAL is updated in the (independent) Watchdog timer. During this time it is not possible to write a new value to WATCHDOG_REG[WDOG_VAL].</p> | 0x0 |
| 2 | R/W | WDOG_FREEZE_EN | <p>0 = Watchdog timer can not be frozen when NMI_RST=0.</p> <p>1 = Watchdog timer can be frozen/resumed using SET_FREEZE_REG[FRZ_WDOG]/RESET_FREEZE_REG[FRZ_WDOG] when NMI_RST=0.</p> | 0x1 |
| 1 | R/W | - | | 0x1 |
| 0 | R/W | NMI_RST | <p>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].</p> <p>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 SW and only be reset with a WDOG (SYS) reset or SW reset.</p> <p>The watchdog is always frozen when the Cortex-M33 is halted in DEBUG State.</p> | 0x0 |

42.21 General Purpose / ΣΔ ADC Registers

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 623: Register map GPADC

| Address | Register | Description |
|------------|------------------------|--|
| 0x50020800 | SDADC_CTRL_REG | Sigma Delta ADC Control Register |
| 0x50020808 | SDADC_TEST_REG | Sigma Delta ADC Test Register |
| 0x5002080C | SDADC_GAIN_CORR_REG | Sigma Delta ADC Gain Correction Register |
| 0x50020810 | SDADC_OFFSETS_CORR_REG | Sigma Delta ADC Offset Correction Register |
| 0x50020814 | SDADC_CLEAR_INT_REG | Sigma Delta ADC Clear Interrupt Register |
| 0x50020818 | SDADC_RESULT_REG | Sigma Delta ADC Result Register |
| 0x50030900 | GP_ADC_CTRL_REG | General Purpose ADC Control Register |
| 0x50030904 | GP_ADC_CTRL2_REG | General Purpose ADC Second Control Register |
| 0x50030908 | GP_ADC_CTRL3_REG | General Purpose ADC Third Control Register |
| 0x5003090C | GP_ADC_OFFSETP_REG | General Purpose ADC Positive Offset Register |
| 0x50030910 | GP_ADC_OFFSETN_REG | General Purpose ADC Negative Offset Register |
| 0x50030914 | GP_ADC_CLEAR_INT_REG | General Purpose ADC Clear Interrupt Register |
| 0x50030918 | GP_ADC_RESULT_REG | General Purpose ADC Result Register |

Table 624: SDADC_CTRL_REG (0x50020800)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|---|-------|
| 17 | R/W | SDADC_DMA_EN | 0: DMA functionality disabled 1: DMA functionality enabled | 0x0 |
| 16 | R/W | SDADC_MINT | 0: Disable (mask) SDADC_ADC_INT. 1: Enable SDADC_ADC_INT to ICU. | 0x0 |
| 15 | R | SDADC_INT | 1: AD conversion ready and has generated an interrupt. Must be cleared by writing any value to SDADC_CLEAR_INT_REG. | 0x0 |
| 14 | R | SDADC_LDO_OK | 1: Internal LDO is ready for use | 0x0 |
| 13 | R/W | SDADC_VREF_SEL | 0: Internal bandgap reference. 1: External reference. | 0x0 |
| 12 | R/W | SDADC_CONT | 0: Manual ADC mode, a single result will be generated after setting the SDADC_START bit. 1: Continuous ADC mode, new ADC results will be constantly stored in SDADC_RESULT_REG. Still SDADC_START has to be set to start the execution. Wait for SDADC_START to become zero after clearing the SDADC_CONT bit to stop the continuous mode. | 0x0 |
| 11:10 | R/W | SDADC_OSR | Oversample Rate 0: 128x 1: 256x 2: 512x | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| | | | 3: 1024x | |
| 9 | R/W | SDADC_SE | 0: Differential mode 1: Single ended mode (Input selection negative side is ignored) | 0x0 |
| 8:6 | R/W | SDADC_INN_SEL | Input selection of negative side. 0: ADC0 / P1[09] 1: ADC1 / P0[25] 2: ADC2 / P0[08] 3: ADC3 / P0[09] 4: ADC4 / P1[14] 5: ADC5 / P1[20] 6: ADC6 / P1[21] 7: ADC7 / P1[22] | 0x0 |
| 5:2 | R/W | SDADC_INP_SEL | Input selection of positive side. 0: ADC0 / P1[09] 1: ADC1 / P0[25] 2: ADC2 / P0[08] 3: ADC3 / P0[09] 4: ADC4 / P1[14] 5: ADC5 / P1[20] 6: ADC6 / P1[21] 7: ADC7 / P1[22] 8: VBAT (via 4x attenuator, INN connected to ground) | 0x0 |
| 1 | R/W | SDADC_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 SDADC_INT bit will be set. It is not allowed to write this bit while it is not (yet) zero. | 0x0 |
| 0 | R/W | SDADC_EN | 0: LDO is off and ADC is disabled. 1: LDO, bias currents and modulator are enabled. | 0x0 |

Table 625: SDADC_TEST_REG (0x50020808)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------|--|-------|
| 15:11 | - | - | | 0xF |
| 10 | - | - | | 0x0 |
| 9:8 | - | - | | 0x1 |
| 7:6 | R/W | SDADC_CLK_FREQ | 0: 250 kHz 1: 500 kHz 2: 1 MHz (default) 3: 2 MHz | 0x2 |
| 5 | - | - | | 0x0 |
| 4 | - | - | | 0x0 |
| 3 | - | - | | 0x0 |
| 2 | - | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------|-------|
| 1 | - | - | | 0x0 |
| 0 | - | - | | 0x0 |

Table 626: SDADC_GAIN_CORR_REG (0x5002080C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|-------------|-------|
| 9:0 | R/W | SDADC_GAIN_CORR | Gain adjust | 0x0 |

Table 627: SDADC_OFFSETS_CORR_REG (0x50020810)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|---------------|-------|
| 9:0 | R/W | SDADC_OFFSETS_CO RR | Offset adjust | 0x0 |

Table 628: SDADC_CLEAR_INT_REG (0x50020814)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 15:0 | R0/W | SDADC_CLR_INT | Writing any value to this register will clear the ADC_INT interrupt. Reading returns 0. | 0x0 |

Table 629: SDADC_RESULT_REG (0x50020818)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 15:0 | R | SDADC_VAL | Returns up to 16 bits linear value of the last AD conversion. The effective resolution depends on the OSR used. | 0x0 |

Table 630: GP_ADC_CTRL_REG (0x50030900)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------|--|-------|
| 18 | R/W | GP_ADC_DIFF_TEMP_EN | 1: Enable the on-chip temperature sensors | 0x0 |
| 17:16 | R/W | GP_ADC_DIFF_TEMP_SEL | 0= Gnd, 1 =sensor near radio, 2 =sensor near charger, 3 =sensor near bandgap with sensors disabled (GP_ADC_DIFF_TEMP_EN = 0) :0 = GND 1 = Z, 2= V(ntc) from charger, 3 = V(temp) from charger | 0x0 |
| 15 | R/W | GP_ADC_LDO_ZERO | 1: Samples and disconnects VREF, should be refreshed frequently. Note that the LDO consumes power when bit is set. | 0x0 |
| 14 | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 13 | 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 |
| 12:8 | R/W | GP_ADC_SEL | ADC input selection. If GP_ADC_SE = 1 (single ended mode): 0: GP ADC0 (P1_09) 1: GP ADC1 (P0_25) 2: GP ADC2 (P0_08) 3: GP ADC3 (P0_09) 4: VDD (internal) 5: V30 (GP_ADC_ATTN3X scaler automatically selected) 6: V30 (GP_ADC_ATTN3X scaler automatically selected) 7: Reserved 8: VBAT1 (5V to 1.2V scaler selected) 9: VSSA (analog ground) 10-15: Reserved 16: GP ADC4 (P1_13) 17: GP ADC5 (P1_12) 18: GP ADC6 (P1_18) 19: GP ADC7 (P1_19) 20: Diff Temperature sensor, See GP_ADC_DIFF_TEMP_SEL 21-31: Reserved If GP_ADC_SE = 0 (differential mode): 0: P1[09] vs P0[25] All other combinations are P0[08] vs P0[09]. | 0x0 |
| 7 | R/W | GP_ADC_MUTE | 0: Normal operation 1: Mute ADC input. Takes sample at mid-scale (to determine 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 | 1: AD conversion ready and has generated an interrupt. Must be cleared by writing any value to GP_ADC_CLEAR_INT_REG. | 0x0 |
| 3 | R/W | GP_ADC_CLK_SEL | 0: Internal high-speed ADC clock used (recommended). 1: Digital clock used (ADC_CLK). | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| 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 631: GP_ADC_CTRL2_REG (0x50030904)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| 15:12 | R/W | GP_ADC_STORE_DELAY | 0: Data is stored after handshake synchronisation 1: Data is stored two ADC_CLK cycles after internal start trigger 15: Data is stored sixteen ADC_CLK cycles after internal start trigger | 0x0 |
| 11:8 | R/W | GP_ADC_SMPL_TIME | 0: The sample time (switch is closed) is one ADC_CLK cycle 1: The sample time is 1*32 ADC_CLK cycles 2: The sample time is 2*32 ADC_CLK cycles 15: The sample time is 15*32 ADC_CLK cycles | 0x0 |
| 7:5 | 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 |
| 4 | R/W | - | | 0x0 |
| 3 | R/W | GP_ADC_DMA_EN | 0: DMA functionality disabled 1: DMA functionality enabled | 0x0 |
| 2 | R/W | GP_ADC_I20U | 1: Adds 20uA constant load current at the ADC LDO to minimize ripple on the reference voltage of the ADC. | 0x0 |
| 1 | R/W | GP_ADC_IDYN | 1: Enables dynamic load current at the ADC LDO to minimize ripple on the reference voltage of the ADC. | 0x0 |
| 0 | R/W | GP_ADC_ATTN3X | 0: Input voltages up to 1.2V allowed. 1: Input voltages up to 3.6V allowed by enabling 3x attenuator. (if ADC_SEL=7 or 8, this bit is automatically set to 1) Enabling the attenuator requires a longer sampling time. | 0x0 |

Table 632: GP_ADC_CTRL3_REG (0x50030908)

| 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. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| | | | 1: 1.024ms interval between two conversions. 2: 2.048ms interval between two conversions. 255: 261.12ms interval between two conversions. | |
| 7:0 | R/W | GP_ADC_EN_DEL | Defines the delay for enabling the ADC after enabling the LDO. 0: Not allowed 1: 32x ADC_CLK period. n: n*32x ADC_CLK period. | 0x40 |

Table 633: GP_ADC_OFFP_REG (0x5003090C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|---|-------|
| 9:0 | R/W | GP_ADC_OFFP | Offset adjust of 'positive' array of ADC-network (effective if "GP_ADC_SE=0", or "GP_ADC_SE=1 AND GP_ADC_SIGN=0") | 0x200 |

Table 634: GP_ADC_OFFN_REG (0x50030910)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|---|-------|
| 9:0 | R/W | GP_ADC_OFFN | Offset adjust of 'negative' array of ADC-network (effective if "GP_ADC_SE=0", or "GP_ADC_SE=1 AND GP_ADC_SIGN=1") | 0x200 |

Table 635: GP_ADC_CLEAR_INT_REG (0x50030914)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 15:0 | R0/W | GP_ADC_CLR_INT | Writing any value to this register will clear the ADC_INT interrupt. Reading returns 0. | 0x0 |

Table 636: GP_ADC_RESULT_REG (0x50030918)

| 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 results in one extra bit and 64 samples results in six extra bits. | 0x0 |

42.22 General Purpose I/O Registers

Table 637: Register map GPIO

| Address | Register | Description |
|------------|-------------|---------------------------------|
| 0x50020A00 | P0_DATA_REG | P0 Data input / output Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|-------------------|---------------------------------|
| 0x50020A04 | P1_DATA_REG | P1 Data input / output Register |
| 0x50020A08 | P0_SET_DATA_REG | P0 Set port pins Register |
| 0x50020A0C | P1_SET_DATA_REG | P1 Set port pins Register |
| 0x50020A10 | P0_RESET_DATA_REG | P0 Reset port pins Register |
| 0x50020A14 | P1_RESET_DATA_REG | P1 Reset port pins Register |
| 0x50020A18 | P0_00_MODE_REG | P0_00 Mode Register |
| 0x50020A1C | P0_01_MODE_REG | P0_01 Mode Register |
| 0x50020A20 | P0_02_MODE_REG | P0_02 Mode Register |
| 0x50020A24 | P0_03_MODE_REG | P0_03 Mode Register |
| 0x50020A28 | P0_04_MODE_REG | P0_04 Mode Register |
| 0x50020A2C | P0_05_MODE_REG | P0_05 Mode Register |
| 0x50020A30 | P0_06_MODE_REG | P0_06 Mode Register |
| 0x50020A34 | P0_07_MODE_REG | P0_07 Mode Register |
| 0x50020A38 | P0_08_MODE_REG | P0_08 Mode Register |
| 0x50020A3C | P0_09_MODE_REG | P0_09 Mode Register |
| 0x50020A40 | P0_10_MODE_REG | P0_10 Mode Register |
| 0x50020A44 | P0_11_MODE_REG | P0_11 Mode Register |
| 0x50020A48 | P0_12_MODE_REG | P0_12 Mode Register |
| 0x50020A4C | P0_13_MODE_REG | P0_13 Mode Register |
| 0x50020A50 | P0_14_MODE_REG | P0_14 Mode Register |
| 0x50020A54 | P0_15_MODE_REG | P0_15 Mode Register |
| 0x50020A58 | P0_16_MODE_REG | P0_16 Mode Register |
| 0x50020A5C | P0_17_MODE_REG | P0_17 Mode Register |
| 0x50020A60 | P0_18_MODE_REG | P0_18 Mode Register |
| 0x50020A64 | P0_19_MODE_REG | P0_19 Mode Register |
| 0x50020A68 | P0_20_MODE_REG | P0_20 Mode Register |
| 0x50020A6C | P0_21_MODE_REG | P0_21 Mode Register |
| 0x50020A70 | P0_22_MODE_REG | P0_22 Mode Register |
| 0x50020A74 | P0_23_MODE_REG | P0_23 Mode Register |
| 0x50020A78 | P0_24_MODE_REG | P0_24 Mode Register |
| 0x50020A7C | P0_25_MODE_REG | P0_25 Mode Register |
| 0x50020A80 | P0_26_MODE_REG | P0_26 Mode Register |
| 0x50020A84 | P0_27_MODE_REG | P0_27 Mode Register |
| 0x50020A88 | P0_28_MODE_REG | P0_28 Mode Register |
| 0x50020A8C | P0_29_MODE_REG | P0_29 Mode Register |
| 0x50020A90 | P0_30_MODE_REG | P0_30 Mode Register |
| 0x50020A94 | P0_31_MODE_REG | P0_31 Mode Register |
| 0x50020A98 | P1_00_MODE_REG | P1_00 Mode Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------|--|
| 0x50020A9C | P1_01_MODE_REG | P1_01 Mode Register |
| 0x50020AA0 | P1_02_MODE_REG | P1_02 Mode Register |
| 0x50020AA4 | P1_03_MODE_REG | P1_03 Mode Register |
| 0x50020AA8 | P1_04_MODE_REG | P1_04 Mode Register |
| 0x50020AAC | P1_05_MODE_REG | P1_05 Mode Register |
| 0x50020AB0 | P1_06_MODE_REG | P1_06 Mode Register |
| 0x50020AB4 | P1_07_MODE_REG | P1_07 Mode Register |
| 0x50020AB8 | P1_08_MODE_REG | P1_08 Mode Register |
| 0x50020ABC | P1_09_MODE_REG | P1_09 Mode Register |
| 0x50020AC0 | P1_10_MODE_REG | P1_10 Mode Register |
| 0x50020AC4 | P1_11_MODE_REG | P1_11 Mode Register |
| 0x50020AC8 | P1_12_MODE_REG | P1_12 Mode Register |
| 0x50020ACC | P1_13_MODE_REG | P1_13 Mode Register |
| 0x50020AD0 | P1_14_MODE_REG | P1_14 Mode Register |
| 0x50020AD4 | P1_15_MODE_REG | P1_15 Mode Register |
| 0x50020AD8 | P1_16_MODE_REG | P1_16 Mode Register |
| 0x50020ADC | P1_17_MODE_REG | P1_17 Mode Register |
| 0x50020AE0 | P1_18_MODE_REG | P1_18 Mode Register |
| 0x50020AE4 | P1_19_MODE_REG | P1_19 Mode Register |
| 0x50020AE8 | P1_20_MODE_REG | P1_20 Mode Register |
| 0x50020AEC | P1_21_MODE_REG | P1_21 Mode Register |
| 0x50020AF0 | P1_22_MODE_REG | P1_22 Mode Register |
| 0x50020AF4 | P0_PADPWR_CTRL_REG | P0 Output Power Control Register |
| 0x50020AF8 | P1_PADPWR_CTRL_REG | P1 Output Power Control Register |
| 0x50020AFC | GPIO_CLK_SEL_REG | Select which clock to map on ports P0/P1 |
| 0x50020B00 | PAD_WEAK_CTRL_REG | Weak Pads Control Register |

Table 638: P0_DATA_REG (0x50020A00)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|---|--------|
| 31:0 | R/W | P0_DATA | Set P0 output register when written; Returns the value of P0 port when read | 0x1410 |

Table 639: P1_DATA_REG (0x50020A04)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|---|-------|
| 22:0 | R/W | P1_DATA | Set P1 output register when written; Returns the value of P1 port when read | 0x0 |

Table 640: P0_SET_DATA_REG (0x50020A08)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--|-------|
| 31:0 | R0/W | P0_SET | Writing a 1 to P0[y] sets P0[y] to 1. Writing 0 is discarded; Reading returns 0 | 0x0 |

Table 641: P1_SET_DATA_REG (0x50020A0C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--|-------|
| 22:0 | R0/W | P1_SET | Writing a 1 to P1[y] sets P1[y] to 1. Writing 0 is discarded; Reading returns 0 | 0x0 |

Table 642: P0_RESET_DATA_REG (0x50020A10)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:0 | R0/W | P0_RESET | Writing a 1 to P0[y] sets P0[y] to 0. Writing 0 is discarded; Reading returns 0 | 0x0 |

Table 643: P1_RESET_DATA_REG (0x50020A14)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 22:0 | R0/W | P1_RESET | Writing a 1 to P1[y] sets P1[y] to 0. Writing 0 is discarded; Reading returns 0 | 0x0 |

Table 644: P0_00_MODE_REG (0x50020A18)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | Function of port: 0: GPIO (see also the PUPD bit-field) 1: UART_RX 2: UART_TX 3: UART2_RX 4: UART2_TX 5: UART2_CTSN 6: UART2_RTSN 7: UART3_RX | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 8: UART3_TX 9: UART3_CTSN 10: UART3_RTSN 11: ISO_CLK 12: ISO_DATA 13: SPI_DI 14: SPI_DO 15: SPI_CLK 16: SPI_EN 17: SPI2_DI 18: SPI2_DO 19: SPI2_CLK 20: SPI2_EN 21: I2C_SCL 22: I2C_SDA 23: I2C2_SCL 24: I2C2_SDA 25: USB_SOF 26: ADC (dedicated pins, see also the "Input/Output Ports" chapter of Datasheet) 27: USB (dedicated pins P0_14 and P0_15) 28: PCM_DI 29: PCM_DO 30: PCM_FSC 31: PCM_CLK 32: PDM_DATA 33: PDM_CLK 34: COEX_EXT_ACT 35: COEX_SMART_ACT 36: COEX_SMART_PRI 37: PORT0_DCF 38: PORT1_DCF 39: PORT2_DCF 40: PORT3_DCF 41: PORT4_DCF 42: CLOCK (see also GPIO_CLK_SEL_REG for the dedicated pins mapping of supported clocks) 43: PG (dedicated pins, see also the "Input/Output Ports" chapter of Datasheet) 44: LCD (dedicated pins see also the "Input/Output Ports" chapter of Datasheet) 45: LCD_SPI_DC 46: LCD_SPI_DO 47: LCD_SPI_CLK 48: LCD_SPI_EN 49: TIM_PWM 50: TIM2_PWM 51: TIM_1SHOT 52: TIM2_1SHOT 53: TIM3_PWM 54: TIM4_PWM 55: Reserved | |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 56: CMAC_DIAG0 57: CMAC_DIAG1 58: CMAC_DIAG2 59: CMAC_DIAGX (dedicated pins, see also the "Input/Output Ports" chapter of Datasheet) 60: Reserved 61: Reserved 62: Reserved 63: Reserved | |

Table 645: P0_01_MODE_REG (0x50020A1C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 646: P0_02_MODE_REG (0x50020A20)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 647: P0_03_MODE_REG (0x50020A24)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------------------|-------|
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 648: P0_04_MODE_REG (0x50020A28)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 | 0x1 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 649: P0_05_MODE_REG (0x50020A2C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 650: P0_06_MODE_REG (0x50020A30)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 651: P0_07_MODE_REG (0x50020A34)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 652: P0_08_MODE_REG (0x50020A38)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 653: P0_09_MODE_REG (0x50020A3C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 654: P0_10_MODE_REG (0x50020A40)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 0x0 |
| 9:8 | R/W | PUPD | 00 = Input, no resistors selected 01 = Input, pull-up selected | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 10 = Input, pull-down selected 11 = Output, no resistors selected In ADC mode, these bits are don't care. | |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 655: P0_11_MODE_REG (0x50020A44)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 656: P0_12_MODE_REG (0x50020A48)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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. | 0x1 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 657: P0_13_MODE_REG (0x50020A4C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------------------|-------|
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 658: P0_14_MODE_REG (0x50020A50)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 659: P0_15_MODE_REG (0x50020A54)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 660: P0_16_MODE_REG (0x50020A58)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 661: P0_17_MODE_REG (0x50020A5C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 662: P0_18_MODE_REG (0x50020A60)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 663: P0_19_MODE_REG (0x50020A64)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 664: P0_20_MODE_REG (0x50020A68)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-----------------------------------|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 0x0 |
| 9:8 | R/W | PUPD | 00 = Input, no resistors selected | 0x2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| 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. | |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 665: P0_21_MODE_REG (0x50020A6C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 666: P0_22_MODE_REG (0x50020A70)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 667: P0_23_MODE_REG (0x50020A74)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------------------|-------|
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 668: P0_24_MODE_REG (0x50020A78)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 669: P0_25_MODE_REG (0x50020A7C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 670: P0_26_MODE_REG (0x50020A80)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 671: P0_27_MODE_REG (0x50020A84)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 672: P0_28_MODE_REG (0x50020A88)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 673: P0_29_MODE_REG (0x50020A8C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 674: P0_30_MODE_REG (0x50020A90)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-----------------------------------|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 0x0 |
| 9:8 | R/W | PUPD | 00 = Input, no resistors selected | 0x2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| 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. | |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 675: P0_31_MODE_REG (0x50020A94)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 676: P1_00_MODE_REG (0x50020A98)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 677: P1_01_MODE_REG (0x50020A9C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 678: P1_02_MODE_REG (0x50020AA0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 679: P1_03_MODE_REG (0x50020AA4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 680: P1_04_MODE_REG (0x50020AA8)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 681: P1_05_MODE_REG (0x50020AAC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 0x0 |
| 9:8 | R/W | PUPD | 00 = Input, no resistors selected 01 = Input, pull-up selected 10 = Input, pull-down selected | 0x2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 11 = Output, no resistors selected In ADC mode, these bits are don't care | |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 682: P1_06_MODE_REG (0x50020AB0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 683: P1_07_MODE_REG (0x50020AB4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 684: P1_08_MODE_REG (0x50020AB8)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 685: P1_09_MODE_REG (0x50020ABC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 686: P1_10_MODE_REG (0x50020AC0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 687: P1_11_MODE_REG (0x50020AC4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 688: P1_12_MODE_REG (0x50020AC8)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-----------------------------------|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 0x0 |
| 9:8 | R/W | PUPD | 00 = Input, no resistors selected | 0x2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| 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. | |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 689: P1_13_MODE_REG (0x50020ACC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 690: P1_14_MODE_REG (0x50020AD0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 691: P1_15_MODE_REG (0x50020AD4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------------------|-------|
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 692: P1_16_MODE_REG (0x50020AD8)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 693: P1_17_MODE_REG (0x50020ADC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 694: P1_18_MODE_REG (0x50020AE0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 695: P1_19_MODE_REG (0x50020AE4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 696: P1_20_MODE_REG (0x50020AE8)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 697: P1_21_MODE_REG (0x50020AEC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 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 |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 698: P1_22_MODE_REG (0x50020AF0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-----------------------------------|-------|
| 10 | R/W | PPOD | 0: Push pull 1: Open drain | 0x0 |
| 9:8 | R/W | PUPD | 00 = Input, no resistors selected | 0x2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| 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. | |
| 7:6 | R | - | | 0x0 |
| 5:0 | R/W | PID | See P0_00_MODE_REG[PID] | 0x0 |

Table 699: P0_PADPWR_CTRL_REG (0x50020AF4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|---|-------|
| 31:6 | R/W | P0_OUT_CTRL | 0 = P0_x port output is powered by the V30 rail (default) 1 = P0_x port output is powered by the 1V8P rail bit x controls the power supply of P0[x] | 0x0 |
| 5:0 | R/W | RESERVED_P0_OUT_CTRL | Reserved bits: P0_05 downto P0_00 are always supplied by V18P. So these bits don't control anything | 0x0 |

Table 700: P1_PADPWR_CTRL_REG (0x50020AF8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 22:0 | R/W | P1_OUT_CTRL | 0 = P1_x port output is powered by the V30 rail (default) 1 = P1_x port output is powered by the 1V8P rail bit x controls the power supply of P1[x] | 0x0 |

Table 701: GPIO_CLK_SEL_REG (0x50020AFC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|--|-------|
| 9 | R/W | DIVN_OUTPUT_EN | DIVN output enable bit-field. When set, it enables the mapping of DIVN clock on dedicated GPIO (P0_15). The specific GPIO must be configured as GPIO output. | 0x0 |
| 8 | R/W | RC32M_OUTPUT_EN | RC32M output enable bit-field. When set, it enables the mapping of RC32M clock on dedicated GPIO (P0_13). The specific GPIO must be configured as GPIO output. | 0x0 |
| 7 | R/W | XTAL32M_OUTPUT_EN | XTAL32M output enable bit-field. When set, it enables the mapping of XTAL32M clock on dedicated GPIO (P0_12). The specific GPIO must be configured as GPIO output. | 0x0 |
| 6 | R/W | RCX_OUTPUT_EN | RCX output enable bit-field. When set, it enables the mapping of RCX clock on dedicated GPIO (P0_16). The specific GPIO must be configured as GPIO output. | 0x0 |
| 5 | R/W | RC32K_OUTPUT_EN | RC32K output enable bit-field. When set, it enables the mapping of RC32K clock on dedicated GPIO (P0_17). The specific GPIO must be configured as | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|---|-------|
| | | | GPIO output. | |
| 4 | R/W | XTAL32K_OUTPUT_EN | XTAL32K output enable bit-field. When set, it enables the mapping of XTAL32K clock on dedicated GPIO (P0_14).The specific GPIO must be configured as GPIO output. | 0x0 |
| 3 | R/W | FUNC_CLOCK_EN | If set, it enables the mapping of the selected clock signal, according to FUNC_CLOCK_SEL bit-field. | 0x0 |
| 2:0 | R/W | FUNC_CLOCK_SEL | Select which clock to map when PID = FUNC_CLOCK. 0x0: XTAL32K 0x1: RC32K 0x2: RCX 0x3: XTAL32M 0x4: RC32M 0x5: DIVN 0x6: Reserved 0x7: Reserved | 0x0 |

Table 702: PAD_WEAK_CTRL_REG (0x50020B00)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| 12 | R/W | P1_09_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P1_09 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P1_PADPWDR_CTRL_REG). | 0x0 |
| 11 | R/W | P1_06_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P1_06 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P1_PADPWDR_CTRL_REG). | 0x0 |
| 10 | R/W | P1_02_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P1_02 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P1_PADPWDR_CTRL_REG). | 0x0 |
| 9 | R/W | P1_01_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P1_01 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P1_PADPWDR_CTRL_REG). | 0x0 |
| 8 | R/W | P1_00_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P1_00 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P1_PADPWDR_CTRL_REG). | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| 7 | R/W | P0_27_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_27 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |
| 6 | R/W | P0_26_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_26 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |
| 5 | R/W | P0_25_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_25 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |
| 4 | R/W | P0_18_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_18 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |
| 3 | R/W | P0_17_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_17 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |
| 2 | R/W | P0_16_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_16 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |
| 1 | R/W | P0_07_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_07 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |
| 0 | R/W | P0_06_LOWDRV | 0 = Normal operation 1 = Reduces the driving strength of P0_06 pad Note: This mode should be coupled with the selection of VDD1V8P supply rail for the specific pad (see also the description of P0_PADPWDR_CTRL_REG). | 0x0 |

42.23 General Purpose Registers

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 703: Register map GPREG

| Address | Register | Description |
|------------|------------------|---|
| 0x50040300 | SET_FREEZE_REG | Controls freezing of various timers/counters (incl. DMA and USB). |
| 0x50040304 | RESET_FREEZE_REG | Controls unfreezing of various timers/counters (incl. DMA and USB). |
| 0x50040308 | DEBUG_REG | Various debug information register. |
| 0x5004030C | GP_STATUS_REG | General purpose system status register. |
| 0x50040310 | GP_CONTROL_REG | General purpose system control register. |
| 0x50040318 | USBPAD_REG | USB pads control register |

Table 704: SET_FREEZE_REG (0x50040300)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---|-------|
| 31:11 | R | - | | 0x0 |
| 10 | R/W | FRZ_CMAC_WDOG | If '1', the CMAC SW Watchdog Timer is frozen, '0' is discarded. | 0x0 |
| 9 | R/W | FRZ_SWTIM4 | If '1', the SW Timer4 is frozen, '0' is discarded. | 0x0 |
| 8 | R/W | FRZ_SWTIM3 | If '1', the SW Timer3 is frozen, '0' is discarded. | 0x0 |
| 7 | R/W | FRZ_PWMLED | If '1', the PWM LED is frozen, '0' is discarded. | 0x0 |
| 6 | R/W | FRZ_SWTIM2 | If '1', the SW Timer2 is frozen, '0' is discarded. | 0x0 |
| 5 | R/W | FRZ_DMA | If '1', the DMA is frozen, '0' is discarded. | 0x0 |
| 4 | R/W | FRZ_USB | If '1', the USB is frozen, '0' is discarded. | 0x0 |
| 3 | R/W | FRZ_SYS_WDOG | If '1', the SYS SW 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_RESERVED | | 0x0 |
| 1 | R/W | FRZ_SWTIM | If '1', the SW Timer is frozen, '0' is discarded. | 0x0 |
| 0 | R/W | FRZ_WKUPTIM | If '1', the Wake Up Timer is frozen, '0' is discarded. | 0x0 |

Table 705: RESET_FREEZE_REG (0x50040304)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---|-------|
| 31:11 | R | - | | 0x0 |
| 10 | R/W | FRZ_CMAC_WDOG | If '1', the CMAC SW Watchdog Timer continues, '0' is discarded. | 0x0 |
| 9 | R/W | FRZ_SWTIM4 | If '1', the SW Timer4 continues, '0' is discarded. | 0x0 |
| 8 | R/W | FRZ_SWTIM3 | If '1', the SW Timer3 continues, '0' is discarded. | 0x0 |
| 7 | R/W | FRZ_PWMLED | If '1', the PWM LED continues, '0' is discarded. | 0x0 |
| 6 | R/W | FRZ_SWTIM2 | If '1', the SW Timer2 continues, '0' is discarded. | 0x0 |
| 5 | R/W | FRZ_DMA | If '1', the DMA continues, '0' is discarded. | 0x0 |
| 4 | R/W | FRZ_USB | If '1', the USB continues, '0' is discarded. | 0x0 |
| 3 | R/W | FRZ_SYS_WDOG | If '1', the SYS SW Watchdog Timer continues, '0' is discarded. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| 2 | R/W | FRZ_RESERVED | | 0x0 |
| 1 | R/W | FRZ_SWTIM | If '1', the SW Timer continues, '0' is discarded. | 0x0 |
| 0 | R/W | FRZ_WKUPTIM | If '1', the Wake Up Timer continues, '0' is discarded. | 0x0 |

Table 706: DEBUG_REG (0x50040308)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------------|---|-------|
| 31:16 | R | - | | 0x0 |
| 15:10 | R | - | | 0x0 |
| 9 | R/W | - | | 0x0 |
| 8 | R/W | CROSS_CPU_HALT_SENSITIVITY | Select the cross CPU halt sensitivity. 0: Level triggered, 1: Pulse triggered. Note: This bit is retained. | 0x1 |
| 7 | R/W | SYS_CPUWAIT_ON_JTAG | 1: Stall the processor core out of reset (only after a wake-up from JTAG). Debugger access continue when the core is stalled. When set to '0' again the core resumes instruction execution. This feature is independent of the PDC (Power Domain Controller) settings. If this bit is set and there is SW/JTAG activity during deep sleep, the SYS CPU is stalled after the wake-up. Note: This bit is retained. | 0x0 |
| 6 | R/W | SYS_CPUWAIT | 1: Stall the processor core out of reset (always after a wake-up). Debugger access continue when the core is stalled. When set to '0' again the core resumes instruction execution. Note: This bit is retained. | 0x0 |
| 5 | R | CMAC_CPU_IS_HALT | 1: CMAC CPU is halted. | 0x0 |
| 4 | R | SYS_CPU_IS_HALT | 1: SYS CPU (ARM CM33) is halted. | 0x0 |
| 3 | R/W | HALT_CMAC_SYS_CPU_EN | 1: Enable CMAC CPU halting to the SYS CPU (ARM CM33). Note 1: This bit is retained. Note 2: Set this bit to '0' before going into deep sleep to prevent unpredictable halting behavior after waking up. | 0x0 |
| 2 | R/W | HALT_SYS_CMAC_CPU_EN | 1: Enable SYS CPU (ARM CM33) halting to the CMAC CPU. Note 1: This bit is retained. Note 2: Set this bit to '0' before going into deep sleep to prevent unpredictable halting behavior after waking up. | 0x0 |
| 1 | R/W | CMAC_CPU_FREEZE_EN | 1: Enable Freezing <u>on-chip peripherals</u> (see Note 2) by the CMAC CPU. Note 1: This bit is retained. Note 2: See [RE]SET_FREEZE_REG for the specific <u>on-chip peripherals</u> . | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|---|-------|
| 0 | R/W | SYS_CPU_FREEZE_EN | <p>1: Enable Freezing on-chip peripherals (see Note 2) by the SYS CPU (ARM CM33).</p> <p>Default '1', freezing of the on-chip peripherals is enabled when the Cortex-M33 is halted in DEBUG State.</p> <p>If '0', freezing of the on-chip peripherals is <u>only</u> depending on [RE]SET_FREEZE_REG except the system watchdog timer. The system watchdog timer is always frozen when the Cortex-M33 is halted in DEBUG State.</p> <p>Note 1: This bit is retained.</p> <p>Note 2: See [RE]SET_FREEZE_REG for the specific <u>on-chip peripherals</u>.</p> | 0x1 |

Table 707: GP_STATUS_REG (0x5004030C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 31:2 | R | - | | 0x0 |
| 1 | R/W | - | | 0x0 |
| 0 | R/W | CAL_PHASE | If '1', it designates that the chip is in Calibration Phase i.e. the OTP has been initially programmed but no Calibration has occurred. | 0x0 |

Table 708: GP_CONTROL_REG (0x50040310)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------------|--|-------|
| 31:2 | R | - | | 0x0 |
| 1 | R/W | CMAC_H2H_BRIDGE_BYPASS | If '1', the AHB-to-AHB bridge is bypassed, reducing the wait cycles needed to access the CMAC Register File, only when the system clock source is the XTAL and both hclk and cmac_hclk are running at 32 MHz, i.e. at the XTAL clock rate. | 0x0 |
| 0 | R/W | - | | 0x0 |

Table 709: USBPAD_REG (0x50040318)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------|--|-------|
| 2 | R/W | USBPHY_FORCE_SW2_ON | 0: Pull up resistor SW2 is controlled by the USB controller. It is off when the USB is not enabled. 1: Force the pull up resistor on USBP to be 2.3Kohm | 0x0 |
| 1 | R/W | USBPHY_FORCE_SW1_OFF | 0: Pull up resistor SW1 is controlled by the USB controller. It is off when the USB is not enabled. 1: Force the pull up resistor on USBP to be switched off. | 0x0 |
| 0 | R/W | USBPAD_EN | 0: The power for the USB PHY and USB pads is switched on when the USB is enabled. 1: The power for the USB PHY and USB pads is forced on. | 0x0 |

42.24 I2C Controller Registers

Table 710: Register map I2C

| Address | Register | Description |
|------------|-----------------------|--|
| 0x50020600 | I2C_CON_REG | I2C Control Register |
| 0x50020604 | I2C_TAR_REG | I2C Target Address Register |
| 0x50020608 | I2C_SAR_REG | I2C Slave Address Register |
| 0x5002060C | I2C_HS_MADDR_REG | I2C High Speed Master Mode Code Address Register |
| 0x50020610 | I2C_DATA_CMD_REG | I2C Rx/Tx Data Buffer and Command Register |
| 0x50020614 | I2C_SS_SCL_HCNT_REG | Standard Speed I2C Clock SCL High Count Register |
| 0x50020618 | I2C_SS_SCL_LCNT_REG | Standard Speed I2C Clock SCL Low Count Register |
| 0x5002061C | I2C_FS_SCL_HCNT_REG | Fast Speed I2C Clock SCL High Count Register |
| 0x50020620 | I2C_FS_SCL_LCNT_REG | Fast Speed I2C Clock SCL Low Count Register |
| 0x50020624 | I2C_HS_SCL_HCNT_REG | High Speed I2C Clock SCL High Count Register |
| 0x50020628 | I2C_HS_SCL_LCNT_REG | High Speed I2C Clock SCL Low Count Register |
| 0x5002062C | I2C_INTR_STAT_REG | I2C Interrupt Status Register |
| 0x50020630 | I2C_INTR_MASK_REG | I2C Interrupt Mask Register |
| 0x50020634 | I2C_RAW_INTR_STAT_REG | I2C Raw Interrupt Status Register |
| 0x50020638 | I2C_RX_TL_REG | I2C Receive FIFO Threshold Register |
| 0x5002063C | I2C_TX_TL_REG | I2C Transmit FIFO Threshold Register |
| 0x50020640 | I2C_CLR_INTR_REG | Clear Combined and Individual Interrupt Register |
| 0x50020644 | I2C_CLR_RX_UNDER_REG | Clear RX_UNDER Interrupt Register |
| 0x50020648 | I2C_CLR_RX_OVER_REG | Clear RX_OVER Interrupt Register |
| 0x5002064C | I2C_CLR_TX_OVER_REG | Clear TX_OVER Interrupt Register |
| 0x50020650 | I2C_CLR_RD_REQ_REG | Clear RD_REQ Interrupt Register |
| 0x50020654 | I2C_CLR_TX_ABRT_REG | Clear TX_ABRT Interrupt Register |
| 0x50020658 | I2C_CLR_RX_DONE_REG | Clear RX_DONE Interrupt Register |
| 0x5002065C | I2C_CLR_ACTIVITY_REG | Clear ACTIVITY Interrupt Register |
| 0x50020660 | I2C_CLR_STOP_DET_REG | Clear STOP_DET Interrupt Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------------|--|
| 0x50020664 | I2C_CLR_START_DET_REG | I2C Clear START_DET Interrupt Register |
| 0x50020668 | I2C_CLR_GEN_CALL_REG | I2C Clear GEN_CALL Interrupt Register |
| 0x5002066C | I2C_ENABLE_REG | I2C Enable Register |
| 0x50020670 | I2C_STATUS_REG | I2C Status Register |
| 0x50020674 | I2C_TXFLR_REG | I2C Transmit FIFO Level Register |
| 0x50020678 | I2C_RXFLR_REG | I2C Receive FIFO Level Register |
| 0x5002067C | I2C_SDA_HOLD_REG | I2C SDA Hold Time Length Register |
| 0x50020680 | I2C_TX_ABRT_SOURCE_REG | I2C Transmit Abort Source Register |
| 0x50020688 | I2C_DMA_CR_REG | DMA Control Register |
| 0x5002068C | I2C_DMA_TDLR_REG | DMA Transmit Data Level Register |
| 0x50020690 | I2C_DMA_RDLR_REG | I2C Receive Data Level Register |
| 0x50020694 | I2C_SDA_SETUP_REG | I2C SDA Setup Register |
| 0x50020698 | I2C_ACK_GENERAL_CALL_REG | I2C ACK General Call Register |
| 0x5002069C | I2C_ENABLE_STATUS_REG | I2C Enable Status Register |
| 0x500206A0 | I2C_IC_FS_SPKLEN_REG | I2C SS and FS spike suppression limit Size |
| 0x500206A4 | I2C_IC_HS_SPKLEN_REG | I2C HS spike suppression limit Size |
| 0x50020700 | I2C2_CON_REG | I2C Control Register |
| 0x50020704 | I2C2_TAR_REG | I2C Target Address Register |
| 0x50020708 | I2C2_SAR_REG | I2C Slave Address Register |
| 0x5002070C | I2C2_HS_MADDR_REG | I2C High Speed Master Mode Code Address Register |
| 0x50020710 | I2C2_DATA_CMD_REG | I2C Rx/Tx Data Buffer and Command Register |
| 0x50020714 | I2C2_SS_SCL_HCNT_REG | Standard Speed I2C Clock SCL High Count Register |
| 0x50020718 | I2C2_SS_SCL_LCNT_REG | Standard Speed I2C Clock SCL Low Count Register |
| 0x5002071C | I2C2_FS_SCL_HCNT_REG | Fast Speed I2C Clock SCL High Count Register |
| 0x50020720 | I2C2_FS_SCL_LCNT_REG | Fast Speed I2C Clock SCL Low Count Register |
| 0x50020724 | I2C2_HS_SCL_HCNT_REG | High Speed I2C Clock SCL High Count Register |
| 0x50020728 | I2C2_HS_SCL_LCNT_REG | High Speed I2C Clock SCL Low Count Register |
| 0x5002072C | I2C2_INTR_STAT_REG | I2C Interrupt Status Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------------|--|
| 0x50020730 | I2C2_INTR_MASK_REG | I2C Interrupt Mask Register |
| 0x50020734 | I2C2_RAW_INTR_STATUS_REG | I2C Raw Interrupt Status Register |
| 0x50020738 | I2C2_RX_TL_REG | I2C Receive FIFO Threshold Register |
| 0x5002073C | I2C2_TX_TL_REG | I2C Transmit FIFO Threshold Register |
| 0x50020740 | I2C2_CLR_INTR_REG | Clear Combined and Individual Interrupt Register |
| 0x50020744 | I2C2_CLR_RX_UNDER_REG | Clear RX_UNDER Interrupt Register |
| 0x50020748 | I2C2_CLR_RX_OVER_REG | Clear RX_OVER Interrupt Register |
| 0x5002074C | I2C2_CLR_TX_OVER_REG | Clear TX_OVER Interrupt Register |
| 0x50020750 | I2C2_CLR_RD_REQ_REG | Clear RD_REQ Interrupt Register |
| 0x50020754 | I2C2_CLR_TX_ABRT_REG | Clear TX_ABRT Interrupt Register |
| 0x50020758 | I2C2_CLR_RX_DONE_REG | Clear RX_DONE Interrupt Register |
| 0x5002075C | I2C2_CLR_ACTIVITY_REG | Clear ACTIVITY Interrupt Register |
| 0x50020760 | I2C2_CLR_STOP_DET_REG | Clear STOP_DET Interrupt Register |
| 0x50020764 | I2C2_CLR_START_DET_REG | Clear START_DET Interrupt Register |
| 0x50020768 | I2C2_CLR_GEN_CALL_REG | Clear GEN_CALL Interrupt Register |
| 0x5002076C | I2C_ENABLE_REG | I2C Enable Register |
| 0x50020770 | I2C_STATUS_REG | I2C Status Register |
| 0x50020774 | I2C_TXFLR_REG | I2C Transmit FIFO Level Register |
| 0x50020778 | I2C_RXFLR_REG | I2C Receive FIFO Level Register |
| 0x5002077C | I2C_SDA_HOLD_REG | I2C SDA Hold Time Length Register |
| 0x50020780 | I2C_TX_ABRT_SOURCE_REG | I2C Transmit Abort Source Register |
| 0x50020788 | I2C_DMA_CR_REG | DMA Control Register |
| 0x5002078C | I2C_DMA_TDRL_REG | DMA Transmit Data Level Register |
| 0x50020790 | I2C_DMA_RDRL_REG | I2C Receive Data Level Register |
| 0x50020794 | I2C_SDA_SETUP_REG | I2C SDA Setup Register |
| 0x50020798 | I2C_ACK_GENERAL_CALL_REG | I2C ACK General Call Register |
| 0x5002079C | I2C_ENABLE_STATUS_REG | I2C Enable Status Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|----------------------|--|
| 0x500207A0 | I2C_IC_FS_SPKLEN_REG | I2C SS and FS spike suppression limit Size |
| 0x500207A4 | I2C_IC_HS_SPKLEN_REG | I2C HS spike suppression limit Size |

Table 711: I2C_CON_REG (0x50020600)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------------|--|-------|
| 31:11 | - | - | | 0x0 |
| 10 | R | I2C_STOP_DET_IF_MASTER_ACTIVE | In Master mode: 1 = issues the STOP_DET interrupt only when master is active. 0 = issues the STOP_DET irrespective of whether master is active or not. | 0x0 |
| 9 | R/W | I2C_RX_FIFO_FULL_HLD_CTRL | This bit controls whether DW_apb_i2c should hold the bus when the Rx FIFO is physically full to its RX_BUFFER_DEPTH 1 = Hold bus when RX_FIFO is full 0 = Overflow when RX_FIFO is full | 0x0 |
| 8 | R/W | I2C_TX_EMPTY_CTRL | This bit controls the generation of the TX_EMPTY interrupt, as described in the IC_RAW_INTR_STAT register. 1 = Controlled generation of TX_EMPTY interrupt 0 = Default behaviour of TX_EMPTY interrupt | 0x0 |
| 7 | R/W | I2C_STOP_DET_IF_ADDRESSED | 1 = slave issues STOP_DET intr only if addressed 0 = slave issues STOP_DET intr always During a general call address, this slave does not issue the STOP_DET interrupt if STOP_DET_IF_ADDRESSED = 1'b1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (SAR). | 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 | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| | | AVE | the controller responds to 7- or 10-bit addresses. 0= 7-bit addressing 1= 10-bit addressing | |
| 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) 3= high speed mode | 0x3 |
| 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 712: I2C_TAR_REG (0x50020604)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 31:12 | - | - | | 0x0 |
| 11 | R/W | SPECIAL | On read 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 On write 1 = Enables programming of GENERAL_CALL or START_BYTE transmission 0 = Disables programming of GENERAL_CALL or START_BYTE transmission Writes to this register succeed only when IC_ENABLE[0] is set to 0. | 0x0 |
| 10 | R/W | GC_OR_START | On read 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 On write 1 = START byte transmission 0 = GENERAL_CALL byte transmission Writes to this register succeed only when IC_ENABLE[0] is set to 0. | 0x0 |
| 9:0 | R/W | IC_TAR | This is the target address for any master | 0x55 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>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.</p> <p>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</p> <p>Writes to this register succeed only when IC_ENABLE[0] is set to 0.</p> | |

Table 713: I2C_SAR_REG (0x50020608)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|--|-------|
| 31:10 | - | - | | 0x0 |
| 9:0 | R/W | IC_SAR | <p>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.</p> <p>Writes to this register succeed only when IC_ENABLE[0] is set to 0.</p> | 0x55 |

Table 714: I2C_HS_MADDR_REG (0x5002060C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| 2:0 | R/W | I2C_IC_HS_MAR | <p>This bit field holds the value of the I2C HS mode master code. HS-mode master codes are reserved 8-bit codes (00001xxx) that are not used for slave addressing or other purposes. Each master has its unique master code; up to eight high-speed mode masters can be present on the same I2C bus system. Valid values are from 0 to 7. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect.</p> | 0x1 |

Table 715: I2C_DATA_CMD_REG (0x50020610)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 30:11 | - | - | | 0x0 |
| 10 | W | I2C_RESTART | <p>This bit controls whether a RESTART is issued before the byte is sent or received.</p> <p>1 = 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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| | | | <p>followed by a START is issued instead.</p> <p>0 = 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.</p> | |
| 9 | W | I2C_STOP | <p>This bit controls whether a STOP is issued after the byte is sent or received.</p> <p>1 = 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.</p> <p>0 = 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.</p> | 0x0 |
| 8 | W | I2C_CMD | <p>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.</p> <p>1 = Read 0 = Write</p> <p>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.</p> <p>If a "1" is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs.</p> <p>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</p> | 0x0 |
| 7:0 | R/W | I2C_DAT | <p>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.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 716: I2C_SS_SCL_HCNT_REG (0x50020614)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 15:0 | R/W | IC_SS_SCL_HCNT | <p>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.</p> <p>The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set.</p> <p>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.</p> | 0x91 |

Table 717: I2C_SS_SCL_LCNT_REG (0x50020618)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | IC_SS_SCL_LCNT | <p>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.</p> <p>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.</p> <p>The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, results in 8 being set.</p> | 0xAB |

Table 718: I2C_FS_SCL_HCNT_REG (0x5002061C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | IC_FS_SCL_HCNT | <p>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.</p> <p>The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set.</p> | 0x1A |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 719: I2C_FS_SCL_LCNT_REG (0x50020620)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | IC_FS_SCL_LCNT | <p>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.</p> <p>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.</p> | 0x32 |

Table 720: I2C_HS_SCL_HCNT_REG (0x50020624)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | IC_HS_SCL_HCNT | <p>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 high speed.refer to "IC_CLK Frequency Configuration".</p> <p>The SCL High time depends on the loading of the bus. For 100pF loading, the SCL High time is 60ns; for 400pF loading, the SCL High time is 120ns. This register goes away and becomes read-only returning 0s if IC_MAX_SPEED_MODE != high.</p> <p>This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect.</p> <p>The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. For designs with APB_DATA_WIDTH = 8 the order of programming is important to ensure the correct operation of the DW_apb_i2c. The lower byte must be programmed first. Then the upper byte is programmed.</p> | 0x6 |

Table 721: I2C_HS_SCL_LCNT_REG (0x50020628)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 15:0 | R/W | IC_HS_SCL_LCNT | <p>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 high speed. For more information, refer to "IC_CLK Frequency Configuration".</p> <p>The SCL low time depends on the loading of the bus. For 100pF loading, the SCL low time is 160ns; for 400pF loading, the SCL low time is 320ns. This register goes away and becomes read-only</p> | 0x10 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>returning 0s if IC_MAX_SPEED_MODE != high.</p> <p>This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect.</p> <p>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 DW_apb_i2c. The lower byte must be programmed first. Then the upper byte is programmed. If the value is less than 8 then the count value gets changed to 8.</p> | |

Table 722: I2C_INTR_STAT_REG (0x5002062C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| 31:15 | - | - | | 0x0 |
| 14 | R | R_SCL_STUCK_AT_LOW | 1 = R_SCL_STUCK_AT_LOW interrupt is active 0 = R_SCL_STUCK_AT_LOW interrupt is inactive | 0x0 |
| 13 | R | R_MASTER_ON_HOLD | Indicates whether master is holding the bus and TX FIFO is empty. Enabled only when I2C_DYNAMIC_TAR_UPDATE=1 and IC_EMPTYFIFO_HOLD_MASTER_EN=1. | 0x0 |
| 12 | R | R_RESTART_DET | Indicates whether a RESTART condition has occurred on the I2C interface when DW_apb_i2c is operating in Slave mode and the slave is being addressed. Enabled only when IC_SLV_RESTART_DET_EN=1. Note: However, in high-speed mode or during a START BYTE transfer, the RESTART comes before the address field as per the I2C protocol. In this case, the slave is not the addressed slave when the RESTART is issued, therefore DW_apb_i2c does not generate the RESTART_DET interrupt. | 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|---|-------|
| | | | => Reading the IC_CLR_ACTIVITY register => Reading the IC_CLR_INTR register => System reset Once 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. | |
| 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 places. 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. Once 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 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 | 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|---|-------|
| | | | 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. | |
| 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 723: I2C_INTR_MASK_REG (0x50020630)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|--|-------|
| 31:15 | - | - | | 0x0 |
| 14 | R | M_SCL_STUCK_AT_LOW | M_SCL_STUCK_AT_LOW Register field Reserved bits | 0x0 |
| 13 | R/W | M_MASTER_ON_HOLD | These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register. | 0x0 |
| 12 | R/W | M_RESTART_DET | These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register. | 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 | These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register. | 0x1 |
| 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| 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 724: I2C_RAW_INTR_STAT_REG (0x50020634)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 31:15 | - | - | | 0x0 |
| 14 | R | SCL_STUCK_AT_LOW | CL_STUCK_AT_LOW Register field Reserved bits | 0x0 |
| 13 | R | MASTER_ON_HOLD | Indicates whether master is holding the bus and TX FIFO is empty. Enabled only when I2C_DYNAMIC_TAR_UPDATE=1 and IC_EMPTYFIFO_HOLD_MASTER_EN=1. | 0x0 |
| 12 | R | RESTART_DET | Indicates whether a RESTART condition has occurred on the I2C interface when DW_apb_i2c is operating in Slave mode and the slave is being addressed. Enabled only when IC_SLV_RESTART_DET_EN=1. Note: However, in high-speed mode or during a START BYTE transfer, the RESTART comes before the address field as per the I2C protocol. In this case, the slave is not the addressed slave when the RESTART is issued, therefore DW_apb_i2c does not generate the RESTART_DET interrupt. | 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 Once 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 | RX_DONE | When the controller is acting as a slave-transmitter, this bit is set to 1 if the master does not | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| | | | acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. | |
| 6 | R | TX_ABRT | <p>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".</p> <p>When this bit is set to 1, the I2C_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places.</p> <p>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. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.</p> | 0x0 |
| 5 | R | RD_REQ | <p>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</p> | 0x0 |
| 4 | R | TX_EMPTY | <p>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.</p> | 0x0 |
| 3 | R | TX_OVER | <p>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</p> | 0x0 |
| 2 | R | RX_FULL | <p>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.</p> | 0x0 |
| 1 | R | RX_OVER | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| | | | machines go into idle, and when ic_en goes to 0, this interrupt is cleared. | |
| 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 725: I2C_RX_TL_REG (0x50020638)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:5 | - | - | | 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 will be 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 726: I2C_TX_TL_REG (0x5002063C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R/W | TX_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 value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be 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 727: I2C_CLR_INTR_REG (0x50020640)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:1 | - | - | | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 728: I2C_CLR_RX_UNDER_REG (0x50020644)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:1 | - | - | | 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 729: I2C_CLR_RX_OVER_REG (0x50020648)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 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 730: I2C_CLR_TX_OVER_REG (0x5002064C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 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 731: I2C_CLR_RD_REQ_REG (0x50020650)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:1 | - | - | | 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 732: I2C_CLR_TX_ABRT_REG (0x50020654)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | CLR_TX_ABRT | Read this register to clear the TX_ABRT interrupt (bit 6) of the I2C_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. Refer to Bit 9 of the I2C_TX_ABRT_SOURCE register for an exception to clearing I2C_TX_ABRT_SOURCE. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 733: I2C_CLR_RX_DONE_REG (0x50020658)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 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 734: I2C_CLR_ACTIVITY_REG (0x5002065C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:1 | - | - | | 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 735: I2C_CLR_STOP_DET_REG (0x50020660)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | CLR_STOP_DET | Read this register to clear the STOP_DET interrupt (bit 9) of the IC_RAW_INTR_STAT register. | 0x0 |

Table 736: I2C_CLR_START_DET_REG (0x50020664)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:1 | - | - | | 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 737: I2C_CLR_GEN_CALL_REG (0x50020668)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | CLR_GEN_CALL | Read this register to clear the GEN_CALL interrupt (bit 11) of I2C_RAW_INTR_STAT register. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 738: I2C_ENABLE_REG (0x5002066C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 31:3 | - | - | | 0x0 |
| 2 | R/W | I2C_TX_CMD_BLOCK | In Master mode: 1 = Blocks the transmission of data on I2C bus even if Tx FIFO has data to transmit. 0.= The transmission of data starts on I2C bus automatically, as soon as the first data is available in the Tx FIFO. | 0x0 |
| 1 | R/W | I2C_ABORT | 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 once 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. | 0x0 |
| 0 | R/W | I2C_EN | 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 739: I2C_STATUS_REG (0x50020670)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------|---|-------|
| 31:11 | - | - | | 0x0 |
| 10 | R | LV_HOLD_RX_FIFO_FULL | This bit indicates the BUS Hold in Slave mode due to Rx FIFO is Full and an additional byte has been received 1 = Slave holds the bus due to Rx FIFO is full 0 = Slave is not holding the bus or Bus hold is not due to Rx FIFO is full | 0x0 |
| 9 | R | SLV_HOLD_TX_FIFO_EMPTY | This bit indicates the BUS Hold in Slave mode for the Read request when the Tx FIFO is empty. The Bus is in hold until the Tx FIFO has data to Transmit for the read request. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------------|---|-------|
| | | | 1 = Slave holds the bus due to Tx FIFO is empty 0 = Slave is not holding the bus or Bus hold is not due to Tx FIFO is empty | |
| 8 | R | MST_HOLD_RX_FIFO_FULL | This bit indicates the BUS Hold in Master mode due to Rx FIFO is Full and additional byte has been received 1 = Master holds the bus due to Rx FIFO is full 0 = Master is not holding the bus or Bus hold is not due to Rx FIFO is full | 0x0 |
| 7 | R | MST_HOLD_TX_FIFO_EMPTY | the DW_apb_i2c master stalls the write transfer when Tx FIFO is empty, and the the last byte does not have the Stop bit set. This bit indicates the BUS hold when the master holds the bus because of the Tx FIFO being empty, and the the previous transferred command does not have the Stop bit set. 1 =Master holds the bus due to Tx FIFO is empty 0 =Master is not holding the bus or Bus hold is not due to Tx FIFO is empty | 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 | 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 | 0x1 |
| 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. | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| | | | 0 = Transmit FIFO is full 1 = Transmit FIFO is not full | |
| 0 | R | I2C_ACTIVITY | I2C Activity Status. | 0x0 |

Table 740: I2C_TXFLR_REG (0x50020674)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:6 | - | - | | 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 741: I2C_RXFLR_REG (0x50020678)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:6 | - | - | | 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 742: I2C_SDA_HOLD_REG (0x5002067C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 23:16 | R/W | I2C_SDA_RX_HOLD | Sets the required SDA hold time in units of ic_clk period, when receiver. | 0x0 |
| 15:0 | R/W | I2C_SDA_TX_HOLD | Sets the required SDA hold time in units of ic_clk period, when transmitter. | 0x1 |

Table 743: I2C_TX_ABRT_SOURCE_REG (0x50020680)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|---|-------|
| 16 | R | ABRT_USER_ABRT | Master-Transmitter : This is a master-mode-only bit. Master has detected the transfer abort (IC_ENABLE[1]) | 0x0 |
| 15 | R | ABRT_SLVRD_INTX | Slave-Transmitter : When the processor side responds to a slave mode request for data to be transmitted to a remote master and user writes a 1 in CMD (bit 8) of 2I2C_DATA_CMD register 1 = Slave trying to transmit to remote master in read mode 0 = Slave trying to transmit to remote master in read mode- scenario not present | 0x0 |
| 14 | R | ABRT_SLV_ARBLOST | Slave-Transmitter : 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 check. For instance, during a data | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------|--|-------|
| | | | 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 own the bus. 1 = Slave lost arbitration to remote master 0 = Slave lost arbitration to remote master- scenario not present | |
| 13 | R | ABRT_SLVFLUSH_TXFIFO | Slave-Transmitter : 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. 1 = Slave flushes existing data in TX-FIFO upon getting read command 0 = Slave flushes existing data in TX-FIFO upon getting read command- scenario not present | 0x0 |
| 12 | R | ARB_LOST | Master-Transmitter or Slave-Transmitter : 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. 1 = Master or Slave-Transmitter lost arbitration 0 = Master or Slave-Transmitter lost arbitration- scenario not present | 0x0 |
| 11 | R | ABRT_MASTER_DIS | Master-Transmitter or Master-Receiver : User tries to initiate a Master operation with the Master mode disabled. 1 = User initiating master operation when MASTER disable 0 = User initiating master operation when MASTER disabled- scenario not present | 0x0 |
| 10 | R | ABRT_10B_RD_NORSTRT | Master-Receiver : The restart is disabled (IC_RESTART_EN bit (I2C_CON[5]) = 0) and the master sends a read command in 10-bit addressing mode. 1 = Master trying to read in 10Bit addressing mode when RESTART disabled 0 = Master not trying to read in 10Bit addressing mode when RESTART disabled | 0x0 |
| 9 | R | ABRT_SBYTE_NORSTRT | Master : 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]). Once 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. 1 = User trying to send START byte when RESTART disabled 0 = User trying to send START byte when RESTART disabled- scenario not present | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|---|-------|
| 8 | R | ABRT_HS_NORSTR_T | <p>Master-Transmitter or Master-Receiver : 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</p> <p>1 = User trying to switch Master to HS mode when RESTART disabled</p> <p>0 = User trying to switch Master to HS mode when RESTART disabled- scenario not present</p> | 0x0 |
| 7 | R | ABRT_SBYTE_ACKDET | <p>Master : Master has sent a START Byte and the START Byte was acknowledged (wrong behavior).</p> <p>1 = ACK detected for START byte</p> <p>0 = ACK detected for START byte- scenario not present</p> | 0x0 |
| 6 | R | ABRT_HS_ACKDET | <p>Master : Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior).</p> <p>1 = HS Master code ACKed in HS Mode</p> <p>0 = HS Master code ACKed in HS Mode- scenario not present</p> | 0x0 |
| 5 | R | ABRT_GCALL_READ | <p>Master-Transmitter : 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).</p> <p>1 = GCALL is followed by read from bus</p> <p>0 = GCALL is followed by read from bus-scenario not present</p> | 0x0 |
| 4 | R | ABRT_GCALL_NOACK | <p>Master-Transmitter : the controller in master mode sent a General Call and no slave on the bus acknowledged the General Call.</p> <p>1 = GCALL not ACKed by any slave</p> <p>0 = GCALL not ACKed by any slave-scenario not present</p> | 0x0 |
| 3 | R | ABRT_TXDATA_NOACK | <p>Master-Transmitter : 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 acknowledge from the remote slave(s).</p> <p>1 = Transmitted data not ACKed by addressed slave</p> <p>0 = Transmitted data non-ACKed by addressed slave-scenario not present</p> | 0x0 |
| 2 | R | ABRT_10ADDR2_NOACK | <p>Master-Transmitter or Master-Receiver : Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave.</p> <p>1= Byte 2 of 10Bit Address not ACKed by any slave</p> <p>0 = This abort is not generated</p> | 0x0 |
| 1 | R | ABRT_10ADDR1_NOACK | <p>Master-Transmitter or Master-Receiver : Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave.</p> <p>1 =Byte 1 of 10Bit Address not ACKed by any slave</p> <p>0 =This abort is not generated</p> | 0x0 |
| 0 | R | ABRT_7B_ADDR_N | Master-Transmitter or Master-Receiver : Master is in 7-bit addressing mode and the address sent was | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | OACK | not acknowledged by any slave. 1 =This abort is generated because of NOACK for 7-bit address 0 =This abort is not generated | |

Table 744: I2C_DMA_CR_REG (0x50020688)

| 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 745: I2C_DMA_TDRL_REG (0x5002068C)

| 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 746: I2C_DMA_RDRL_REG (0x50020690)

| 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 747: I2C_SDA_SETUP_REG (0x50020694)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 15:8 | - | - | | 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 | 0x64 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 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. It is recommended that if the required delay is 1000ns, 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 748: I2C_ACK_GENERAL_CALL_REG (0x50020698)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 15:1 | - | - | | 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. 1 = Generate ACK for a General Call 0 = Generate NACK for General Call | 0x0 |

Table 749: I2C_ENABLE_STATUS_REG (0x5002069C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------------|--|-------|
| 15:3 | - | - | | 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. 1 = Slave RX Data is lost 0 = Slave RX Data is not lost | 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, | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>(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.</p> <p>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 will also be set to 1.</p> <p>When read as 0, the controller is deemed to have been disabled when there is master activity, or when the I2C bus is idle.</p> <p>NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0.</p> <p>1 = Slave is disabled when it is active 0 = Slave is disabled when it is idle</p> | |
| 0 | R | IC_EN | <p>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. When read as 0, the controller is deemed completely inactive.</p> <p>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).</p> <p>1 = I2C enabled 0 = I2C disabled</p> | 0x0 |

Table 750: I2C_IC_FS_SPKLEN_REG (0x500206A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 15:8 | - | - | | 0x0 |
| 7:0 | R/W | I2C_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 |

Table 751: I2C_IC_HS_SPKLEN_REG (0x500206A4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| 7:0 | R/W | I2C_HS_SPKLEN | This register must be set before any I2C bus transaction can take place to ensure stable | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>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[0] register being set to 0. Writes at other times have no effect.</p> <p>The minimum valid value is 1; hardware prevents values less than this being written, and if attempted results in 1 being set.</p> | |

Table 752: I2C2_CON_REG (0x50020700)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------------|---|-------|
| 31:11 | - | - | | 0x0 |
| 10 | R | I2C_STOP_DET_IF_MASTER_ACTIVE | <p>In Master mode:</p> <p>1 = issues the STOP_DET interrupt only when master is active.</p> <p>0 = issues the STOP_DET irrespective of whether master is active or not.</p> | 0x0 |
| 9 | R/W | I2C_RX_FIFO_FULL_HLD_CTRL | <p>This bit controls whether DW_apb_i2c should hold the bus when the Rx FIFO is physically full to its RX_BUFFER_DEPTH</p> <p>1 = Hold bus when RX_FIFO is full</p> <p>0 = Overflow when RX_FIFO is full</p> | 0x0 |
| 8 | R/W | I2C_TX_EMPTY_CTRL | <p>This bit controls the generation of the TX_EMPTY interrupt, as described in the IC_RAW_INTR_STAT register.</p> <p>1 = Controlled generation of TX_EMPTY interrupt</p> <p>0 = Default behaviour of TX_EMPTY interrupt</p> | 0x0 |
| 7 | R/W | I2C_STOP_DET_IF_ADDRESSED | <p>1 = slave issues STOP_DET intr only if addressed</p> <p>0 = slave issues STOP_DET intr always</p> <p>During a general call address, this slave does not issue the STOP_DET interrupt if STOP_DET_IF_ADDRESSED = 1'b1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (SAR).</p> | 0x0 |
| 6 | R/W | I2C_SLAVE_DISABLE | <p>Slave enabled or disabled after reset is applied, which means software does not have to configure the slave.</p> <p>0=slave is enabled</p> <p>1=slave is disabled</p> <p>Software should ensure that if this bit is written with '0', then bit 0 should also be written with a '0'.</p> | 0x1 |
| 5 | R/W | I2C_RESTART_EN | <p>Determines whether RESTART conditions may be sent when acting as a master</p> <p>0= disable</p> <p>1=enable</p> | 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 | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|---|-------|
| | | | master. 0= 7-bit addressing 1= 10-bit addressing | |
| 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) 3= high speed mode | 0x3 |
| 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 753: I2C2_TAR_REG (0x50020704)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 31:12 | - | - | | 0x0 |
| 11 | R/W | SPECIAL | On read 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 On write 1 = Enables programming of GENERAL_CALL or START_BYTE transmission 0 = Disables programming of GENERAL_CALL or START_BYTE transmission Writes to this register succeed only when IC_ENABLE[0] is set to 0. | 0x0 |
| 10 | R/W | GC_OR_START | On read 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 On write 1 = START byte transmission | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 0 = GENERAL_CALL byte transmission Writes to this register succeed only when IC_ENABLE[0] is set to 0. | |
| 9:0 | R/W | IC_TAR | <p>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.</p> <p>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</p> <p>Writes to this register succeed only when IC_ENABLE[0] is set to 0.</p> | 0x55 |

Table 754: I2C2_SAR_REG (0x50020708)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|--|-------|
| 31:10 | - | - | | 0x0 |
| 9:0 | R/W | IC_SAR | <p>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.</p> <p>Writes to this register succeed only when IC_ENABLE[0] is set to 0.</p> | 0x55 |

Table 755: I2C2_HS_MADDR_REG (0x5002070C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| 2:0 | R/W | I2C_IC_HS_MAR | <p>This bit field holds the value of the I2C HS mode master code. HS-mode master codes are reserved 8-bit codes (00001xxx) that are not used for slave addressing or other purposes. Each master has its unique master code; up to eight high-speed mode masters can be present on the same I2C bus system. Valid values are from 0 to 7. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect.</p> | 0x1 |

Table 756: I2C2_DATA_CMD_REG (0x50020710)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 30:11 | - | - | | 0x0 |
| 10 | W | I2C_RESTART | <p>This bit controls whether a RESTART is issued before the byte is sent or received.</p> <p>1 = If IC_RESTART_EN is 1, a RESTART is issued</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| | | | <p>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.</p> <p>0 = 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.</p> | |
| 9 | W | I2C_STOP | <p>This bit controls whether a STOP is issued after the byte is sent or received.</p> <p>1 = 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.</p> <p>0 = 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.</p> | 0x0 |
| 8 | W | I2C_CMD | <p>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.</p> <p>1 = Read 0 = Write</p> <p>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.</p> <p>If a "1" is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs.</p> <p>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</p> | 0x0 |
| 7:0 | R/W | I2C_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 757: I2C2_SS_SCL_HCNT_REG (0x50020714)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 15:0 | R/W | IC_SS_SCL_HCNT | <p>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.</p> <p>The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set.</p> <p>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.</p> | 0x91 |

Table 758: I2C2_SS_SCL_LCNT_REG (0x50020718)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | IC_SS_SCL_LCNT | <p>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.</p> <p>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.</p> <p>The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, results in 8 being set.</p> | 0xAB |

Table 759: I2C2_FS_SCL_HCNT_REG (0x5002071C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | IC_FS_SCL_HCNT | <p>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.</p> <p>The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set.</p> | 0x1A |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 760: I2C2_FS_SCL_LCNT_REG (0x50020720)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 15:0 | R/W | IC_FS_SCL_LCNT | <p>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 IC_ENABLE register being set to 0. Writes at other times have no effect.</p> <p>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.</p> | 0x32 |

Table 761: I2C2_HS_SCL_HCNT_REG (0x50020724)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 15:0 | R/W | IC_HS_SCL_HCNT | <p>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 high speed.refer to "IC_CLK Frequency Configuration".</p> <p>The SCL High time depends on the loading of the bus. For 100pF loading, the SCL High time is 60ns; for 400pF loading, the SCL High time is 120ns. This register goes away and becomes read-only returning 0s if IC_MAX_SPEED_MODE != high.</p> <p>This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect.</p> <p>The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. For designs with APB_DATA_WIDTH = 8 the order of programming is important to ensure the correct operation of the DW_apb_i2c. The lower byte must be programmed first. Then the upper byte is programmed.</p> | 0x6 |

Table 762: I2C2_HS_SCL_LCNT_REG (0x50020728)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 15:0 | R/W | IC_HS_SCL_LCNT | <p>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 high speed. For more information, refer to "IC_CLK Frequency Configuration".</p> <p>The SCL low time depends on the loading of the bus. For 100pF loading, the SCL low time is 160ns; for 400pF loading, the SCL low time is 320ns. This register goes away and becomes read-only</p> | 0x10 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>returning 0s if IC_MAX_SPEED_MODE != high.</p> <p>This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect.</p> <p>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 DW_apb_i2c. The lower byte must be programmed first. Then the upper byte is programmed. If the value is less than 8 then the count value gets changed to 8.</p> | |

Table 763: I2C2_INTR_STAT_REG (0x5002072C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| 31:15 | - | - | | 0x0 |
| 14 | R | R_SCL_STUCK_AT_LOW | 1 = R_SCL_STUCK_AT_LOW interrupt is active 0 = R_SCL_STUCK_AT_LOW interrupt is inactive | 0x0 |
| 13 | R | R_MASTER_ON_HOLD | Indicates whether master is holding the bus and TX FIFO is empty. Enabled only when I2C_DYNAMIC_TAR_UPDATE=1 and IC_EMPTYFIFO_HOLD_MASTER_EN=1. | 0x0 |
| 12 | R | R_RESTART_DET | Indicates whether a RESTART condition has occurred on the I2C interface when DW_apb_i2c is operating in Slave mode and the slave is being addressed. Enabled only when IC_SLV_RESTART_DET_EN=1. Note: However, in high-speed mode or during a START BYTE transfer, the RESTART comes before the address field as per the I2C protocol. In this case, the slave is not the addressed slave when the RESTART is issued, therefore DW_apb_i2c does not generate the RESTART_DET interrupt. | 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|---|-------|
| | | | => Reading the IC_CLR_ACTIVITY register => Reading the IC_CLR_INTR register => System reset Once 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. | |
| 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 places. 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. Once 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 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 | 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|---|-------|
| | | | 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. | |
| 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 764: I2C2_INTR_MASK_REG (0x50020730)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|--|-------|
| 31:15 | - | - | | 0x0 |
| 14 | R | M_SCL_STUCK_AT_LOW | M_SCL_STUCK_AT_LOW Register field Reserved bits | 0x0 |
| 13 | R/W | M_MASTER_ON_HOLD | These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register. | 0x0 |
| 12 | R/W | M_RESTART_DET | These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register. | 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 | These bits mask their corresponding interrupt status bits in the I2C_INTR_STAT register. | 0x1 |
| 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| 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 765: I2C2_RAW_INTR_STAT_REG (0x50020734)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 31:15 | - | - | | 0x0 |
| 14 | R | SCL_STUCK_AT_LOW | CL_STUCK_AT_LOW Register field Reserved bits | 0x0 |
| 13 | R | MASTER_ON_HOLD | Indicates whether master is holding the bus and TX FIFO is empty. Enabled only when I2C_DYNAMIC_TAR_UPDATE=1 and IC_EMPTYFIFO_HOLD_MASTER_EN=1. | 0x0 |
| 12 | R | RESTART_DET | Indicates whether a RESTART condition has occurred on the I2C interface when DW_apb_i2c is operating in Slave mode and the slave is being addressed. Enabled only when IC_SLV_RESTART_DET_EN=1. Note: However, in high-speed mode or during a START BYTE transfer, the RESTART comes before the address field as per the I2C protocol. In this case, the slave is not the addressed slave when the RESTART is issued, therefore DW_apb_i2c does not generate the RESTART_DET interrupt. | 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 Once 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 | RX_DONE | When the controller is acting as a slave-transmitter, this bit is set to 1 if the master does not | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| | | | acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. | |
| 6 | R | TX_ABRT | <p>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".</p> <p>When this bit is set to 1, the I2C_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places.</p> <p>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. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.</p> | 0x0 |
| 5 | R | RD_REQ | <p>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</p> | 0x0 |
| 4 | R | TX_EMPTY | <p>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.</p> | 0x0 |
| 3 | R | TX_OVER | <p>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</p> | 0x0 |
| 2 | R | RX_FULL | <p>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.</p> | 0x0 |
| 1 | R | RX_OVER | <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| | | | machines go into idle, and when ic_en goes to 0, this interrupt is cleared. | |
| 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 766: I2C2_RX_TL_REG (0x50020738)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:5 | - | - | | 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 will be 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 767: I2C2_TX_TL_REG (0x5002073C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--|-------|
| 31:5 | - | - | | 0x0 |
| 4:0 | R/W | TX_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 value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be 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 768: I2C2_CLR_INTR_REG (0x50020740)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 31:1 | - | - | | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 769: I2C2_CLR_RX_UNDER_REG (0x50020744)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:1 | - | - | | 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 770: I2C2_CLR_RX_OVER_REG (0x50020748)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 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 771: I2C2_CLR_TX_OVER_REG (0x5002074C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 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 772: I2C2_CLR_RD_REQ_REG (0x50020750)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:1 | - | - | | 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 773: I2C2_CLR_TX_ABRT_REG (0x50020754)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | CLR_TX_ABRT | Read this register to clear the TX_ABRT interrupt (bit 6) of the I2C_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. Refer to Bit 9 of the I2C_TX_ABRT_SOURCE register for an exception to clearing I2C_TX_ABRT_SOURCE. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 774: I2C2_CLR_RX_DONE_REG (0x50020758)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:1 | - | - | | 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 775: I2C2_CLR_ACTIVITY_REG (0x5002075C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:1 | - | - | | 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 776: I2C2_CLR_STOP_DET_REG (0x50020760)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | CLR_STOP_DET | Read this register to clear the STOP_DET interrupt (bit 9) of the IC_RAW_INTR_STAT register. | 0x0 |

Table 777: I2C2_CLR_START_DET_REG (0x50020764)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 31:1 | - | - | | 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 778: I2C2_CLR_GEN_CALL_REG (0x50020768)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:1 | - | - | | 0x0 |
| 0 | R | CLR_GEN_CALL | Read this register to clear the GEN_CALL interrupt (bit 11) of I2C_RAW_INTR_STAT register. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 779: I2C2_ENABLE_REG (0x5002076C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 31:3 | - | - | | 0x0 |
| 2 | R/W | I2C_TX_CMD_BLOCK | In Master mode: 1 = Blocks the transmission of data on I2C bus even if Tx FIFO has data to transmit. 0.= The transmission of data starts on I2C bus automatically, as soon as the first data is available in the Tx FIFO. | 0x0 |
| 1 | R/W | I2C_ABORT | 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 once 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. | 0x0 |
| 0 | R/W | I2C_EN | 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 780: I2C2_STATUS_REG (0x50020770)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------|---|-------|
| 31:11 | - | - | | 0x0 |
| 10 | R | LV_HOLD_RX_FIFO_FULL | This bit indicates the BUS Hold in Slave mode due to Rx FIFO is Full and an additional byte has been received 1 = Slave holds the bus due to Rx FIFO is full 0 = Slave is not holding the bus or Bus hold is not due to Rx FIFO is full | 0x0 |
| 9 | R | SLV_HOLD_TX_FIFO_EMPTY | This bit indicates the BUS Hold in Slave mode for the Read request when the Tx FIFO is empty. The Bus is in hold until the Tx FIFO has data to Transmit for the read request. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------------|---|-------|
| | | | 1 = Slave holds the bus due to Tx FIFO is empty 0 = Slave is not holding the bus or Bus hold is not due to Tx FIFO is empty | |
| 8 | R | MST_HOLD_RX_FIFO_FULL | This bit indicates the BUS Hold in Master mode due to Rx FIFO is Full and additional byte has been received 1 = Master holds the bus due to Rx FIFO is full 0 = Master is not holding the bus or Bus hold is not due to Rx FIFO is full | 0x0 |
| 7 | R | MST_HOLD_TX_FIFO_EMPTY | the DW_apb_i2c master stalls the write transfer when Tx FIFO is empty, and the the last byte does not have the Stop bit set. This bit indicates the BUS hold when the master holds the bus because of the Tx FIFO being empty, and the the previous transferred command does not have the Stop bit set. 1 =Master holds the bus due to Tx FIFO is empty 0 =Master is not holding the bus or Bus hold is not due to Tx FIFO is empty | 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 | 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 | 0x1 |
| 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. | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|--|-------|
| | | | 0 = Transmit FIFO is full 1 = Transmit FIFO is not full | |
| 0 | R | I2C_ACTIVITY | I2C Activity Status. | 0x0 |

Table 781: I2C2_TXFLR_REG (0x50020774)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:6 | - | - | | 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 782: I2C2_RXFLR_REG (0x50020778)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|---|-------|
| 31:6 | - | - | | 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 783: I2C2_SDA_HOLD_REG (0x5002077C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 23:16 | R/W | I2C_SDA_RX_HOLD | Sets the required SDA hold time in units of ic_clk period, when receiver. | 0x0 |
| 15:0 | R/W | I2C_SDA_TX_HOLD | Sets the required SDA hold time in units of ic_clk period, when transmitter. | 0x1 |

Table 784: I2C2_TX_ABRT_SOURCE_REG (0x50020780)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|--|-------|
| 16 | R | ABRT_USER_ABRT | Master-Transmitter : This is a master-mode-only bit. Master has detected the transfer abort (IC_ENABLE[1]) | 0x0 |
| 15 | R | ABRT_SLVRD_INTX | Slave-Transmitter : When the processor side responds to a slave mode request for data to be transmitted to a remote master and user writes a 1 in CMD (bit 8) of 2IC_DATA_CMD register 1 = Slave trying to transmit to remote master in read mode 0 = Slave trying to transmit to remote master in read mode- scenario not present | 0x0 |
| 14 | R | ABRT_SLV_ARBLOST | Slave-Transmitter : 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 check. For instance, during a data | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------|--|-------|
| | | | 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 own the bus. 1 = Slave lost arbitration to remote master 0 = Slave lost arbitration to remote master- scenario not present | |
| 13 | R | ABRT_SLVFLUSH_TXFIFO | Slave-Transmitter : 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. 1 = Slave flushes existing data in TX-FIFO upon getting read command 0 = Slave flushes existing data in TX-FIFO upon getting read command- scenario not present | 0x0 |
| 12 | R | ARB_LOST | Master-Transmitter or Slave-Transmitter : 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. 1 = Master or Slave-Transmitter lost arbitration 0 = Master or Slave-Transmitter lost arbitration- scenario not present | 0x0 |
| 11 | R | ABRT_MASTER_DIS | Master-Transmitter or Master-Receiver : User tries to initiate a Master operation with the Master mode disabled. 1 = User initiating master operation when MASTER disable 0 = User initiating master operation when MASTER disabled- scenario not present | 0x0 |
| 10 | R | ABRT_10B_RD_NORSTRT | Master-Receiver : The restart is disabled (IC_RESTART_EN bit (I2C_CON[5]) = 0) and the master sends a read command in 10-bit addressing mode. 1 = Master trying to read in 10Bit addressing mode when RESTART disabled 0 = Master not trying to read in 10Bit addressing mode when RESTART disabled | 0x0 |
| 9 | R | ABRT_SBYTE_NORSTRT | Master : 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]). Once 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. 1 = User trying to send START byte when RESTART disabled 0 = User trying to send START byte when RESTART disabled- scenario not present | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|---|-------|
| 8 | R | ABRT_HS_NORSTR_T | <p>Master-Transmitter or Master-Receiver : 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</p> <p>1 = User trying to switch Master to HS mode when RESTART disabled</p> <p>0 = User trying to switch Master to HS mode when RESTART disabled- scenario not present</p> | 0x0 |
| 7 | R | ABRT_SBYTE_ACKDET | <p>Master : Master has sent a START Byte and the START Byte was acknowledged (wrong behavior).</p> <p>1 = ACK detected for START byte</p> <p>0 = ACK detected for START byte- scenario not present</p> | 0x0 |
| 6 | R | ABRT_HS_ACKDET | <p>Master : Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior).</p> <p>1 = HS Master code ACKed in HS Mode</p> <p>0 = HS Master code ACKed in HS Mode- scenario not present</p> | 0x0 |
| 5 | R | ABRT_GCALL_READ | <p>Master-Transmitter : 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).</p> <p>1 = GCALL is followed by read from bus</p> <p>0 = GCALL is followed by read from bus-scenario not present</p> | 0x0 |
| 4 | R | ABRT_GCALL_NOACK | <p>Master-Transmitter : the controller in master mode sent a General Call and no slave on the bus acknowledged the General Call.</p> <p>1 = GCALL not ACKed by any slave</p> <p>0 = GCALL not ACKed by any slave-scenario not present</p> | 0x0 |
| 3 | R | ABRT_TXDATA_NOACK | <p>Master-Transmitter : 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 acknowledge from the remote slave(s).</p> <p>1 = Transmitted data not ACKed by addressed slave</p> <p>0 = Transmitted data non-ACKed by addressed slave-scenario not present</p> | 0x0 |
| 2 | R | ABRT_10ADDR2_NOACK | <p>Master-Transmitter or Master-Receiver : Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave.</p> <p>1= Byte 2 of 10Bit Address not ACKed by any slave</p> <p>0 = This abort is not generated</p> | 0x0 |
| 1 | R | ABRT_10ADDR1_NOACK | <p>Master-Transmitter or Master-Receiver : Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave.</p> <p>1 =Byte 1 of 10Bit Address not ACKed by any slave</p> <p>0 =This abort is not generated</p> | 0x0 |
| 0 | R | ABRT_7B_ADDR_N | Master-Transmitter or Master-Receiver : Master is in 7-bit addressing mode and the address sent was | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | OACK | not acknowledged by any slave. 1 =This abort is generated because of NOACK for 7-bit address 0 =This abort is not generated | |

Table 785: I2C2_DMA_CR_REG (0x50020788)

| 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 786: I2C2_DMA_TDRL_REG (0x5002078C)

| 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 <code>dma_tx_req</code> 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 787: I2C2_DMA_RDRL_REG (0x50020790)

| 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, <code>dma_rx_req</code> 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 <code>dma_rx_req</code> is asserted when 1 or more data entries are present in the receive FIFO. | 0x0 |

Table 788: I2C2_SDA_SETUP_REG (0x50020794)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------|---|-------|
| 15:8 | - | - | | 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 | 0x64 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | 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. It is recommended that if the required delay is 1000ns, 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 789: I2C2_ACK_GENERAL_CALL_REG (0x50020798)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 15:1 | - | - | | 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. 1 = Generate ACK for a General Call 0 = Generate NACK for General Call | 0x0 |

Table 790: I2C2_ENABLE_STATUS_REG (0x5002079C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------------|--|-------|
| 15:3 | - | - | | 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. 1 = Slave RX Data is lost 0 = Slave RX Data is not lost | 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, | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>(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.</p> <p>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 will also be set to 1.</p> <p>When read as 0, the controller is deemed to have been disabled when there is master activity, or when the I2C bus is idle.</p> <p>NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0.</p> <p>1 = Slave is disabled when it is active 0 = Slave is disabled when it is idle</p> | |
| 0 | R | IC_EN | <p>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. When read as 0, the controller is deemed completely inactive.</p> <p>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).</p> <p>1 = I2C enabled 0 = I2C disabled</p> | 0x0 |

Table 791: I2C2_IC_FS_SPKLEN_REG (0x500207A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|---|-------|
| 15:8 | - | - | | 0x0 |
| 7:0 | R/W | I2C_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 |

Table 792: I2C2_IC_HS_SPKLEN_REG (0x500207A4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| 7:0 | R/W | I2C_HS_SPKLEN | This register must be set before any I2C bus transaction can take place to ensure stable | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>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[0] register being set to 0. Writes at other times have no effect.</p> <p>The minimum valid value is 1; hardware prevents values less than this being written, and if attempted results in 1 being set.</p> | |

42.25 Power Domain Controller Registers

Table 793: Register map PDC

| Address | Register | Description |
|------------|-----------------------|--------------------------------|
| 0x50000200 | PDC_CTRL0_REG | PDC control register |
| 0x50000204 | PDC_CTRL1_REG | PDC control register |
| 0x50000208 | PDC_CTRL2_REG | PDC control register |
| 0x5000020C | PDC_CTRL3_REG | PDC control register |
| 0x50000210 | PDC_CTRL4_REG | PDC control register |
| 0x50000214 | PDC_CTRL5_REG | PDC control register |
| 0x50000218 | PDC_CTRL6_REG | PDC control register |
| 0x5000021C | PDC_CTRL7_REG | PDC control register |
| 0x50000220 | PDC_CTRL8_REG | PDC control register |
| 0x50000224 | PDC_CTRL9_REG | PDC control register |
| 0x50000228 | PDC_CTRL10_REG | PDC control register |
| 0x5000022C | PDC_CTRL11_REG | PDC control register |
| 0x50000230 | PDC_CTRL12_REG | PDC control register |
| 0x50000234 | PDC_CTRL13_REG | PDC control register |
| 0x50000238 | PDC_CTRL14_REG | PDC control register |
| 0x5000023C | PDC_CTRL15_REG | PDC control register |
| 0x50000280 | PDC_ACKNOWLEDGE_REG | Clear a pending PDC bit |
| 0x50000284 | PDC_PENDING_REG | Shows any pending wakeup event |
| 0x50000288 | PDC_PENDING_SNC_REG | Shows any pending IRQ to SNC |
| 0x5000028C | PDC_PENDING_CM33_REG | Shows any pending IRQ to CM33 |
| 0x50000290 | PDC_PENDING_CMA_C_REG | Shows any pending IRQ to CM33 |
| 0x50000294 | PDC_SET_PENDING_REG | Set a pending PDC bit |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 794: PDC_CTRL0_REG (0x50000200)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|--|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | Selects which individual bit from the selected bank is used for wakeup. For the peripheral banks, selected with TRIG_SELECT = 0x2 or 0x3, only the lower 4 bits are considered. | 0x0 |
| 1:0 | R/W | TRIG_SELECT | Selects which bank is used as wakeup trigger When TRIG_SELECT is 0x0, selects GPIO port0 through the WAKEUP block. When TRIG_SELECT is 0x1, selects GPIO port1 through the WAKEUP block. When TRIG_SELECT is 0x2 or 0x3, selects the peripheral IRQ. peripheral IRQ table: 0x0: Timer 0x1: Timer2 0x2: Timer3 0x3: Timer4 0x4: RTC Alarm/Rollover 0x5: RTC Timer 0x6: CMAC Timer OR wake up from CMAC debugger 0x7: Motor Controller 0x8: XTAL32MRDY_IRQ 0x9: RFDIAG_IRQ 0xA: CMAC2SYS_IRQ OR VBUS Present IRQ OR JTAG present OR Debounced IO 0xB: Sensor Node Controller 0xC to 0xE: reserved 0xF: Software trigger only | 0x0 |

Table 795: PDC_CTRL1_REG (0x50000204)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| | | | 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 796: PDC_CTRL2_REG (0x50000208)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 797: PDC_CTRL3_REG (0x5000020C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---------------------------|-------|
| | | | PDC_CTRL0_REG.TRIG_SELECT | |

Table 798: PDC_CTRL4_REG (0x50000210)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 799: PDC_CTRL5_REG (0x50000214)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 800: PDC_CTRL6_REG (0x50000218)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|---|-------|
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 801: PDC_CTRL7_REG (0x5000021C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 802: PDC_CTRL8_REG (0x50000220)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 803: PDC_CTRL9_REG (0x50000224)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 804: PDC_CTRL10_REG (0x50000228)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 805: PDC_CTRL11_REG (0x5000022C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 806: PDC_CTRL12_REG (0x50000230)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 807: PDC_CTRL13_REG (0x50000234)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 808: PDC_CTRL14_REG (0x50000238)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|---|-------|
| | | | 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 809: PDC_CTRL15_REG (0x5000023C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|---|-------|
| 12:11 | R/W | PDC_MASTER | Chooses which master is triggered when waking up 0x0: entry is disabled. 0x1: PD_SYS is woken up and CM33 is triggered 0x2: PD_RAD is woken up and CMAC is triggered 0x3: PD_COM is woken up and SNC is triggered | 0x0 |
| 10 | R/W | EN_COM | If set, enables PD_COM for GPIO access. This bit is implied when PDC_MASTER=SNC | 0x0 |
| 9 | R/W | EN_PER | If set, enables PD_PER | 0x0 |
| 8 | R/W | EN_TMR | If set, enables PD_TMR | 0x0 |
| 7 | R/W | EN_XTAL | If set, the XTAL32M will be started | 0x0 |
| 6:2 | R/W | TRIG_ID | For description, see PDC_CTRL0_REG.TRIG_ID | 0x0 |
| 1:0 | R/W | TRIG_SELECT | For description, see PDC_CTRL0_REG.TRIG_SELECT | 0x0 |

Table 810: PDC_ACKNOWLEDGE_REG (0x50000280)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|--|-------|
| 4:0 | W | PDC_ACKNOWLEDGE | Writing to this field acknowledges the PDC IRQ request. The data controls which request is acknowledged | 0x0 |

Table 811: PDC_PENDING_REG (0x50000284)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|-------------------------------------|-------|
| 15:0 | R | PDC_PENDING | Indicates which IRQ ids are pending | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 812: PDC_PENDING_SNC_REG (0x50000288)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 15:0 | R | PDC_PENDING | Indicates which IRQ ids are pending towards the SensorNodeController | 0x0 |

Table 813: PDC_PENDING_CM33_REG (0x5000028C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 15:0 | R | PDC_PENDING | Indicates which IRQ ids are pending towards the CM33 | 0x0 |

Table 814: PDC_PENDING_CMAC_REG (0x50000290)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 15:0 | R | PDC_PENDING | Indicates which IRQ ids are pending towards the CMAC | 0x0 |

Table 815: PDC_SET_PENDING_REG (0x50000294)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|---|-------|
| 4:0 | W | PDC_SET_PENDIN G | Writing to this field sets the PDC wakeup request and IRQ. The data controls which request is acknowledged | 0x0 |

42.26 DCDC Converter Registers

Table 816: Register map DCDC

| Address | Register | Description |
|------------|---------------------|--------------------------------|
| 0x50000304 | DCDC_CTRL1_REG | DCDC First Control Register |
| 0x50000308 | DCDC_CTRL2_REG | DCDC Second Control Register |
| 0x5000030C | DCDC_V14_REG | DCDC V14 Control Register |
| 0x50000310 | DCDC_VDD_REG | DCDC VDD Control Register |
| 0x50000314 | DCDC_V18_REG | DCDC V18 Control Register |
| 0x50000318 | DCDC_V18P_REG | DCDC V18P Control Register |
| 0x50000320 | DCDC_STATUS1_REG | DCDC First Status Register |
| 0x50000330 | DCDC_IRQ_STATUS_REG | DCDC Interrupt Status Register |
| 0x50000334 | DCDC_IRQ_CLEAR_REG | DCDC Interrupt Clear Register |
| 0x50000338 | DCDC_IRQ_MASK_REG | DCDC Interrupt Mask Register |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 817: DCDC_CTRL1_REG (0x50000304)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------------------|--|-------|
| 31 | R/W | DCDC_SH_ENABLE | Enables sample and hold circuit in output comparators. | 0x1 |
| 30:26 | R/W | DCDC_STARTUP_DELAY | Delay between turning bias on and converter becoming active 0 - 31 us, 1 us step size | 0xA |
| 25:20 | R/W | DCDC_IDLE_MAX_FAST_DOWNRAMP | Maximum output idle time for fast current limit downramping. 0 - 7875 ns, 125 ns step size | 0x20 |
| 19:15 | R/W | DCDC_SW_TIMEOUT | P and N switch timeout, if switch is closed longer than this a timeout is generated and the FSM is forced to the next state Writing 0 disables timeout functionality 62.5 - 1937.5 ns, 62.5 ns step size | 0x10 |
| 14 | R/W | DCDC_FAST_STARTUP | Set current limit to maximum during initial startup | 0x0 |
| 13 | R/W | DCDC_MAN_LV_MODE | Manually activates low voltage settings | 0x0 |
| 12 | R/W | DCDC_AUTO_LV_MODE | Switches to low voltage settings when battery voltage drops below 2.5 V | 0x1 |
| 11:10 | R/W | DCDC_IDLE_CLK_DIV | Idle Clock Divider 00 = 2 01 = 4 10 = 8 11 = 16 | 0x1 |
| 9:2 | R/W | DCDC_PRIORITY | Charge priority register (4x 2 bit ID) Charge sequence is [1:0] > [3:2] > [5:4] > [7:6] V14 = 00 V18 = 01 VDD = 10 V18P = 11 | 0xB4 |
| 1 | R/W | DCDC_FW_ENABLE | Freewheel switch enable | 0x0 |
| 0 | R/W | DCDC_ENABLE | Enable setting for DCDC converter | 0x0 |

Table 818: DCDC_CTRL2_REG (0x50000308)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------------|--|-------|
| 27:24 | R/W | DCDC_V_NOK_CNT_MAX | Maximum number of V_NOK events on an output before V_AVAILABLE is reset | 0x8 |
| 23 | R | - | | 0x0 |
| 22 | R/W | DCDC_N_COMP_T_RIM_MAN | Enables manual trimming for N side comparator | 0x0 |
| 21:16 | R/W | DCDC_N_COMP_T_RIM_VAL | Manual trim value for N side comparator Signed magnitude representation | 0x8 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------|--|-------|
| | | | 011111 = +13 mV 000000 = 100000 = -22 mV 111111 = -56 mV | |
| 15:12 | R/W | DCDC_TIMEOUT_I_RQ_TRIG | Number of timeout events before timeout interrupt is generated | 0x8 |
| 11:8 | R/W | DCDC_TIMEOUT_I_RQ_RES | Number of successive non-timed out charge events required to clear timeout event counter | 0x8 |
| 7:6 | R/W | DCDC_SLOPE_CONTROL | Sets strength of N and P switch drivers | 0x3 |
| 5:4 | R/W | DCDC_VBTSTRP_TRIM | Trim bootstrap voltage $V = 1.6 \text{ V} + 100 \text{ mV} * N$ | 0x1 |
| 3:2 | R/W | DCDC_LSSUP_TRIM | Trim low side supply voltage $V = 2 \text{ V} + 300 \text{ mV} * N$ | 0x3 |
| 1:0 | R/W | DCDC_HSGND_TRIM | Trim high side ground $V = V_{BAT} - (2 \text{ V} + 400 \text{ mV} * N)$ | 0x3 |

Table 819: DCDC_V14_REG (0x5000030C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|--|-------|
| 31 | R/W | DCDC_V14_FAST_RAMPING | Fast current ramping (improves response time at the cost of more ripple) | 0x0 |
| 30:28 | R | - | | 0x0 |
| 27 | R/W | DCDC_V14_TRIM | Output voltage trim Steps of 25 mV | 0x0 |
| 26:22 | R/W | DCDC_V14_CUR_LIM_MAX_HV | Maximum current limit (high battery voltage mode) $I = 30 \text{ mA} * (1 + N)$ | 0xD |
| 21:17 | R/W | DCDC_V14_CUR_LIM_MAX_LV | Maximum current limit (low battery voltage mode) $I = 30 \text{ mA} * (1 + N)$ | 0x6 |
| 16:12 | R/W | DCDC_V14_CUR_LIM_MIN | Minimum current limit $I = 30 \text{ mA} * (1 + N)$ | 0x4 |
| 11:7 | R/W | DCDC_V14_IDLE_HYST | Idle time hysteresis 0 - 3875 ns, 125 ns step size $IDLE_MAX = IDLE_MIN + IDLE_HYST$ Maximum idle time before decreasing CUR_LIM | 0x4 |
| 6:2 | R/W | DCDC_V14_IDLE_MIN | Minimum idle time 0 - 3875 ns, 125 ns step size Minimum idle time, CUR_LIM is increased if this limit is not reached | 0x10 |
| 1 | R/W | DCDC_V14_ENABLE_HV | Output enable (high battery voltage mode) 0 = Disabled 1 = Enabled | 0x1 |
| 0 | R/W | DCDC_V14_ENABLE_LV | Output enable (low battery voltage mode) 0 = Disabled 1 = Enabled | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 820: DCDC_VDD_REG (0x50000310)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|---|-------|
| 31 | R/W | DCDC_VDD_FAST_RAMPING | Fast current ramping (improves response time at the cost of more ripple) | 0x0 |
| 30 | R | - | | 0x0 |
| 29:27 | R/W | DCDC_VDD_TRIM | Output voltage trim Steps of 25 mV | 0x4 |
| 26:22 | R/W | DCDC_VDD_CUR_LIM_MAX_HV | Maximum current limit (high battery voltage mode) $I = 30 \text{ mA} * (1 + N)$ | 0x18 |
| 21:17 | R/W | DCDC_VDD_CUR_LIM_MAX_LV | Maximum current limit (low battery voltage mode) $I = 30 \text{ mA} * (1 + N)$ | 0xD |
| 16:12 | R/W | DCDC_VDD_CUR_LIM_MIN | Minimum current limit $I = 30 \text{ mA} * (1 + N)$ | 0x4 |
| 11:7 | R/W | DCDC_VDD_IDLE_HYST | Idle time hysteresis 0 - 3875 ns, 125 ns step size IDLE_MAX = IDLE_MIN + IDLE_HYST Maximum idle time before decreasing CUR_LIM | 0x4 |
| 6:2 | R/W | DCDC_VDD_IDLE_MIN | Minimum idle time 0 - 3875 ns, 125 ns step size Minimum idle time, CUR_LIM is increased if this limit is not reached | 0x10 |
| 1 | R/W | DCDC_VDD_ENABL_E_HV | Output enable (high battery voltage mode) 0 = Disabled 1 = Enabled | 0x1 |
| 0 | R/W | DCDC_VDD_ENABL_E_LV | Output enable (low battery voltage mode) 0 = Disabled 1 = Enabled | 0x1 |

Table 821: DCDC_V18_REG (0x50000314)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|---|-------|
| 31 | R/W | DCDC_V18_FAST_RAMPING | Fast current ramping (improves response time at the cost of more ripple) | 0x0 |
| 30:27 | R/W | DCDC_V18_TRIM | Output voltage trim Steps of 25 mV | 0x8 |
| 26:22 | R/W | DCDC_V18_CUR_LIM_MAX_HV | Maximum current limit (high battery voltage mode) $I = 30 \text{ mA} * (1 + N)$ | 0x1F |
| 21:17 | R/W | DCDC_V18_CUR_LIM_MAX_LV | Maximum current limit (low battery voltage mode) $I = 30 \text{ mA} * (1 + N)$ | 0x1F |
| 16:12 | R/W | DCDC_V18_CUR_LIM_MIN | Minimum current limit $I = 30 \text{ mA} * (1 + N)$ | 0x4 |
| 11:7 | R/W | DCDC_V18_IDLE_HYST | Idle time hysteresis 0 - 3875 ns, 125 ns step size IDLE_MAX = IDLE_MIN + IDLE_HYST Maximum idle time before decreasing CUR_LIM | 0x4 |
| 6:2 | R/W | DCDC_V18_IDLE_M | Minimum idle time | 0x10 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|---|-------|
| | | IN | 0 - 3875 ns, 125 ns step size Minimum idle time, CUR_LIM is increased if this limit is not reached | |
| 1 | R/W | DCDC_V18_ENABL_E_HV | Output enable (high battery voltage mode) 0 = Disabled 1 = Enabled | 0x1 |
| 0 | R/W | DCDC_V18_ENABL_E_LV | Output enable (low battery voltage mode) 0 = Disabled 1 = Enabled | 0x0 |

Table 822: DCDC_V18P_REG (0x50000318)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------------|---|-------|
| 31 | R/W | DCDC_V18P_FAST_RAMPING | Fast current ramping (improves response time at the cost of more ripple) | 0x0 |
| 30:27 | R/W | DCDC_V18P_TRIM | Output voltage trim Steps of 25 mV | 0x8 |
| 26:22 | R/W | DCDC_V18P_CUR_LIM_MAX_HV | Maximum current limit (high battery voltage mode) I = 30 mA * (1 + N) | 0x1F |
| 21:17 | R/W | DCDC_V18P_CUR_LIM_MAX_LV | Maximum current limit (low battery voltage mode) I = 30 mA * (1 + N) | 0x1F |
| 16:12 | R/W | DCDC_V18P_CUR_LIM_MIN | Minimum current limit I = 30 mA * (1 + N) | 0x4 |
| 11:7 | R/W | DCDC_V18P_IDLE_HYST | Idle time hysteresis 0 - 3875 ns, 125 ns step size IDLE_MAX = IDLE_MIN + IDLE_HYST Maximum idle time before decreasing CUR_LIM | 0x4 |
| 6:2 | R/W | DCDC_V18P_IDLE_MIN | Minimum idle time 0 - 3875 ns, 125 ns step size Minimum idle time, CUR_LIM is increased if this limit is not reached | 0x10 |
| 1 | R/W | DCDC_V18P_ENABLE_HV | Output enable (high battery voltage mode) 0 = Disabled 1 = Enabled | 0x1 |
| 0 | R/W | DCDC_V18P_ENABLE_LV | Output enable (low battery voltage mode) 0 = Disabled 1 = Enabled | 0x0 |

Table 823: DCDC_STATUS1_REG (0x50000320)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------|---|-------|
| 27 | R | DCDC_V18P_AVAIL_ABLE | Indicates whether V18P is available Requires that converter is enabled, output is enabled and V_OK has occurred. Reset when too many V_NOK events have occurred. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|--|-------|
| 26 | R | DCDC_VDD_AVAIL_ABLE | Indicates whether VDD is available Requires that converter is enabled, output is enabled and V_OK has occurred. Reset when too many V_NOK events have occurred. | 0x0 |
| 25 | R | DCDC_V18_AVAILABLE | Indicates whether V18 is available Requires that converter is enabled, output is enabled and V_OK has occurred. Reset when too many V_NOK events have occurred. | 0x0 |
| 24 | R | DCDC_V14_AVAILABLE | Indicates whether V14 is available Requires that converter is enabled, output is enabled and V_OK has occurred. Reset when too many V_NOK events have occurred. | 0x0 |
| 23 | R | DCDC_V18P_COMP_OK | OK output of V18P comparator | 0x0 |
| 22 | R | DCDC_VDD_COMP_OK | OK output of VDD comparator | 0x0 |
| 21 | R | DCDC_V18_COMP_OK | OK output of V18 comparator | 0x0 |
| 20 | R | DCDC_V14_COMP_OK | OK output of V14 comparator | 0x0 |
| 19 | R | DCDC_V18P_COMP_NOK | NOK output of V18P comparator | 0x0 |
| 18 | R | DCDC_VDD_COMP_NOK | NOK output of VDD comparator | 0x0 |
| 17 | R | DCDC_V18_COMP_NOK | NOK output of V18 comparator | 0x0 |
| 16 | R | DCDC_V14_COMP_NOK | NOK output of V14 comparator | 0x0 |
| 15:12 | R | - | | 0x0 |
| 11 | R | DCDC_N_COMP_P | DCDC N side dynamic comparator P output | 0x0 |
| 10 | R | DCDC_N_COMP_N | DCDC N side dynamic comparator N output | 0x0 |
| 9 | R | DCDC_P_COMP | DCDC P side continuous time comparator output | 0x0 |
| 8 | R | DCDC_N_COMP | DCDC N side continuous time comparator output | 0x0 |
| 7 | R | DCDC_LV_MODE | Indicates if the converter is in low battery voltage mode | 0x0 |
| 6 | R | DCDC_V18P_SW_STATE | DCDC state machine V18P output | 0x0 |
| 5 | R | DCDC_VDD_SW_STATE | DCDC state machine VDD output | 0x0 |
| 4 | R | DCDC_V18_SW_STATE | DCDC state machine V18 output | 0x0 |
| 3 | R | DCDC_V14_SW_STATE | DCDC state machine V14 output | 0x0 |
| 2 | R | DCDC_N_SW_STATE | DCDC state machine NSW output | 0x0 |
| 1 | R | DCDC_P_SW_STATE | DCDC state machine PSW output | 0x0 |
| 0 | R | DCDC_STARTUP_C | Indicates if the converter is enabled and the startup | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------|--|-------|
| | | OMPLETE | counter has expired (internal biasing settled) | |

Table 824: DCDC_IRQ_STATUS_REG (0x50000330)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------------------|---|-------|
| 4 | R | DCDC_LOW_VBAT_IRQ_STATUS | Low VBAT detector triggered (battery voltage below 2.5 V) | 0x0 |
| 3 | R | DCDC_V18P_TIME_OUT_IRQ_STATUS | Timeout occurred on V18P output | 0x0 |
| 2 | R | DCDC_VDD_TIMEO_UT_IRQ_STATUS | Timeout occurred on VDD output | 0x0 |
| 1 | R | DCDC_V18_TIMEO_UT_IRQ_STATUS | Timeout occurred on V18 output | 0x0 |
| 0 | R | DCDC_V14_TIMEO_UT_IRQ_STATUS | Timeout occurred on V14 output | 0x0 |

Table 825: DCDC_IRQ_CLEAR_REG (0x50000334)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------------------|------------------------------|-------|
| 4 | R0/W | DCDC_LOW_VBAT_IRQ_CLEAR | Clear low VBAT interrupt | 0x0 |
| 3 | R0/W | DCDC_V18P_TIME_OUT_IRQ_CLEAR | Clear V18P timeout interrupt | 0x0 |
| 2 | R0/W | DCDC_VDD_TIMEO_UT_IRQ_CLEAR | Clear VDD timeout interrupt | 0x0 |
| 1 | R0/W | DCDC_V18_TIMEO_UT_IRQ_CLEAR | Clear V18 timeout interrupt | 0x0 |
| 0 | R0/W | DCDC_V14_TIMEO_UT_IRQ_CLEAR | Clear V14 timeout interrupt | 0x0 |

Table 826: DCDC_IRQ_MASK_REG (0x50000338)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------------|-----------------------------|-------|
| 4 | R/W | DCDC_LOW_VBAT_IRQ_MASK | Mask low VBAT interrupt | 0x0 |
| 3 | R/W | DCDC_V18P_TIME_OUT_IRQ_MASK | Mask V18P timeout interrupt | 0x1 |
| 2 | R/W | DCDC_VDD_TIMEO_UT_IRQ_MASK | Mask VDD timeout interrupt | 0x1 |
| 1 | R/W | DCDC_V18_TIMEO_UT_IRQ_MASK | Mask V18 timeout interrupt | 0x1 |
| 0 | R/W | DCDC_V14_TIMEO_UT_IRQ_MASK | Mask V14 timeout interrupt | 0x1 |

42.27 DMA Controller Registers

Table 827: Register map DMA

| Address | Register | Description |
|------------|------------------|--|
| 0x50040800 | DMA0_A_START_REG | Start address A of DMA channel 0 |
| 0x50040804 | DMA0_B_START_REG | Start address B of DMA channel 0 |
| 0x50040808 | DMA0_INT_REG | DMA receive interrupt register channel 0 |
| 0x5004080C | DMA0_LEN_REG | DMA receive length register channel 0 |
| 0x50040810 | DMA0_CTRL_REG | Control register for the DMA channel 0 |
| 0x50040814 | DMA0_IDX_REG | Index value of DMA channel 0 |
| 0x50040820 | DMA1_A_START_REG | Start address A of DMA channel 1 |
| 0x50040824 | DMA1_B_START_REG | Start address B of DMA channel 1 |
| 0x50040828 | DMA1_INT_REG | DMA receive interrupt register channel 1 |
| 0x5004082C | DMA1_LEN_REG | DMA receive length register channel 1 |
| 0x50040830 | DMA1_CTRL_REG | Control register for the DMA channel 1 |
| 0x50040834 | DMA1_IDX_REG | Index value of DMA channel 1 |
| 0x50040840 | DMA2_A_START_REG | Start address A of DMA channel 2 |
| 0x50040844 | DMA2_B_START_REG | Start address B of DMA channel 2 |
| 0x50040848 | DMA2_INT_REG | DMA receive interrupt register channel 2 |
| 0x5004084C | DMA2_LEN_REG | DMA receive length register channel 2 |
| 0x50040850 | DMA2_CTRL_REG | Control register for the DMA channel 2 |
| 0x50040854 | DMA2_IDX_REG | Index value of DMA channel 2 |
| 0x50040860 | DMA3_A_START_REG | Start address A of DMA channel 3 |
| 0x50040864 | DMA3_B_START_REG | Start address B of DMA channel 3 |
| 0x50040868 | DMA3_INT_REG | DMA receive interrupt register channel 3 |
| 0x5004086C | DMA3_LEN_REG | DMA receive length register channel 3 |
| 0x50040870 | DMA3_CTRL_REG | Control register for the DMA channel 3 |
| 0x50040874 | DMA3_IDX_REG | Index value of DMA channel 3 |
| 0x50040880 | DMA4_A_START_REG | Start address A of DMA channel 4 |
| 0x50040884 | DMA4_B_START_REG | Start address B of DMA channel 4 |
| 0x50040888 | DMA4_INT_REG | DMA receive interrupt register channel 4 |
| 0x5004088C | DMA4_LEN_REG | DMA receive length register channel 4 |
| 0x50040890 | DMA4_CTRL_REG | Control register for the DMA channel 4 |
| 0x50040894 | DMA4_IDX_REG | Index value of DMA channel 4 |
| 0x500408A0 | DMA5_A_START_REG | Start address A of DMA channel 5 |
| 0x500408A4 | DMA5_B_START_REG | Start address B of DMA channel 5 |
| 0x500408A8 | DMA5_INT_REG | DMA receive interrupt register channel 5 |
| 0x500408AC | DMA5_LEN_REG | DMA receive length register channel 5 |
| 0x500408B0 | DMA5_CTRL_REG | Control register for the DMA channel 5 |
| 0x500408B4 | DMA5_IDX_REG | Index value of DMA channel 5 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------|--|
| 0x500408C0 | DMA6_A_START_REG | Start address A of DMA channel 6 |
| 0x500408C4 | DMA6_B_START_REG | Start address B of DMA channel 6 |
| 0x500408C8 | DMA6_INT_REG | DMA receive interrupt register channel 6 |
| 0x500408CC | DMA6_LEN_REG | DMA receive length register channel 6 |
| 0x500408D0 | DMA6_CTRL_REG | Control register for the DMA channel 6 |
| 0x500408D4 | DMA6_IDX_REG | Index value of DMA channel 6 |
| 0x500408E0 | DMA7_A_START_REG | Start address A of DMA channel 7 |
| 0x500408E4 | DMA7_B_START_REG | Start address B of DMA channel 7 |
| 0x500408E8 | DMA7_INT_REG | DMA receive interrupt register channel 7 |
| 0x500408EC | DMA7_LEN_REG | DMA receive length register channel 7 |
| 0x500408F0 | DMA7_CTRL_REG | Control register for the DMA channel 7 |
| 0x500408F4 | DMA7_IDX_REG | Index value of DMA channel 7 |
| 0x50040900 | DMA_REQ_MUX_REG | DMA channel assignments |
| 0x50040904 | DMA_INT_STATUS_REG | DMA interrupt status register |
| 0x50040908 | DMA_CLEAR_INT_REG | DMA clear interrupt register |
| 0x5004090C | DMA_INT_MASK_REG | DMA Interrupt mask register |

Table 828: DMA0_A_START_REG (0x50040800)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|----------------------|-------|
| 31:0 | R/W | DMA0_A_START | Source start address | 0x0 |

Table 829: DMA0_B_START_REG (0x50040804)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---------------------------|-------|
| 31:0 | R/W | DMA0_B_START | Destination start address | 0x0 |

Table 830: DMA0_INT_REG (0x50040808)

| 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 DMA _X _INT_REG is equal to DMA _X _IDX_REG and before DMA _X _IDX_REG is incremented. The bit-field DMA _X _IRQ_ENABLE _X of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 831: DMA0_LEN_REG (0x5004080C)

| 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 832: DMA0_CTRL_REG (0x50040810)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 15 | R/W | BUS_ERROR_DETECT | <p>0 = Ignores bus error response from the AHB bus, so DMA continues normally.</p> <p>1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically.</p> <p>It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is switched-on again, in order to perform a new transfer.</p> | 0x1 |
| 14:13 | R/W | BURST_MODE | <p>Enables the DMA read/write bursts, according to the following configuration:</p> <p>00 = Bursts are disabled</p> <p>01 = Bursts of 4 are enabled</p> <p>10 = Bursts of 8 are enabled</p> <p>11 = Reserved</p> | 0x0 |
| 12 | R/W | REQ_SENSE | <p>0 = DMA operates with level-sensitive peripheral requests (default)</p> <p>1 = DMA operates with (positive) edge-sensitive peripheral requests</p> | 0x0 |
| 11 | R/W | DMA_INIT | <p>0 = DMA performs copy A1 to B1, A2 to B2, etc ...</p> <p>1 = DMA performs copy of A1 to B1, B2, etc ...</p> <p>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'.</p> | 0x0 |
| 10 | R/W | DMA_IDLE | <p>0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority.</p> <p>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.</p> | 0x0 |
| 9:7 | R/W | DMA_PRIO | <p>The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. In specific:</p> <p>000 = lowest priority</p> <p>111 = highest priority</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | 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 will first be granted access to the bus. | |
| 6 | 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 |
| 5 | 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 |
| 4 | 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 |
| 3 | 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 |
| 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 will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set. Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 833: DMA0_IDX_REG (0x50040814)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 15:0 | R | DMA0_IDX | <p>This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on.</p> <p>When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again.</p> | 0x0 |

Table 834: DMA1_A_START_REG (0x50040820)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|----------------------|-------|
| 31:0 | R/W | DMA1_A_START | Source start address | 0x0 |

Table 835: DMA1_B_START_REG (0x50040824)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---------------------------|-------|
| 31:0 | R/W | DMA1_B_START | Destination start address | 0x0 |

Table 836: DMA1_INT_REG (0x50040828)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R/W | DMA1_INT | <p>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 DMA_IRQ_ENABLEx of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt.</p> | 0x0 |

Table 837: DMA1_LEN_REG (0x5004082C)

| 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 838: DMA1_CTRL_REG (0x50040830)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 15 | R/W | BUS_ERROR_DETECT | <p>0 = Ignores bus error response from the AHB bus, so DMA continues normally.</p> <p>1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically.</p> <p>It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is switched-on again, in order to perform a new transfer.</p> | 0x1 |
| 14:13 | R/W | BURST_MODE | <p>Enables the DMA read/write bursts, according to the following configuration:</p> <p>00 = Bursts are disabled</p> <p>01 = Bursts of 4 are enabled</p> <p>10 = Bursts of 8 are enabled</p> <p>11 = Reserved</p> | 0x0 |
| 12 | R/W | REQ_SENSE | <p>0 = DMA operates with level-sensitive peripheral requests (default)</p> <p>1 = DMA operates with (positive) edge-sensitive peripheral requests</p> | 0x0 |
| 11 | R/W | DMA_INIT | <p>0 = DMA performs copy A1 to B1, A2 to B2, etc ...</p> <p>1 = DMA performs copy of A1 to B1, B2, etc ...</p> <p>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'.</p> | 0x0 |
| 10 | R/W | DMA_IDLE | <p>0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority.</p> <p>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.</p> | 0x0 |
| 9:7 | R/W | DMA_PRIO | <p>The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. In specific:</p> <p>000 = lowest priority</p> <p>111 = highest priority</p> <p>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 will first be granted access to the bus.</p> | 0x0 |
| 6 | 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | 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. | |
| 5 | 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 |
| 4 | 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 |
| 3 | 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 |
| 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 will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set. Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | 0x0 |

Table 839: DMA1_IDX_REG (0x50040834)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R | DMA1_IDX | This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on. When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again. | |

Table 840: DMA2_A_START_REG (0x50040840)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|----------------------|-------|
| 31:0 | R/W | DMA2_A_START | Source start address | 0x0 |

Table 841: DMA2_B_START_REG (0x50040844)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---------------------------|-------|
| 31:0 | R/W | DMA2_B_START | Destination start address | 0x0 |

Table 842: DMA2_INT_REG (0x50040848)

| 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 DMA_IRQ_ENABLEx of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt. | 0x0 |

Table 843: DMA2_LEN_REG (0x5004084C)

| 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 844: DMA2_CTRL_REG (0x50040850)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|---|-------|
| 15 | R/W | BUS_ERROR_DETECT | 0 = Ignores bus error response from the AHB bus, so DMA continues normally. 1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically. It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| | | | switched-on again, in order to perform a new transfer. | |
| 14:13 | R/W | BURST_MODE | Enables the DMA read/write bursts, according to the following configuration: 00 = Bursts are disabled 01 = Bursts of 4 are enabled 10 = Bursts of 8 are enabled 11 = Reserved | 0x0 |
| 12 | R/W | REQ_SENSE | 0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests | 0x0 |
| 11 | R/W | DMA_INIT | 0 = DMA performs copy A1 to B1, A2 to B2, etc ... 1 = DMA performs copy of A1 to B1, B2, etc ... 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 |
| 10 | 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 |
| 9:7 | R/W | DMA_PRIO | The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are 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 will first be granted access to the bus. | 0x0 |
| 6 | 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 |
| 5 | 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 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| 4 | R/W | BINC | Enable increment of destination address 0 = do not increment 1 = increment according value of BW | 0x0 |
| 3 | 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 |
| 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 will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set. Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | 0x0 |

Table 845: DMA2_IDX_REG (0x50040854)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R | DMA2_IDX | This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on. When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again. | 0x0 |

Table 846: DMA3_A_START_REG (0x50040860)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|----------------------|-------|
| 31:0 | R/W | DMA3_A_START | Source start address | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 847: DMA3_B_START_REG (0x50040864)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---------------------------|-------|
| 31:0 | R/W | DMA3_B_START | Destination start address | 0x0 |

Table 848: DMA3_INT_REG (0x50040868)

| 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 DMA_IRQ_ENABLEx of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt. | 0x0 |

Table 849: DMA3_LEN_REG (0x5004086C)

| 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 850: DMA3_CTRL_REG (0x50040870)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 15 | R/W | BUS_ERROR_DETECT | 0 = Ignores bus error response from the AHB bus, so DMA continues normally. 1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically. It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is switched-on again, in order to perform a new transfer. | 0x1 |
| 14:13 | R/W | BURST_MODE | Enables the DMA read/write bursts, according to the following configuration: 00 = Bursts are disabled 01 = Bursts of 4 are enabled 10 = Bursts of 8 are enabled 11 = Reserved | 0x0 |
| 12 | R/W | REQ_SENSE | 0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests | 0x0 |
| 11 | R/W | DMA_INIT | 0 = DMA performs copy A1 to B1, A2 to B2, etc ... 1 = DMA performs copy of A1 to B1, B2, etc ... This feature is useful for memory initialization to any | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| | | | 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'. | |
| 10 | 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 |
| 9:7 | R/W | DMA_PRIO | The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are 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 will first be granted access to the bus. | 0x0 |
| 6 | 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 |
| 5 | 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 |
| 4 | 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 |
| 3 | 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 |
| 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) | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | transfers) 11 = Reserved | |
| 0 | R/W | DMA_ON | 0 = DMA channel is off, clocks are disabled 1 = DMA channel is enabled. This bit will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set. Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | 0x0 |

Table 851: DMA3_IDX_REG (0x50040874)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R | DMA3_IDX | This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on. When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again. | 0x0 |

Table 852: DMA4_A_START_REG (0x50040880)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|----------------------|-------|
| 31:0 | R/W | DMA4_A_START | Source start address | 0x0 |

Table 853: DMA4_B_START_REG (0x50040884)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---------------------------|-------|
| 31:0 | R/W | DMA4_B_START | Destination start address | 0x0 |

Table 854: DMA4_INT_REG (0x50040888)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R/W | DMA4_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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | before DMAx_IDX_REG is incremented. The bit-field DMA IRQ_ENABLEEx of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt. | |

Table 855: DMA4_LEN_REG (0x5004088C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R/W | DMA4_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 856: DMA4_CTRL_REG (0x50040890)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 15 | R/W | BUS_ERROR_DETECT | 0 = Ignores bus error response from the AHB bus, so DMA continues normally. 1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically. It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is switched-on again, in order to perform a new transfer. | 0x1 |
| 14:13 | R/W | BURST_MODE | Enables the DMA read/write bursts, according to the following configuration: 00 = Bursts are disabled 01 = Bursts of 4 are enabled 10 = Bursts of 8 are enabled 11 = Reserved | 0x0 |
| 12 | R/W | REQ_SENSE | 0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests | 0x0 |
| 11 | R/W | DMA_INIT | 0 = DMA performs copy A1 to B1, A2 to B2, etc ... 1 = DMA performs copy of A1 to B1, B2, etc ... 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 |
| 10 | 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', | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | DMA_IDLE is don't care. | |
| 9:7 | R/W | DMA_PRIO | <p>The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. In specific:</p> <p>000 = lowest priority 111 = highest priority</p> <p>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 will first be granted access to the bus.</p> | 0x0 |
| 6 | R/W | CIRCULAR | <p>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.</p> <p>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.</p> | 0x0 |
| 5 | R/W | AINC | <p>Enable increment of source address.</p> <p>0 = do not increment (source address stays the same during the transfer)</p> <p>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")</p> | 0x0 |
| 4 | R/W | BINC | <p>Enable increment of destination address.</p> <p>0 = do not increment (destination address stays the same during the transfer)</p> <p>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")</p> | 0x0 |
| 3 | R/W | DREQ_MODE | <p>0 = DMA channel starts immediately</p> <p>1 = DMA channel must be triggered by peripheral DMA request (see also the description of DMA_REQ_MUX_REG)</p> | 0x0 |
| 2:1 | R/W | BW | <p>Bus transfer width:</p> <p>00 = 1 Byte (suggested for peripherals like UART and 8-bit SPI)</p> <p>01 = 2 Bytes (suggested for peripherals like I2C and 16-bit SPI)</p> <p>10 = 4 Bytes (suggested for Memory-to-Memory transfers)</p> <p>11 = Reserved</p> | 0x0 |
| 0 | R/W | DMA_ON | <p>0 = DMA channel is off, clocks are disabled</p> <p>1 = DMA channel is enabled. This bit will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set.</p> <p>Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | |

Table 857: DMA4_IDX_REG (0x50040894)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R | DMA4_IDX | This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on. When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again. | 0x0 |

Table 858: DMA5_A_START_REG (0x500408A0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|----------------------|-------|
| 31:0 | R/W | DMA5_A_START | Source start address | 0x0 |

Table 859: DMA5_B_START_REG (0x500408A4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---------------------------|-------|
| 31:0 | R/W | DMA5_B_START | Destination start address | 0x0 |

Table 860: DMA5_INT_REG (0x500408A8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 15:0 | R/W | DMA5_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 DMA_IRQ_ENABLEx of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt. | 0x0 |

Table 861: DMA5_LEN_REG (0x500408AC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R/W | DMA5_LEN | DMA channel's transfer length. DMAx_LEN of value 0, 1, 2, ... results into an actual transfer length of 1, | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------|-------|
| | | | 2, 3, ... | |

Table 862: DMA5_CTRL_REG (0x500408B0)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 15 | R/W | BUS_ERROR_DETECT | <p>0 = Ignores bus error response from the AHB bus, so DMA continues normally.</p> <p>1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically.</p> <p>It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is switched-on again, in order to perform a new transfer.</p> | 0x1 |
| 14:13 | R/W | BURST_MODE | <p>Enables the DMA read/write bursts, according to the following configuration:</p> <p>00 = Bursts are disabled</p> <p>01 = Bursts of 4 are enabled</p> <p>10 = Bursts of 8 are enabled</p> <p>11 = Reserved</p> | 0x0 |
| 12 | R/W | REQ_SENSE | <p>0 = DMA operates with level-sensitive peripheral requests (default)</p> <p>1 = DMA operates with (positive) edge-sensitive peripheral requests</p> | 0x0 |
| 11 | R/W | DMA_INIT | <p>0 = DMA performs copy A1 to B1, A2 to B2, etc ...</p> <p>1 = DMA performs copy of A1 to B1, B2, etc ...</p> <p>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'.</p> | 0x0 |
| 10 | R/W | DMA_IDLE | <p>0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority.</p> <p>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.</p> | 0x0 |
| 9:7 | R/W | DMA_PRIO | <p>The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. In specific:</p> <p>000 = lowest priority</p> <p>111 = highest priority</p> <p>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</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | DMA1 channels have the same priority level, then DMA0 will first be granted access to the bus. | |
| 6 | 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 |
| 5 | 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 |
| 4 | 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 |
| 3 | 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 |
| 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 will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set. Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | 0x0 |

Table 863: DMA5_IDX_REG (0x500408B4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R | DMA5_IDX | This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on.</p> <p>When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again.</p> | |

Table 864: DMA6_A_START_REG (0x500408C0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|----------------------|-------|
| 31:0 | R/W | DMA6_A_START | Source start address | 0x0 |

Table 865: DMA6_B_START_REG (0x500408C4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---------------------------|-------|
| 31:0 | R/W | DMA6_B_START | Destination start address | 0x0 |

Table 866: DMA6_INT_REG (0x500408C8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R/W | DMA6_INT | <p>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 DMA_IRQ_ENABLEx of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt.</p> | 0x0 |

Table 867: DMA6_LEN_REG (0x500408CC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R/W | DMA6_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 868: DMA6_CTRL_REG (0x500408D0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|--|-------|
| 15 | R/W | BUS_ERROR_DETECT | <p>0 = Ignores bus error response from the AHB bus, so DMA continues normally.</p> <p>1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| | | | <p>write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically.</p> <p>It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is switched-on again, in order to perform a new transfer.</p> | |
| 14:13 | R/W | BURST_MODE | <p>Enables the DMA read/write bursts, according to the following configuration:</p> <ul style="list-style-type: none"> 00 = Bursts are disabled 01 = Bursts of 4 are enabled 10 = Bursts of 8 are enabled 11 = Reserved | 0x0 |
| 12 | R/W | REQ_SENSE | <p>0 = DMA operates with level-sensitive peripheral requests (default)</p> <p>1 = DMA operates with (positive) edge-sensitive peripheral requests</p> | 0x0 |
| 11 | R/W | DMA_INIT | <p>0 = DMA performs copy A1 to B1, A2 to B2, etc ...</p> <p>1 = DMA performs copy of A1 to B1, B2, etc ...</p> <p>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'.</p> | 0x0 |
| 10 | R/W | DMA_IDLE | <p>0 = Blocking mode, the DMA performs a fast back-to-back copy, disabling bus access for any bus master with lower priority.</p> <p>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.</p> | 0x0 |
| 9:7 | R/W | DMA_PRIO | <p>The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. In specific:</p> <ul style="list-style-type: none"> 000 = lowest priority 111 = highest priority <p>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 will first be granted access to the bus.</p> | 0x0 |
| 6 | R/W | CIRCULAR | <p>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.</p> <p>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.</p> | 0x0 |
| 5 | R/W | AINC | Enable increment of source address. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | 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") | |
| 4 | 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 |
| 3 | 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 |
| 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 will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set. Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | 0x0 |

Table 869: DMA6_IDX_REG (0x500408D4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R | DMA6_IDX | This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on. When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again. | 0x0 |

Table 870: DMA7_A_START_REG (0x500408E0)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:0 | R/W | DMA7_A_START | Source start address NOTE: See also the DMA chapter of the Datasheet for the allowed range of the DMA7 channel's source address in Secure Boot mode. | 0x0 |

Table 871: DMA7_B_START_REG (0x500408E4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:0 | R/W | DMA7_B_START | Destination start address NOTE: See also the DMA chapter of the Datasheet for the allowed range of the DMA7 channel's destination address in Secure Boot mode. | 0x0 |

Table 872: DMA7_INT_REG (0x500408E8)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 15:0 | R/W | DMA7_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 DMA_IRQ_ENABLEx of DMA_INT_MASK_REG must be set to '1' to let the controller generate the interrupt. | 0x0 |

Table 873: DMA7_LEN_REG (0x500408EC)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R/W | DMA7_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 874: DMA7_CTRL_REG (0x500408F0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|---|-------|
| 15 | R/W | BUS_ERROR_DETECT | 0 = Ignores bus error response from the AHB bus, so DMA continues normally. 1 = Detects the bus response and tracks any bus error may occur during the transfer. If a bus error is detected, the channel completes the current read-write DMA cycle (either in burst or single transfers mode) and then closes the transfer, de-asserting DMA_ON bit automatically. It is noted that the respective bus error detection status bit of DMA_INT_STATUS_REG is automatically cleared as soon as the channel is switched-on again, in order to perform a new transfer. NOTE: In secure boot mode, the bus error detection | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|--|-------|
| | | | mode of DMA7 channel is always enabled, overruling the specific bit-field's programmed value. | |
| 14:13 | R/W | BURST_MODE | <p>Enables the DMA read/write bursts, according to the following configuration:</p> <p>00 = Bursts are disabled 01 = Bursts of 4 are enabled 10 = Bursts of 8 are enabled 11 = Reserved</p> | 0x0 |
| 12 | R/W | REQ_SENSE | <p>0 = DMA operates with level-sensitive peripheral requests (default) 1 = DMA operates with (positive) edge-sensitive peripheral requests</p> | 0x0 |
| 11 | R/W | DMA_INIT | <p>0 = DMA performs copy A1 to B1, A2 to B2, etc ... 1 = DMA performs copy of A1 to B1, B2, etc ...</p> <p>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'.</p> <p>NOTE: This bit-field is overruled to '0' when the DMA7 channel is configured as "trusted" channel (in Secure Boot mode).</p> | 0x0 |
| 10 | R/W | DMA_IDLE | <p>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.</p> <p>*NOTE: This bit-field is overruled to '0' when the DMA7 channel is configured as "trusted" channel (in Secure Boot mode).</p> | 0x0 |
| 9:7 | R/W | DMA_PRIO | <p>The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. In specific:</p> <p>000 = lowest priority 111 = highest priority</p> <p>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 will first be granted access to the bus.</p> | 0x0 |
| 6 | R/W | CIRCULAR | <p>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.</p> <p>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.</p> | 0x0 |
| 5 | R/W | AINC | Enable increment of source address. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|--|-------|
| | | | 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") | |
| 4 | 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 |
| 3 | 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) *NOTE: This bit-field is overruled to '0' when channel DMA7 is configured as "trusted" channel (in Secure Boot mode). | 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 NOTE: This bit-field is overruled to "10" when channel DMA7 is configured as "trusted" channel (in Secure Boot mode). | 0x0 |
| 0 | R/W | DMA_ON | 0 = DMA channel is off, clocks are disabled 1 = DMA channel is enabled. This bit will be automatically cleared after the completion of a transfer, if circular mode is not enabled. In circular mode, this bit stays set. Note: If DMA_ON is disabled by SW while the DMA channel is active, it cannot be enabled again until the channel has completed the last on-going read-write cycle and has stopped. Thus, the SW has to check that the reading of DMAx_CTRL_REG.DMA_ON returns 0, before setting again the specific bit-field. | 0x0 |

Table 875: DMA7_IDX_REG (0x500408F4)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|--|-------|
| 15:0 | R | DMA7_IDX | This (read-only) register determines the data items already transferred by the DMA channel. Hence, if its value is 1, then the DMA channel has already copied one data item and it is currently performing the next copy. If its value is 2, then two items have already been copied and so on. When the transfer is completed (so when DMAx_CTRL_REG.DMA_ON has been cleared) and DMAx_CTRL_REG.CIRCULAR is not set, the | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | register keeps its (last) value (which should be equal to DMAx_LEN_REG) and it is automatically reset to 0 upon starting a new transfer. In CIRCULAR mode, the register is automatically initialized to 0 as soon as the DMA channel starts-over again. | |

Table 876: DMA_REQ_MUX_REG (0x50040900)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|--|-------|
| 15:12 | R/W | DMA67_SEL | Select which combination of peripherals are mapped on the DMA channels. The peripherals are mapped as pairs on two channels. Here, the first DMA request is mapped on channel 6 and the second on channel 7. See DMA01_SEL for the peripheral mapping. | 0xF |
| 11:8 | R/W | DMA45_SEL | Select which combination of peripherals are mapped on the DMA channels. The peripherals are mapped as pairs on two channels. Here, the first DMA request is mapped on channel 4 and the second on channel 5. See DMA01_SEL for the peripherals' mapping. | 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. Here, the first DMA request is mapped on channel 2 and the second on channel 3. See DMA01_SEL for the peripherals' mapping. | 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. Here, the first DMA request is mapped on channel 0 and the second on channel 1. 0x0: SPI_rx / SPI_tx 0x1: SPI2_rx / SPI2_tx 0x2: UART_rx / UART_tx 0x3: UART2_rx / UART2_tx 0x4: I2C_rx / I2C_tx 0x5: I2C2_rx / I2C2_tx 0x6: USB_rx / USB_tx 0x7: UART3_rx/UART3_tx 0x8: PCM_rx / PCM_tx 0x9: SRC_out / SRC_in (for all the supported conversions) 0xA: Reserved 0xB: Reserved 0xC: GP_ADC / - 0xD: SD_ADC / - 0xE: Reserved 0xF: None Note: If any of the four available peripheral selector | 0xF |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | fields (DMA01_SEL, DMA23_SEL, DMA45_SEL, DMA67_SEL) have the same value, the lesser significant selector has higher priority and will control the DMA acknowledge signal driven to the selected peripheral. Hence, if DMA01_SEL = DMA23_SEL, the channels 0 and 1 will provide the Rx and Tx DMA acknowledge signals for the selected peripheral. Consequently, it is suggested to assign the intended peripheral value to a unique selector field. | |

Table 877: DMA_INT_STATUS_REG (0x50040904)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|---|-------|
| 15 | R | DMA_BUS_ERR7 | 0 = No bus error response is detected for channel 7 1 = Bus error response detected for channel 7 NOTE: This bit-field is auto-clear and it is initialized to '0' as soon as a new transfer is started. It is also noted that when the specific channel becomes secure (so when either of the PROT_AES_KEY_READ and PROT_QSPI_KEY_READ bits of SECURE_BOOT_REG is set), this bit-field is overruled to '0', masking the bus error status reporting to the user. | 0x0 |
| 14 | R | DMA_BUS_ERR6 | 0 = No bus error response is detected for channel 6 1 = Bus error response detected for channel 6 NOTE: This bit-field is auto-clear and it is initialized to '0' as soon as a new transfer is started. | 0x0 |
| 13 | R | DMA_BUS_ERR5 | 0 = No bus error response is detected for channel 5 1 = Bus error response detected for channel 5 NOTE: This bit-field is auto-clear and it is initialized to '0' as soon as a new transfer is started. | 0x0 |
| 12 | R | DMA_BUS_ERR4 | 0 = No bus error response is detected for channel 4 1 = Bus error response detected for channel 4 NOTE: This bit-field is auto-clear and it is initialized to '0' as soon as a new transfer is started. | 0x0 |
| 11 | R | DMA_BUS_ERR3 | 0 = No bus error response is detected for channel 3 1 = Bus error response detected for channel 3 NOTE: This bit-field is auto-clear and it is initialized to '0' as soon as a new transfer is started. | 0x0 |
| 10 | R | DMA_BUS_ERR2 | 0 = No bus error response is detected for channel 2 1 = Bus error response detected for channel 2 NOTE: This bit-field is auto-clear and it is initialized to '0' as soon as a new transfer is started. | 0x0 |
| 9 | R | DMA_BUS_ERR1 | 0 = No bus error response is detected for channel 1 1 = Bus error response detected for channel 1 NOTE: This bit-field is auto-clear and it is initialized to '0' as soon as a new transfer is started. | 0x0 |
| 8 | R | DMA_BUS_ERR0 | 0 = No bus error response is detected for channel 0 1 = Bus error response detected for channel 0 NOTE: This bit-field is auto-clear and it is initialized | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| | | | to '0' as soon as a new transfer is started. | |
| 7 | R | DMA_IRQ_CH7 | 0 = IRQ on channel 7 is not set 1 = IRQ on channel 7 is set | 0x0 |
| 6 | R | DMA_IRQ_CH6 | 0 = IRQ on channel 6 is not set 1 = IRQ on channel 6 is set | 0x0 |
| 5 | R | DMA_IRQ_CH5 | 0 = IRQ on channel 5 is not set 1 = IRQ on channel 5 is set | 0x0 |
| 4 | R | DMA_IRQ_CH4 | 0 = IRQ on channel 4 is not set 1 = IRQ on channel 4 is set | 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 878: DMA_CLEAR_INT_REG (0x50040908)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 7 | R0/W | DMA_RST_IRQ_CH7 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 7 ; writing a 0 will have no effect | 0x0 |
| 6 | R0/W | DMA_RST_IRQ_CH6 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 6 ; writing a 0 will have no effect | 0x0 |
| 5 | R0/W | DMA_RST_IRQ_CH5 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 5 ; writing a 0 will have no effect | 0x0 |
| 4 | R0/W | DMA_RST_IRQ_CH4 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 4 ; writing a 0 will have no effect | 0x0 |
| 3 | R0/W | DMA_RST_IRQ_CH3 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 3 ; writing a 0 will have no effect | 0x0 |
| 2 | R0/W | DMA_RST_IRQ_CH2 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 2 ; writing a 0 will have no effect | 0x0 |
| 1 | R0/W | DMA_RST_IRQ_CH1 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 1 ; writing a 0 will have no effect | 0x0 |
| 0 | R0/W | DMA_RST_IRQ_CH0 | Writing a 1 will reset the status bit of DMA_INT_STATUS_REG for channel 0 ; writing a 0 will have no effect | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 879: DMA_INT_MASK_REG (0x5004090C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|---|-------|
| 7 | R/W | DMA_IRQ_ENABLE_7 | 0 = disable interrupts on channel 7 1 = enable interrupts on channel 7 | 0x0 |
| 6 | R/W | DMA_IRQ_ENABLE_6 | 0 = disable interrupts on channel 6 1 = enable interrupts on channel 6 | 0x0 |
| 5 | R/W | DMA_IRQ_ENABLE_5 | 0 = disable interrupts on channel 5 1 = enable interrupts on channel 5 | 0x0 |
| 4 | R/W | DMA_IRQ_ENABLE_4 | 0 = disable interrupts on channel 4 1 = enable interrupts on channel 4 | 0x0 |
| 3 | R/W | DMA_IRQ_ENABLE_3 | 0 = disable interrupts on channel 3 1 = enable interrupts on channel 3 | 0x0 |
| 2 | R/W | DMA_IRQ_ENABLE_2 | 0 = disable interrupts on channel 2 1 = enable interrupts on channel 2 | 0x0 |
| 1 | R/W | DMA_IRQ_ENABLE_1 | 0 = disable interrupts on channel 1 1 = enable interrupts on channel 1 | 0x0 |
| 0 | R/W | DMA_IRQ_ENABLE_0 | 0 = disable interrupts on channel 0 1 = enable interrupts on channel 0 | 0x0 |

42.28 Charger Registers

Table 880: Register map CHARGER

| Address | Register | Description |
|------------|--------------------------------|---|
| 0x50040400 | CHARGER_CTRL_REG | Charger main control register |
| 0x50040404 | CHARGER_TEST_CTL_REG | Charger test control register |
| 0x50040408 | CHARGER_STATUS_REG | Charger main status register |
| 0x5004040C | CHARGER_VOLTAGE_PARAM_REG | Charger voltage settings register |
| 0x50040410 | CHARGER_CURRENT_PARAM_REG | Charger current settings register |
| 0x50040414 | CHARGER_TEMPSET_PARAM_REG | Charger battery temperature settings register |
| 0x50040418 | CHARGER_PRE_CHARGE_TIMER_REG | Maximum pre-charge time limit register |
| 0x5004041C | CHARGER_CC_CHARGE_TIMER_REG | Maximum CC-charge time limit register |
| 0x50040420 | CHARGER_CV_CHARGE_TIMER_REG | Maximum CV-charge time limit register |
| 0x50040424 | CHARGER_TOTAL_CHARGE_TIMER_REG | Maximum total charge time limit register |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------------------|--|
| 0x50040428 | CHARGER_JEITA_V_CHARGE_REG | JEITA-compliant Charge voltage settings register |
| 0x5004042C | CHARGER_JEITA_V_PRECHARGE_REG | JEITA-compliant Pre-Charge voltage settings register |
| 0x50040430 | CHARGER_JEITA_V_REPLENISH_REG | JEITA-compliant Replenish settings register |
| 0x50040434 | CHARGER_JEITA_V_OVP_REG | JEITA-compliant OVP settings register |
| 0x50040438 | CHARGER_JEITA_CURRENT_REG | JEITA-compliant current settings register |
| 0x5004043C | CHARGER_VBAT_COMP_TIMER_REG | Main Vbat comparator timer register |
| 0x50040440 | CHARGER_VOVP_COMP_TIMER_REG | Vbat OVP comparator timer register |
| 0x50040444 | CHARGER_TDIE_COMP_TIMER_REG | Die temperature comparator timer register |
| 0x50040448 | CHARGER_TBAT_MONITOR_TIMER_REG | Battery temperature monitor interval timer |
| 0x5004044C | CHARGER_TBAT_COMP_TIMER_REG | Battery temperature (main) comparator timer |
| 0x50040450 | CHARGER_THOT_COMP_TIMER_REG | Battery temperature comparator timer for "Hot" zone |
| 0x50040454 | CHARGER_PWR_UP_TIMER_REG | Charger power-up (settling) timer |
| 0x50040458 | CHARGER_STATE IRQ_MASK_REG | Mask register of Charger FSM IRQs |
| 0x5004045C | CHARGER_ERROR IRQ_MASK_REG | Mask register of Charger Error IRQs |
| 0x50040460 | CHARGER_STATE IRQ_STATUS_REG | Status register of Charger FSM IRQs |
| 0x50040464 | CHARGER_ERROR IRQ_STATUS_REG | Status register of Charger Error IRQs |
| 0x50040468 | CHARGER_STATE IRQ_CLR_REG | Interrupt clear register of Charger FSM IRQs |
| 0x5004046C | CHARGER_ERROR IRQ_CLR_REG | Interrupt clear register of Charger Error IRQs |

Table 881: CHARGER_CTRL_REG (0x50040400)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------------|--|-------|
| 27:22 | R | EOC_INTERVAL_C HECK_TIMER | <p>The specific bit-field determines the current state of the timer used to periodically check the End-of-Charge signal, as soon as the Charger's FSM is either in CC_CHARGE or CV_CHARGE state. Thus, as soon as the Charger's FSM enters the CC_CHARGE state:</p> <ul style="list-style-type: none"> - The timer starts increasing when a positive edge detection on End-of-Charge signal occurs. - It keeps increasing until reaching the | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------------|---|-------|
| | | | <p>programmed EOC_INTERVAL_CHECK_THRES value, if and only if there is no detection of a negative edge on End-of-Charge signal. If this happens, the timer resets and starts over with a new End-of-Charge positive edge.</p> <ul style="list-style-type: none"> - The timer also resets after having reached its programmed threshold or when the Charger's FSM next state is END_OF_CHARGE. This happens only after having found End-of-Charge signal asserted for 4 consecutive checks and provided that the specific signal has not de-asserted during the timer's interval. <p>Note: It must be noted that out of these two states, the specific timer is kept to zero. It is also noted that this timer runs at the 1Mhz clock of the Charger's block and its value always ranges from 0 to the EOC_INTERVAL_CHECK_THRES value set in the respective bit-field of CHARGER_CTRL_REG.</p> | |
| 21:16 | R/W | EOC_INTERVAL_CHECK_THRES | <p>This bit-field determines the periodic interval of checking the End-of-Charge signal, when the Charger's FSM is either in CC_CHARGE or in CV_CHARGE state. To implement this, a dedicated timer has been used, counting from zero up to the value programmed into this bit-field (see also EOC_INTERVAL_CHECK_TIMER field's description).</p> <p>As soon as this timer reaches the programmed value, the End-of-Charge signal is sampled and depending on its status (high or low), another counter, keeping the number of consecutive End-of-Charge events, is increased or not. See also the description of the EOC_DEBOUNCE_CNT bit-field of CHARGER_STATUS_REG, for this counter.</p> <p>Note: The specific bit-field should always be programmed to a non-zero value.</p> | 0x3F |
| 15 | R/W | REPLENISH_MODE | <p>When this bit-field is set and the Charger's FSM is in the BYPASSED state (thus, in Bypass mode), the internal multiplexer inside the digital part of the charger selects the Replenish, instead of the Pre-charge setting to be driven to the main Vbat comparator of the Charger's analogue circuitry. By this way, SW can read the respective analogue comparator's output in CHARGER_STATUS_REG (bit-field MAIN_VBAT_COMP_OUT), after the battery's voltage has reached the End-of-Charge level, and determine if the battery voltage has dropped below the Replenish level, re-starting the battery charging accordingly.</p> <p>Note: When the Charger's FSM is active and operational, this bit-field is don't care and the FSM determines which level (Pre-charge or Replenish) will be selected and driven to the analogue, depending on the current state. It is also noted that the supported Pre-charge and Replenish levels can be viewed in the respective bit-fields defined in CHARGER_VOLTAGE_PARAM_REG register.</p> | 0x0 |
| 14 | R/W | PRE_CHARGE_MODE | When set, this bit-field enables a signal of the same name with the bit-field, driven from the Charger's digital part towards the analogue circuitry, in order to determine the current in Pre-Charge mode. If the | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------|---|-------|
| | | | <p>Charger's FSM is active and operational, the specific bit-field is don't care. Hence, it is considered only when the Charger's FSM has reached the BYPASSED state (thus, in Bypass mode).</p> <p>With the Charger's FSM being bypassed, SW should take over control and set the specific bit-field, in order to deliver the Pre-Charge instead of the normal Charge current to the Charger's analogue circuitry, during the Pre-Charge phase.</p> <p>Note: See also the description of CHARGER_CURRENT_PARAM_REG register for the Pre-Charge and normal Charge current levels supported.</p> | |
| 13 | R/W | CHARGE_LOOP_HOLD | <p>When set, this bit-field disables charging, provided that the Charger's FSM has switched to the BYPASSED state. This is possible only by setting the CHARGER_BYPASS bit-field of this register.</p> <p>Thus, as soon as the Charger's FSM is bypassed, the respective signal driven by the FSM is overruled by this bit-field, making the analogue part of the Charger controllable also in this mode. If the Charger's FSM is not bypassed, this bit-field is don't care.</p> | 0x1 |
| 12 | R/W | JEITA_SUPPORT_DISABLED | <p>0 = Charger's JEITA FSM monitoring the battery temperature checks also if battery temperature is in the Warm or Cool zones.</p> <p>In that case, it updates accordingly all the Charger's voltage levels (Charge, Pre-Charge, Replenish and OVP) programmed in CHARGER_VOLTAGE_PARAM_REG, as well as the charge and pre-charge current settings of CHARGER_CURRENT_PARAM_REG, depending on the temperature zone determined by the analogue circuitry of the Charger (see also the JEITA registers of the Charger's register file for the Voltage/Current levels in Warm and Cool temperature zones).</p> <p>1 = Charger's JEITA FSM monitoring the battery temperature checks only if battery temperature is either in the Hot or Cold zones. In that case, it notifies the main Charger FSM to stop charging automatically, when in Hot zone. The same will happen also for the case of Cold, unless the NTC_LOW_DISABLE bit-field of CHARGER_CTRL_REG is set.</p> <p>Note : It is not recommended to have the specific bit-field kept to '0' (and thus the JEITA support enabled), if at the same time the bit-field TBAT_PROT_ENABLE of the same register is also '0'. Thus, JEITA support should be coupled with the Battery's temperature protection.</p> | 0x0 |
| 11:10 | R/W | TBAT_MONITOR_MODE | <p>Battery temperature pack monitoring modes, according to the following encoding:</p> <p>00 = Battery temperature state checked and updated once, as soon as the charger is powered-up and settled.</p> <p>01 = Battery temperature state checked periodically, depending on</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------------|--|-------|
| | | | <p>TBAT_MON_TIMER_REG.TBAT_MON_INTERVAL and provided that Charger has been powered-up and charger's FSM is enabled.</p> <p>10 = Battery temperature state checked periodically depending on TBAT_MON_TIMER_REG.TBAT_MON_INTERVAL , provided that Charger is powered-up and regardless if the Charger's FSM is enabled or not. Hence, this mode can be effective regardless of the state of CHARGE_START bit-field of CHARGER_CTRL_REG.</p> <p>11 = When selected, it freezes the Battery temperature monitor FSM, as soon as the latter reaches the CHECK_IDLE state (see also CHARGER_STATUS_REG.CHARGER_JEITA_ST ATE bit-field's description for the states of this FSM). In this mode, the monitoring of Battery temperature is possible only by checking the status of TBAT_HOT_COMP_OUT and MAIN_TBAT_COMP_OUT bit-fields of CHARGER_STATUS_REG, thus by letting SW take over monitoring. This setting may be used in conjunction with Bypass mode (by setting CHARGER_BYPASS of CHARGER_CTRL_REG), so that both charging and battery temperature status monitoring are controlled by SW.</p> | |
| 9 | R/W | CHARGE_TIMERS_HALT_ENABLE | <p>0 = Charge timeout timers continue running when charging is disabled because of a Die or of a Battery temperature error.</p> <p>1 = Charge timeout timers are halted in case of a Die or of a Battery temperature error.</p> <p>In that case, the global charge timer is stopped as soon as the Charger's FSM moves to TDIE_PROT or TBAT_PROT state. Also, either the Pre-Charge, the CC_CHARGE or the CV_CHARGE timer is also stopped, depending on the charging state of the FSM when the Die/Battery temperature error has been detected.</p> | 0x1 |
| 8 | R/W | - | | 0x0 |
| 7 | R/W | NTC_LOW_DISABLE | <p>0 = Charging is disabled when the battery temperature is found to have reached the "COLD" region. Therefore, the Charger's FSM moves directly to "TBAT_PROT" error and generates an IRQ to notify the system accordingly, in case the respective IRQ mask bit of CHARGER_ERROR_IRQ_MASK_REG is set. Also, CHARGER_ERROR_IRQ_STATUS_REG.TBAT_ERROR_IRQ field is updated accordingly.</p> <p>1 = Charging is allowed to continue, even when the battery temperature pack reaches the "COLD" region. Consequently, the FSM continues charging and no battery temperature error event is generated.</p> | 0x0 |
| 6 | R/W | TBAT_PROT_ENABLE | <p>0 = Battery temperature protection disabled</p> <p>1 = Battery temperature protection enabled.</p> <p>Charging will be stopped in case Battery temperature reaches "Hot" zone. It will also be disabled when reaching "Cold" zone, provided that CHARGER_CTRL_REG.NTC_LOW_DISABLE is</p> | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|--|-------|
| | | | not set. This is handled by the Charger's FSM, which moves directly to the respective error state (TBAT_PROT), also generating an Error IRQ if the respective IRQ mask bit is set (see also CHARGER_ERROR_IRQ_MASK_REG). | |
| 5 | R/W | TDIE_ERROR_RESUME | <p>0 = FSM will not resume from a Die temperature error. Consequently, its state will be staying to "TDIE_PROT", for as long as this bit-field is kept low, regardless of the status of the die temperature comparator. Also, disabling the specific bit-field will reset the Die temperature error debounce counter, when the Charger's FSM is in TDIE_PROT state (so when a Die temperature error has been already detected) and the specific counter will remain frozen to 0 until the TDIE_ERROR_RESUME bit-field is set (see also the TDIE_ERROR_DEBOUNCE_CNT bit-field of CHARGER_STATUS_REG).</p> <p>1 = FSM will resume from a Die temperature error, as soon as the respective analogue compator confirms that die temperature is again below the maximum allowed level.</p> <p>It is noted that the maximum Die temperature level is programmable via the CHARGER_TEMPSET_PARAM_REG register's respective bit-field (T_DIE_MAX).</p> | 0x1 |
| 4 | R/W | TDIE_PROT_ENABLE | <p>0 = Die temperature protection is disabled, thus charging will not be disabled by the Charger's FSM in case of a Die temperature error.</p> <p>1 = Die temperature protection is enabled, thus the Charger's FSM will move to "TDIE_PROT" state, disabling charging at the same time.</p> <p>It is noted that the Die temperature error event will be logged in the respective status bit of CHARGER_IRQ_ERROR_STATUS_REG and an IRQ will be generated, if and only if the corresponding mask bit of CHARGER_IRQ_MASK_REG is set.</p> | 0x1 |
| 3 | R/W | CHARGER_RESUME | <p>0 = Charger's FSM is not able to resume from a charge timeout error or a Vbat OVP (Over-Voltage Protection) error. Consequently, FSM stays in "ERROR" state.</p> <p>1 = Charger's FSM will resume from a charge timeout or from an OVP error, thus its state will move from "ERROR" to "DISABLED" state, so that the charge cycle starts-over.</p> <p>It is noted that in the case of a Vbat OVP error, the FSM will leave "ERROR" state, as soon as the Vbat comparator for the OVP level shows that Vbat is again OK (so lower than the OVP setting).</p> | 0x1 |
| 2 | R/W | CHARGER_BYPASS | <p>0 = Charger's FSM is active and running, notifying SW upon switching between its states</p> <p>1 = Charger's FSM is bypassed, so its state stays to "BYPASS", so SW should take over the monitoring of the battery voltage and control of the charger.</p> | 0x0 |
| 1 | R/W | CHARGE_START | <p>0 = Charger's FSM is disabled, FSM stays at "DISABLED" state</p> <p>1 = Charger's FSM is enabled, so FSM's state can</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|--|-------|
| | | | move from DISABLED to the actual charge states, starting from "PRE_CHARGE". | |
| 0 | R/W | CHARGER_ENABL E | <p>0 = Charger's analogue circuitry is powered-down 1 = Charger's analogue circuitry is being powered-up and will be available after a certain settling time (in ms).</p> <p>As soon as this bit-field is set, the Charger's FSM waits for this settling time, before proceeding into DISABLED state, where it checks the Vbat level, as well as the Die temperature and the Battery temperature states. This is mandatory, before the actual charging begins, so before the FSM moves to PRE_CHARGE state.</p> <p>It is finally noted that the settling time is configurable via CHARGER_PWR_UP_TIMER_REG, counting with the 1Khz clock.</p> <p>Note: The Charger clocks must have been enabled first, by setting the CLK_SYS_REG[CLK_CHG_EN] bit-field to '1', in order to let the FSM proceed.</p> | 0x0 |

Table 882: CHARGER_TEST_CTRL_REG (0x50040404)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|-------------|-------|
| 16:14 | - | - | | 0x0 |
| 13:9 | - | - | | 0xF |
| 8:4 | - | - | | 0x8 |
| 3:0 | - | - | | 0x8 |

Table 883: CHARGER_STATUS_REG (0x50040408)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------------------|--|-------|
| 29:27 | R | OVP_EVENTS_DEB OUNCE_CNT | <p>The specific bit-field returns the consecutive number of times Vbat has exceeded the programmed Over-Voltage Protection (OVP) level. It is used to determine when the Charger's FSM will exit any of the charging states (PRE/CC/CV_CHARGE) and will switch to the ERROR state due to an OVP error. This will happen as soon as the respective counter of OVP events reaches or exceeds a fixed number (4), similar to the approach adopted in the End-of-Charge and Die Temperature debouncing mechanisms.</p> <p>The specific counter increases only while the Charger's FSM is in any of the three charging states, the Vbat OVP interval check timer has reached the threshold set and when Vbat OVP comparator's output is asserted.</p> <p>Note 1 : By default, as soon as the counter reaches 4, the FSM will switch to the ERROR state and the counter will reset again. Thus, in that case the specific counter ranges from 0 to 4 and vice-versa. However, if the monitoring of Vbat OVP comparator's state is less frequent than 5 (4+1)</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|--|-------|
| | | | <p>times the CHARGER_OVP_COMP_TIMER_REG[OVP_INTEGRAL_CHECK_THRES] and Vbat has exceeded the OVP voltage level based on the comparator's output signal, then this counter will exceed 4 and may overflow.</p> <p>This will not harm, however, the detection of the OVP event, as it only increases the number of OVP event occurrences by the debounce timer, until the OVP comparator timer's settling time has expired. Thus, the Charger FSM will again switch to ERROR when the counter has reached or exceeded 4 (bit [2] of OVP_EVENTS_DEBOUNCE_CNT is set) and the OVP comparator's timer has expired.</p> <p>Note 2: See also the OVP_INTERVAL_CHECK_TIMER, OVP_INTERVAL_CHECK_THRES of CHARGER_OVP_COMP_TIMER_REG, for the debouncing mechanism of the Vbat OVP comparator's output.</p> | |
| 26:24 | R | EOC_EVENTS_DEBOUNCE_CNT | <p>The specific bit-field returns the number of times the End-of-Charge signal has been consecutively found to be high. It is used to determine when the Charger's FSM will switch from CV_CHARGE to END_OF_CHARGE state, implementing a debounce mechanism on End-of-Charge signal, coming from the analogue circuitry of the Charger towards the FSM.</p> <p>The specific counter, running with the Charger's 1Mhz clock:</p> <ul style="list-style-type: none"> - Increases after detecting that the End-of-Charge signal is high when the respective interval for the End-of-Charge check expires. This actually happens after having detected a positive edge on End-of-Charge signal, since only after that is it possible for the interval timer to start ticking. - Resets to zero when End-of-Charge is seen low when the interval timer has expired or when an End-of-Charge negative edge is seen before the timer's expiration, starting-over. - Does not count if End-of-Charge signal is seen high and either the CV_MODE signal (also driven by the analogue circuitry) or the End-of-Charge signal of the previous clock cycle is seen low. - Is reset when the Charger's FSM is not in either the CC_CHARGE or the CV_CHARGE state or after having reached "100"(4). This is the threshold after which the End-of-Charge signal is considered stable by the Charger's FSM, to switch to the END_OF_CHARGE state. Thus, in practice, the specific counter (and bit-field) ranges between 0 and 4. <p>Note: See also the EOC_INTERVAL_CHECK_TIMER/THRES bit-fields of CHARGER_CTRL_REG.</p> | 0x0 |
| 23:21 | R | TDIE_ERROR_DEBOUNCE_CNT | The specific bit-field returns the consecutive number of times the Die temperature is seen either above (for the case of an error) or below (for the case of recovering from an error) the set Die temperature level. This is performed by a counter, | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|--|-------|
| | | | <p>which is increased:</p> <ul style="list-style-type: none"> - Each time the Die temperature comparator shows that Die temperature exceeds the set level, and while charging is active, provided that Die temperature protection is enabled. If, however, the CHARGER_CTRL_REG.TDIE_PROT_ENABLE bit-field is not set, the counter is reset and stays frozen to zero. - Each time the Die temperature comparator shows that Die temperature is again below the set level, and while the FSM is in the Die temperature protection error state (TDIE_PROT) and the TDIE_ERROR_RESUME bit-field of CHARGER_CTRL_REG is set. If the specific bit-field is not set, the debounce counter is reset to 0 and it is kept frozen until the FSM is again enabled to resume from Die temperature errors. <p>If the Die temperature comparator of the Charger's analogue circuitry shows that temperature has exceeded the programmed level for four consecutive times and charging is active, the Charger's FSM considers this as a Die temperature error and moves to the TDIE_PROT state, resetting the timer at the same time and of course halting charging.</p> <p>To recover from this state and resume charging, the FSM needs to see that Die temperature is below the programmed level for four consecutive times, again, provided that the TDIE_ERROR_RESUME bit-field of CHARGER_CTRL_REG is set. As soon as this happens, the error counter is again reset and the Charger's FSM resumes, by moving to PRE_CHARGE state. Consequently, the counter's value always ranges from 0 to 4.</p> <p>Note: When the Charger's FSM is in BYPASSED state, then this bit-field is reset and kept frozen to zero. Consequently, the number of times Die temperature has exceeded the pre-programmed threshold should be determined by SW.</p> | |
| 20:18 | R | CHARGER_JEITA_STATE | <p>Returns the state of the Charger's JEITA FSM. This FSM is used to update the state of the battery temperature pack, depending on the value programmed in CHARGER_CTRL_REG.TBAT_MONITOR_MODE bit-field. The encoding of the states is as follows:</p> <ul style="list-style-type: none"> 0x0 = CHECK_IDLE 0x1 = CHECK_THOT 0x2 = CHECK_TCOLD 0x3 = CHECK_TWARM 0x4 = CHECK_TCOOL 0x5 = CHECK_TNORMAL 0x6 = UPDATE_TBAT <p>The FSM initially is in CHECK_IDLE state and starts checking the battery's temperature by visiting the states that check for the respective temperature area (Hot, Cold, Warm, Cool, Normal), in this order. If the battery temperature is found to be in one of the aforementioned zones, it directly moves to UPDATE_TBAT state, to update the battery</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| | | | temperature's state and notify the main FSM of the Charger about the battery temperature status, before returning to the CHECK_IDLE state. A Charger State IRQ will also be generated upon refreshing the battery temperature status (see also the description of CHARGER_STATE_IRQ_MASK_REG register). | |
| 17:14 | R | CHARGER_STATE | <p>Indicating the state of the Charger's main FSM, based on the following encoding:</p> <p>0x0 = POWER_UP (Charger's power-up not yet set)</p> <p>0x1 = INIT (Charger is being power-up, FSM waiting for the analogue to settle)</p> <p>0x2 = DISABLED (Charger powered-up but charging not yet started)</p> <p>0x3 = PRE_CHARGE (Pre-Charge state)</p> <p>0x4 = CC_CHARGE (Constant Current state)</p> <p>0x5 = CV_CHARGE (Constant Voltage state)</p> <p>0x6 = END_OF_CHARGE (End-of-Charge state)</p> <p>0x7 = TDIE_PROT (Die temperature protection state, visited when Die temperature limit is exceeded)</p> <p>0x8 = TBAT_PROT (Battery temperature protection state, visited when Battery temperature is either COLD or HOT)</p> <p>0x9 = BYPASSED (Bypassed state, visited only when the FSM is bypassed and SW takes over control)</p> <p>0xA = ERROR (Error state, visited when a charge time-out occurs or in the case of Vbat exceeding over-voltage level)</p> | 0x0 |
| 13:9 | R | TBAT_STATUS | <p>Battery pack temperature status, according to the following ("1-Hot"-like) encoding:</p> <p>0x1 : Battery temperature in COLD zone (default)</p> <p>0x2 : Battery temperature in COOL zone</p> <p>0x4 : Battery temperature in NORMAL zone (above COOL and below WARM zones)</p> <p>0x8 : Battery temperature in WARM zone</p> <p>0x10 : Battery temperature in HOT zone</p> <p>It is noted that, according to the JEITA standard (supported if the JEITA_SUPPORT_DISABLED bit-field of CHARGER_CTRL_REG is not set), if the battery pack temperature is in the "HOT" zone, charging will always be stopped. The same will happen also for the case of the COLD zone, unless the "NTC_LOW_DISABLE" bit-field of CHARGER_CTRL_REG is set. In that case, charging will be continued.</p> <p>It is finally noted that only the aforementioned values are available for this bit-field, since it is 1-Hot encoding based. Not more than 1 bit can be high at the same time, since this would mean that battery temperature is at two different temperature zones concurrently.</p> | 0x1 |
| 8 | R | MAIN_TBAT_COMP_OUT | Returns the status of the main battery temperature comparator. This comparator by default checks if | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|---|-------|
| | | | <p>the battery temperature is in the Cold zone. However, if JEITA support is enabled and battery temperature is found to not be in either the Hot or the Cold zone, the same comparator is used to check for the Warm and Cool zones, as JEITA suggests.</p> <p>The specific bit-field is suggested to be used in bypass mode and when the JEITA support is disabled (so when the battery temperature is checked against the Hot and the Cold zones). In that case, the comparator checks the battery temperature against the Cold level and its status can be as follows:</p> <p>0 = Battery temperature pack is found to be below the Cold level, so in the non-allowed Cold temperature zone. Thus, charging will be disabled, provided that the NTC_LOW_DISABLE bit-field of CHARGER_CTRL_REG is not set.</p> <p>1 = Battery temperature pack is found to be above the non-allowed Cold temperature zone. Thus, charging will be continued, provided that battery temperature will not be in the Hot zone as well.</p> <p>When the Charger's main FSM is active and JEITA is enabled, the Charger's digital block takes over and controls the respective comparator's output.</p> | |
| 7 | R | TBAT_HOT_COMP_OUT | <p>Returns the status of the battery temperature comparator dedicated to the Hot temperature zone.</p> <p>0 = Battery temperature pack is found to be below the Hot zone</p> <p>1 = Battery temperature pack is found to be in the non-allowed Hot temperature zone. Thus, charging will be disabled, provided that battery temperature protection is enabled.</p> | 0x0 |
| 6 | R | TDIE_COMP_OUT | <p>0 = Die temperature is found to be below the programmed level, set in CHARGER_TEMPSET_PARAM_REG.TDIE_SET level (normal operation)</p> <p>1 = Die temperature is found to be above the set level.</p> <p>Charging will be disabled if Die temperature protection is enabled and the Die temperature is found to be above the set level four consecutive times (see also TDIE_ERROR_DEBOUNCE_CNT bit-field). In that case, the Charger's FSM will also move the respective error state (TDIE_PROT) and an IRQ may be generated, if the respective mask bit of CHARGER_ERROR_IRQ_MASK_REG is set.</p> | 0x0 |
| 5 | R | VBAT_OVP_COMP_OUT | <p>0 = Vbat has not exceeded the Over-Voltage Protection (OVP) voltage limit, according to the respective analogue comparator's output.</p> <p>1 = Vbat is found to have exceeded the OVP voltage setting, thus charging should be disabled.</p> <p>The OVP voltage settings are defined in CHARGER_VOLTAGE_PARAM_REG.V_OVP (for the Normal battery temperature zone), as well as in CHARGER_JEITA_V_OVP_REG (for Cool and Warm temperature zones, to comply with JEITA).</p> | 0x0 |
| 4 | R | MAIN_VBAT_COMP | This bit-field reflects the status of the main Vbat | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------|---|-------|
| | | _OUT | <p>comparator residing in the analogue circuitry of the Charger.</p> <p>This comparator is used to check Vbat against either the Pre-Charge or the Replenish voltage level, depending on what is driven by the Charger's digital block.</p> <p>Thus, when the FSM is active, the comparator gets as reference the Replenish setting as soon as the FSM has reached the END_OF_CHARGE state. Otherwise, the Pre-Charge voltage setting is driven, including the Bypass mode.</p> <p>According to the above, the encoding is as follows for the case the comparator compares Vbat against the Pre-Charge level:</p> <p>0 = Vbat has not exceeded the set Pre-Charge voltage level.</p> <p>1 = Vbat has reached or exceeded the set Pre-Charge voltage level.</p> <p>For the case the comparator compares against the Replenish level (when the FSM has reached the END_OF_CHARGE state, so when the charging has been completed), the encoding is as follows:</p> <p>0 = Vbat has dropped below the set Replenish level, so charging will re-start and the FSM will move to the PRE_CHARGE state.</p> <p>1 = Vbat is still greater or equal to the set Replenish level, thus charging remains in hold and the FSM in END_OF_CHARGE state.</p> | |
| 3 | R | END_OF_CHARGE | <p>0 = Actual charge current is above the current level programmed in I-END_OF_CHARGE field of CHARGER_CURRENT_PARAM_REG (or charger is off)</p> <p>1 = Actual charge current is below the current level programmed in I-END_OF_CHARGE bit-field of CHARGER_CURRENT_PARAM_REG.</p> | 0x0 |
| 2 | R | CHARGER_CV_MODE | <p>0 = Charger's voltage loop not in regulation (or Charger is off)</p> <p>1 = Charger's Constant Voltage (CV) mode active, voltage loop in regulation</p> | 0x0 |
| 1 | R | CHARGER_CC_MODE | <p>0 = Charger's Current loop not in regulation (or Charger is off)</p> <p>1 = Charger's Constant Current (CC) mode active, current loop in regulation</p> | 0x0 |
| 0 | R | CHARGER_IS_POWERED_UP | <p>0 = Charger is either off or it is being powered-on but the analogue circuitry is not yet settled. The charger's main FSM is either in POWER_UP or INIT states.</p> <p>1 = Charger is powered-up, so its analogue circuitry should now be settled. The Charger's FSM has left both power-up states (POWER_UP, INIT), so charging can start.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 884: CHARGER_VOLTAGE_PARAM_REG (0x5004040C)

| Bit | Mode | Symbol | Description | Reset | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-------|---------|-------------|--|-------|---------|---|---------|---|---------|---|---------|---|---------|---|---------|---|---------|---|---------|---|---------|---|---------|----|---------|----|---------|----|---------|----|---------|----|---------|----|---------|----|---------|----|---------|------|
| 23:18 | R/W | V_OVP | <p>This bit-field determines the VBAT Over-voltage protection limit. This Over-voltage protection level is used by the Charger's analogue circuitry and specifically by a dedicated comparator, the output of which is sampled by the digital block of the Charger. As soon as VBAT is detected to have reached or exceeded this level, the Charger's FSM moves to ERROR state, interrupting charging. If the respective Error IRQ mask bit is set, an Error IRQ pulse will be also generated.</p> <p>Regarding the actual range of supported values for this bit-field, see the the description of V_CHARGE bit-field of this register.</p> | 0x32 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 17:12 | R/W | V_REPLENISH | <p>This bit-field determines the absolute value (in V) of the Replenish voltage threshold. As soon as charging has been completed and the Charger's FSM has reached the END_OF_CHARGE state, the respective analogue comparator of the Charger compares VBAT with the Replenish level. If VBAT is found to have dropped below this level, charging should start-over again and in that case, the FSM moves again to the PRE_CHARGE state.</p> <p>Regarding the supported Replenish voltage levels, see the description of V_CHARGE bit-field.</p> | 0x21 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11:6 | R/W | V_PRECHARGE | <p>This bit-field determines the voltage level at which the battery is considered as Pre-charged and therefore the Charger's FSM should move to the CC_CHARGE state, entering the Constant Current charging phase.</p> <p>Regarding the supported Pre-Charge voltage levels, see also the description of V_CHARGE bit-field of this register.</p> | 0x8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5:0 | R/W | V_CHARGE | <p>This bit-field determines the charge voltage levels supported. The supported levels are determined according to the following encoding:</p> <table> <tbody> <tr><td>0</td><td>: 2.80V</td></tr> <tr><td>1</td><td>: 2.85V</td></tr> <tr><td>2</td><td>: 2.90V</td></tr> <tr><td>3</td><td>: 2.95V</td></tr> <tr><td>4</td><td>: 3.00V</td></tr> <tr><td>5</td><td>: 3.05V</td></tr> <tr><td>6</td><td>: 3.10V</td></tr> <tr><td>7</td><td>: 3.15V</td></tr> <tr><td>8</td><td>: 3.20V</td></tr> <tr><td>9</td><td>: 3.25V</td></tr> <tr><td>10</td><td>: 3.30V</td></tr> <tr><td>11</td><td>: 3.35V</td></tr> <tr><td>12</td><td>: 3.40V</td></tr> <tr><td>13</td><td>: 3.45V</td></tr> <tr><td>14</td><td>: 3.50V</td></tr> <tr><td>15</td><td>: 3.55V</td></tr> <tr><td>16</td><td>: 3.60V</td></tr> <tr><td>17</td><td>: 3.65V</td></tr> </tbody> </table> | 0 | : 2.80V | 1 | : 2.85V | 2 | : 2.90V | 3 | : 2.95V | 4 | : 3.00V | 5 | : 3.05V | 6 | : 3.10V | 7 | : 3.15V | 8 | : 3.20V | 9 | : 3.25V | 10 | : 3.30V | 11 | : 3.35V | 12 | : 3.40V | 13 | : 3.45V | 14 | : 3.50V | 15 | : 3.55V | 16 | : 3.60V | 17 | : 3.65V | 0x2B |
| 0 | : 2.80V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | : 2.85V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | : 2.90V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | : 2.95V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | : 3.00V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | : 3.05V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | : 3.10V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | : 3.15V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | : 3.20V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | : 3.25V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | : 3.30V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | : 3.35V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | : 3.40V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 13 | : 3.45V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 14 | : 3.50V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | : 3.55V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 16 | : 3.60V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 17 | : 3.65V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 18 : 3.70V 19 : 3.75V 20 : 3.80V 21 : 3.82V 22 : 3.84V 23 : 3.86V 24 : 3.88V 25 : 3.90V 26 : 3.92V 27 : 3.94V 28 : 3.96V 29 : 3.98V 30 : 4.00V 31 : 4.02V 32 : 4.04V 33 : 4.06V 34 : 4.08V 35 : 4.10V 36 : 4.12V 37 : 4.14V 38 : 4.16V 39 : 4.18V 40 : 4.20V 41 : 4.22V 42 : 4.24V 43 : 4.26V 44 : 4.28V 45 : 4.30V 46 : 4.32V 47 : 4.34V 48 : 4.36V 49 : 4.38V 50 : 4.40V 51 : 4.42V 52 : 4.44V 53 : 4.46V 54 : 4.48V 55 : 4.50V 56 : 4.52V 57 : 4.54V 58 : 4.56V 59 : 4.58V 60 : 4.60V 61 : 4.70V 62 : 4.80V 63 : 4.90V* It has to be noted that the specific values correspond to the normal battery temperature zone. However, the specific register field may be updated by the JEITA FSM (which checks the battery | |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | <p>temperature either once or periodically), in order to adapt the charge voltage to the battery temperature zone (see also CHARGER_CTRL_REG.TBAT_MONITOR_MODE field as well). This is valid also for the other three fields of the current register. Consequently, in that case the register returns the Charge voltage settings that abide to the JEITA requirements for the battery (either COOL, WARM or NORMAL).</p> <p>Note: Option "63" (4.90V) is not supported for V_CHARGE, V_PRECHARGE and V_REPLENISH bit-fields (and respective levels). It should be used only in the V_OVP bit-field, as the (maximum) Over-voltage protection level.</p> | |

Table 885: CHARGER_CURRENT_PARAM_REG (0x50040410)

| Bit | Mode | Symbol | Description | Reset | | | | | | | | | | | | | | | | |
|-------------|------|--------------------|---|------------|------------|----------|------------|-----------|-----------|-----------|-----------|------------|-------------|-------------|-------------|-----------|-------------|-------------|-------------|-----|
| 15 | R/W | I_EOC_DOUBLE_RANGE | <p>When set, the specific bit-field enables an increase of the (%) range of End-of-Charge current setting. Consequently, the default lower and upper limits of 4% of I_CHARGE (value 0x0 of I_END_OF_CHARGE bit-field) and 16% (value 0x7 of the same bit-field) are increased to 8.8% and 35.2% respectively, as soon as the I_EOC_DOUBLE_RANGE field is set.</p> | 0x0 | | | | | | | | | | | | | | | | |
| 14:12 | R/W | I_END_OF_CHARGE | <p>End-of-Charge current setting, ranging from 4%("000") to 16% ("111") of the charge current set, with a step size of 1.5% for the first 4 settings and 2% for the last 4 settings, as follows (when I_EOC_DOUBLE_RANGE = 0):</p> <table style="margin-left: 20px;"> <tr><td>000 : 4%</td></tr> <tr><td>001 : 5.5%</td></tr> <tr><td>010 : 7%</td></tr> <tr><td>011 : 8.5%</td></tr> <tr><td>100 : 10%</td></tr> <tr><td>101 : 12%</td></tr> <tr><td>110 : 14%</td></tr> <tr><td>111 : 16%</td></tr> </table> <p>When I_EOC_DOUBLE_RANGE = 1, the range is:</p> <table style="margin-left: 20px;"> <tr><td>000 : 8.8%</td></tr> <tr><td>001 : 12.1%</td></tr> <tr><td>010 : 15.4%</td></tr> <tr><td>011 : 18.7%</td></tr> <tr><td>100 : 22%</td></tr> <tr><td>101 : 26.4%</td></tr> <tr><td>110 : 30.8%</td></tr> <tr><td>111 : 35.2%</td></tr> </table> | 000 : 4% | 001 : 5.5% | 010 : 7% | 011 : 8.5% | 100 : 10% | 101 : 12% | 110 : 14% | 111 : 16% | 000 : 8.8% | 001 : 12.1% | 010 : 15.4% | 011 : 18.7% | 100 : 22% | 101 : 26.4% | 110 : 30.8% | 111 : 35.2% | 0x2 |
| 000 : 4% | | | | | | | | | | | | | | | | | | | | |
| 001 : 5.5% | | | | | | | | | | | | | | | | | | | | |
| 010 : 7% | | | | | | | | | | | | | | | | | | | | |
| 011 : 8.5% | | | | | | | | | | | | | | | | | | | | |
| 100 : 10% | | | | | | | | | | | | | | | | | | | | |
| 101 : 12% | | | | | | | | | | | | | | | | | | | | |
| 110 : 14% | | | | | | | | | | | | | | | | | | | | |
| 111 : 16% | | | | | | | | | | | | | | | | | | | | |
| 000 : 8.8% | | | | | | | | | | | | | | | | | | | | |
| 001 : 12.1% | | | | | | | | | | | | | | | | | | | | |
| 010 : 15.4% | | | | | | | | | | | | | | | | | | | | |
| 011 : 18.7% | | | | | | | | | | | | | | | | | | | | |
| 100 : 22% | | | | | | | | | | | | | | | | | | | | |
| 101 : 26.4% | | | | | | | | | | | | | | | | | | | | |
| 110 : 30.8% | | | | | | | | | | | | | | | | | | | | |
| 111 : 35.2% | | | | | | | | | | | | | | | | | | | | |
| 11:6 | R/W | I_PRECHARGE | <p>This bit-field determines the Pre-Charge current, in mA, ranging from 0.5 to 56mA, according to the following encoding:</p> <table style="margin-left: 20px;"> <tr><td>0 : 0.5 mA</td></tr> <tr><td>1 : 1 mA</td></tr> </table> | 0 : 0.5 mA | 1 : 1 mA | 0x3 | | | | | | | | | | | | | | |
| 0 : 0.5 mA | | | | | | | | | | | | | | | | | | | | |
| 1 : 1 mA | | | | | | | | | | | | | | | | | | | | |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 2 : 1.5mA 3 : 2 mA 4 : 2.5mA 5 : 3 mA 6 : 3.5mA 7 : 4 mA 8 : 4.5mA 9 : 5 mA 10 : 5.5mA 11 : 6 mA 12 : 6.5mA 13 : 7 mA 14 : 7.5mA 15 : 8 mA 16 : 9 mA 17 : 10 mA 18 : 11 mA 19 : 12 mA 20 : 13 mA 21 : 14 mA 22 : 15 mA 23 : 16 mA 24 : 17 mA 25 : 18 mA 26 : 19 mA 27 : 20 mA 28 : 21 mA 29 : 22 mA 30 : 23 mA 31 : 24 mA 32 : 26 mA 33 : 28 mA 34 : 30 mA 35 : 32 mA 36 : 34 mA 37 : 36 mA 38 : 38 mA 39 : 40 mA 40 : 42 mA 41 : 44 mA 42 : 46 mA 43 : 48 mA 44 : 50 mA 45 : 52 mA 46 : 54 mA 47 : 56 mA 48 : 56 mA 49 : 56 mA 50 : 56 mA | |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| | | | 51 : 56 mA 52 : 56 mA 53 : 56 mA 54 : 56 mA 55 : 56 mA 56 : 56 mA 57 : 56 mA 58 : 56 mA 59 : 56 mA 60 : 56 mA 61 : 56 mA 62 : 56 mA 63 : 56 mA | |
| 5:0 | R/W | I_CHARGE | <p>This bit-field determines the charge current range, in mA. The range is from 5mA to 560mA, according to the following encoding:</p> 0 : 5 mA 1 : 10 mA 2 : 15 mA 3 : 20 mA 4 : 25 mA 5 : 30 mA 6 : 35 mA 7 : 40 mA 8 : 45 mA 9 : 50 mA 10 : 55 mA 11 : 60 mA 12 : 65 mA 13 : 70 mA 14 : 75 mA 15 : 80 mA 16 : 90 mA 17 : 100 mA 18 : 110 mA 19 : 120 mA 20 : 130 mA 21 : 140 mA 22 : 150 mA 23 : 160 mA 24 : 170 mA 25 : 180 mA 26 : 190 mA 27 : 200 mA 28 : 210 mA 29 : 220 mA 30 : 230 mA 31 : 240 mA 32 : 260 mA | 0x6 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>33 : 280 mA 34 : 300 mA 35 : 320 mA 36 : 340 mA 37 : 360 mA 38 : 380 mA 39 : 400 mA 40 : 420 mA 41 : 440 mA 42 : 460 mA 43 : 480 mA 44 : 500 mA 45 : 520 mA 46 : 540 mA 47 : 560 mA 48 : 560 mA 49 : 560 mA 50 : 560 mA 51 : 560 mA 52 : 560 mA 53 : 560 mA 54 : 560 mA 55 : 560 mA 56 : 560 mA 57 : 560 mA 58 : 560 mA 59 : 560 mA 60 : 560 mA 61 : 560 mA 62 : 560 mA 63 : 560 mA</p> <p>Note: It has to be noted that the specific values correspond to the normal battery temperature zone. However, the specific register field may be updated by the JEITA FSM (which checks the battery temperature either once or periodically), in order to adapt the Charge current to the new battery temperature zone (see also CHARGER_CTRL_REG.TBAT_MONITOR_MODE field as well). This is valid also for the Pre-Charge current field of this register and provided that JEITA support is enabled in CHARGER_CTRL_REG. Consequently, in that case the register return the Charge current settings that abide to the JEITA requirements for the battery (either COOL, WARM or NORMAL).</p> | |

Table 886: CHARGER_TEMPSET_PARAM_REG (0x50040414)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------|---|-------|
| 26:24 | R/W | TDIE_MAX | This bit-field determines the maximum Die | 0x3 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---|-------|
| | | | temperature level limit, ranging from 0C to 130C, according to the following encoding: 000: 0 C (mainly for test purposes) 001: 50 C 010: 80 C 011: 90 C 100: 100 C 101: 110 C 110: 120 C 111: 130 C | |
| 23:18 | R/W | TBAT_HOT | This bit-field determines the battery temperature above which the charge current is zero, defining the "Hot" battery temperature zone. It ranges from minus 10C to 53C. The range is the same with the one defined in detail in TBAT_COLD bit-field. | 0x37 |
| 17:12 | R/W | TBAT_WARM | This bit-field determines the battery temperature above which the charge current is reduced, defining the "Warm" temperature zone. It ranges from minus 10C to 53C. The range is the same with the one defined in detail in TBAT_COLD bit-field. | 0x2D |
| 11:6 | R/W | TBAT_COOL | This bit-field determines the battery temperature below which the charge current is reduced, defining the "Cool" temperature zone. It ranges from minus 10C to 53C and the range is the same with the one defined in TBAT_COLD bit-field. | 0x14 |
| 5:0 | R/W | TBAT_COLD | This bit-field determines the battery temperature below which the charge current is zero, defining the "Cold" temperature zone. It ranges from minus 10C to 53C, according to the following encoding: 0 : -10 C 1 : -9 C 2 : -8 C 3 : -7 C 4 : -6 C 5 : -5 C 6 : -4 C 7 : -3 C 8 : -2 C 9 : -1 C 10 : 0 C 11 : 1 C 12 : 2 C 13 : 3 C 14 : 4 C 15 : 5 C 16 : 6 C 17 : 7 C 18 : 8 C 19 : 9 C 20 : 10 C 21 : 11 C 22 : 12 C | 0xA |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------|-------|
| | | | 23 : 13 C | |
| | | | 24 : 14 C | |
| | | | 25 : 15 C | |
| | | | 26 : 16 C | |
| | | | 27 : 17 C | |
| | | | 28 : 18 C | |
| | | | 29 : 19 C | |
| | | | 30 : 20 C | |
| | | | 31 : 21 C | |
| | | | 32 : 22 C | |
| | | | 33 : 23 C | |
| | | | 34 : 24 C | |
| | | | 35 : 25 C | |
| | | | 36 : 26 C | |
| | | | 37 : 27 C | |
| | | | 38 : 28 C | |
| | | | 39 : 29 C | |
| | | | 40 : 30 C | |
| | | | 41 : 31 C | |
| | | | 42 : 32 C | |
| | | | 43 : 33 C | |
| | | | 44 : 34 C | |
| | | | 45 : 35 C | |
| | | | 46 : 36 C | |
| | | | 47 : 37 C | |
| | | | 48 : 38 C | |
| | | | 49 : 39 C | |
| | | | 50 : 40 C | |
| | | | 51 : 41 C | |
| | | | 52 : 42 C | |
| | | | 53 : 43 C | |
| | | | 54 : 44 C | |
| | | | 55 : 45 C | |
| | | | 56 : 46 C | |
| | | | 57 : 47 C | |
| | | | 58 : 48 C | |
| | | | 59 : 49 C | |
| | | | 60 : 50 C | |
| | | | 61 : 51 C | |
| | | | 62 : 52 C | |
| | | | 63 : 53 C | |

Table 887: CHARGER_PRE_CHARGE_TIMER_REG (0x50040418)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 30:16 | R | PRE_CHARGE_TIMER | Returns the current value of the Pre-Charge timeout counter, running at a 1Hz clock. The range of the specific timer is identical to the one of the CC-Charge and the CV-Charge timers, so it may count | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| | | | up to 6 hours, ranging from 0 to MAX_PRE_CHARGE_TIME. It is reset to 0 when the Charger's FSM is either in DISABLED or in END_OF_CHARGE state. | |
| 15 | R | - | | 0x0 |
| 14:0 | R/W | MAX_PRE_CHARGE_TIME | This bit-field determines the maximum time (measured in ticks of the Charger's 1Hz clock) allowed for the Pre-Charge stage. If this is exceeded, a Pre-Charge time-out error will be captured by the Charger's control unit and its FSM will move to the respective state (ERROR). In order to exit this state and re-start charging, the CHARGER_RESUME bit-field of CHARGER_CTRL_REG must be set. Note: The specific bit-field should be always set to a non-zero value. | 0x708 |

Table 888: CHARGER_CC_CHARGE_TIMER_REG (0x5004041C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|--------|
| 30:16 | R | CC_CHARGE_TIME_R | Returns the current value of the CC-Charge timeout counter, running at a 1Hz clock. The range of the specific timer is identical to the one of the Pre-Charge and the CV-Charge timers, so it may count up to 6 hours, ranging from 0 to MAX_CC_CHARGE_TIME. It is reset to 0 when the Charger's FSM is either in DISABLED or in END_OF_CHARGE state. | 0x0 |
| 15 | R | - | | 0x0 |
| 14:0 | R/W | MAX_CC_CHARGE_TIME | This bit-field determines the maximum time (measured in ticks of the Charger's 1Hz clock) allowed for the CC (Constant Current) charging stage. If this is exceeded, a CC charge time-out error will be captured by the Charger's control unit and its FSM will move to the ERROR state. In order to exit this state and re-start charging, the CHARGER_RESUME bit-field of CHARGER_CTRL_REG must be set. Note: The specific bit-field should be always set to a non-zero value. | 0x1C20 |

Table 889: CHARGER_CV_CHARGE_TIMER_REG (0x50040420)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|---|-------|
| 30:16 | R | CV_CHARGE_TIME_R | Returns the current value of the CV-Charge timeout counter, running at a 1Hz clock. The range of the specific timer is identical to the one of the Pre-Charge and the CC-Charge timers, so it may count up to 6 hours, ranging from 0 to MAX_CV_CHARGE_TIME. It is reset to 0 when the Charger's FSM is either in DISABLED or in END_OF_CHARGE state. | 0x0 |
| 15 | R | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|--|--------|
| 14:0 | R/W | MAX_CV_CHARGE_TIME | <p>This bit-field determines the maximum time (measured in ticks of the Charger's 1Hz clock) allowed for the CV (Constant Voltage) charging stage. If this is exceeded, a CV charge time-out error will be captured by the Charger's control unit and its FSM will move to the ERROR state. In order to exit this state and re-start charging, the CHARGER_RESUME bit-field of CHARGER_CTRL_REG must be set.</p> <p>Note: The specific bit-field should be always set to a non-zero value.</p> | 0x1C20 |

Table 890: CHARGER_TOTAL_CHARGE_TIMER_REG (0x50040424)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------------|--|--------|
| 31:16 | R | TOTAL_CHARGE_TIMER | <p>Returns the current value of the overall charge timeout counter, running at a 1Hz clock. This timer has been set to 16 bits, so that it can count up to 10.5 hours, and ranges from 0 to MAX_TOTAL_CHARGE_TIME. It is reset to 0 when the Charger's FSM is either in DISABLED or in END_OF_CHARGE state.</p> | 0x0 |
| 15:0 | R/W | MAX_TOTAL_CHARGE_TIME | <p>This bit-field determines the maximum overall charging time allowed (measured in ticks of the 1Hz clock). If this is exceeded, a total charge time-out error will be captured by the Charger's controller and its FSM will move to the ERROR state. An IRQ will be also generated if the respective IRQ mask bit of CHARGER_ERROR_IRQ_MASK_REG is already set. In order to exit this state, the "CHARGER_RESUME" bit-field of CHARGER_CTRL_REG must be set, to enable the Charger's FSM switch from ERROR to DISABLED state and start-over.</p> <p>Note: The specific bit-field should be always set to a non-zero value.</p> | 0x3F48 |

Table 891: CHARGER_JEITA_V_CHARGE_REG (0x50040428)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|--|-------|
| 11:6 | R/W | V_CHARGE_TWAR_M | Charge voltage setting for the Warm battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG register. | 0x29 |
| 5:0 | R/W | V_CHARGE_TCOLD_L | Charge voltage setting for the Cool battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG register. | 0x28 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 892: CHARGER_JEITA_V_PRECHARGE_REG (0x5004042C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|--|-------|
| 11:6 | R/W | V_PRECHARGE_T_WARM | Pre-Charge voltage setting for the Warm battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG register. | 0x6 |
| 5:0 | R/W | V_PRECHARGE_TC_OOL | Pre-Charge current setting for the Cool battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG register. | 0x7 |

Table 893: CHARGER_JEITA_V_REPLENISH_REG (0x50040430)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------------|--|-------|
| 11:6 | R/W | V_REPLENISH_TW_ARM | Replenish voltage setting for the Warm battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG. | 0x1E |
| 5:0 | R/W | V_REPLENISH_TC_OOL | Replenish voltage setting for the Cool battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG. | 0x1F |

Table 894: CHARGER_JEITA_V_OVP_REG (0x50040434)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 11:6 | R/W | V_OVP_TWARM | VBAT Over-voltage Protection (OVP) setting for the Warm battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG. | 0x35 |
| 5:0 | R/W | V_OVP_TCOLD | VBAT Over-voltage Protection (OVP) setting for the Cool battery temperature zone. Regarding the range of values of this bit-field, see also the description of V_CHARGE field of CHARGER_VOLTAGE_PARAM_REG. | 0x36 |

Table 895: CHARGER_JEITA_CURRENT_REG (0x50040438)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| 23:18 | R/W | I_PRECHARGE_TW_ARM | Pre-Charge current setting for the Warm battery temperature zone. Regarding the range of values of this bit-field, see also the description of I_PRECHARGE field of CHARGER_CURRENT_PARAM_REG register. | 0x1 |
| 17:12 | R/W | I_PRECHARGE_TC_OOL | Pre-Charge current setting for the Cool battery temperature zone. Regarding the range of values of this bit-field, see also the description of I_PRECHARGE field of CHARGER_CURRENT_PARAM_REG register. | 0x2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 11:6 | R/W | I_CHARGE_TWARM | Charge current setting for the Warm battery temperature pack zone. Regarding the range of values of this bit-field, see also the description of I_CHARGE field of CHARGER_CURRENT_PARAM_REG register. | 0x4 |
| 5:0 | R/W | I_CHARGE_TCOOL | Charge current setting for the "COOL" battery temperature level. Regarding the range of values of this bit-field, see also the description of I_CHARGE field of CHARGER_CURRENT_PARAM_REG register. | 0x5 |

Table 896: CHARGER_VBAT_COMP_TIMER_REG (0x5004043C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|--|-------|
| 25:16 | R | VBAT_COMP_TIMER | Returns the current value of the timer used to determine when the output of the Vbat comparator (checking Vbat vs Pre_Charge and Replenish levels) must be sampled by the digital. As soon as the timer expires (down-counting to 0, starting from the value set in VBAT_COMP_SETTLING), the comparator's output is latched by the Charger's digital block and used by the FSM. Note: When the Charger's FSM is in BYPASSED state, this timer is kept to zero and the SW takes over. In this mode, the specific comparator checks the level of Vbat against the Pre-Charge level. Hence, SW can periodically sample the status of this comparator by reading the MAIN_VBAT_COMP_OUT bit-field of CHARGER_STATUS_REG, to determine if Vbat has exceeded the Pre-Charge level or not. | 0x0 |
| 15:10 | R | - | | 0x0 |
| 9:0 | R/W | VBAT_COMP_SETTLING | Settling time threshold (in us) for the Vbat comparator checking Vbat vs the programmed Pre-Charge and Replenish levels. The settings (voltage levels) of the comparator are controlled by the digital block of the Charger and they are driven based on the state of the main FSM (PRE_CHARGE, END_OF_CHARGE). | 0x63 |

Table 897: CHARGER_VOVP_COMP_TIMER_REG (0x50040440)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------------|---|-------|
| 31:26 | R | OVP_INTERVAL_CHECK_TIMER | The specific bit-field determines the current state of the timer used to periodically check the output of the Over-Voltage Protection comparator's output signal, as soon as the Charger's FSM reaches any of the charging states (PRE/CC/CV_CHARGE). When this happens, the timer starts ticking with the 1Mhz clock, ranging from 0 up to the programmed interval threshold (see also OVP_INTERVAL_CHECK_THRES field). As soon as this timer reaches the programmed threshold value, the Vbat OVP comparator's output is evaluated, increasing or not the counter keeping the | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------------|---|-------|
| | | | <p>consecutive OVP events. It is noted that out of the charging states, the specific timer is kept frozen to zero, not counting.</p> <p>Note : See also the OVP_OCCURRENCES_CNT bit-field of CHARGER_STATUS_REG for the consecutive OVP events counter.</p> | |
| 25:16 | R | VBAT_OVP_COMP_TIMER | <p>Returns the current value of the timer used to determine when the Vbat Over-Voltage protection (OVP) comparator's output must be sampled by the digital. As soon as the timer expires (down-counting to 0, starting from VBAT_OVP_COMP_SETTLING), the comparator's output is latched by the Charger's digital block and used by the main FSM.</p> <p>Note: When the Charger's FSM is in BYPASSED state, this timer is kept to zero and the SW takes over, sampling the status of the VBAT_OVP_COMP_OUT bit-field of CHARGER_STATUS_REG to determine if the Vbat has exceeded the OVP limit.</p> | 0x0 |
| 15:10 | R/W | OVP_INTERVAL_CHECK_THRES | <p>This bit-field determines the periodic interval of checking the dedicated Vbat OVP comparator's output, when the Charger's FSM is in any of the charging states (PRE/CC/CV_CHARGE). The implementation is based on a dedicated timer, counting from zero up to the value programmed into this bit-field (see also OVP_INTERVAL_CHECK_TIMER field's description) and only when the FSM is in any of the three charging states. Out of these states, the timer is kept frozen to zero.</p> <p>As soon as this timer reaches the programmed threshold, the Vbat OVP comparator's output is sampled and depending on its level, (high or low), another counter, keeping the number of consecutive OVP events, is increased or not. The programmed threshold value should always be non-zero.</p> <p>Note: See also the OVP_DEBOUNCE_CNT bit-field of CHARGER_STATUS_REG, for the consecutive OVP events counter.</p> | 0x3F |
| 9:0 | R/W | VBAT_OVP_COMP_SETTLING | Settling time threshold (in us) for the Vbat comparator checking Vbat vs the programmed Over-Voltage level. | 0x63 |

Table 898: CHARGER_TDIE_COMP_TIMER_REG (0x50040444)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 25:16 | R | TDIE_COMP_TIMER | Returns the current value of the timer used to determine when the Die temperature comparator's output must be sampled by the digital. As soon as the timer expires (down-counting to 0, starting from TDIE_COMP_SETTLING) the comparator's output is latched by the Charger's digital block and used by the main FSM. After expiring, the timer starts-over again, down-counting, to enable the continuous monitoring of Die temperature by the digital. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| | | | Note: When the Charger's FSM is in BYPASSED state, this timer is kept to zero and the SW takes over, sampling the status of the TDIE_PROT_COMP_OUT bit-field of CHARGER_STATUS_REG to determine if the Die temperature limit has been exceeded. | |
| 15:10 | R | - | | 0x0 |
| 9:0 | R/W | TDIE_COMP_SETTLING | Settling time threshold (in us) for the Die temperature comparator. | 0x63 |

Table 899: CHARGER_TBAT_MON_TIMER_REG (0x50040448)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|---|-------|
| 25:16 | R | TBAT_MON_TIMER | This is the battery temperature monitoring timer, counting with the Charger's 1KHz clock. If the battery monitor mode is accordingly set in the TBAT_MONITOR_MODE bit-field of CHARGER_CTRL_REG (so either to 0x1 or 0x2), this timer is initially loaded with the value set in TBAT_MON_INTERVAL bit-field in the subsequent 1khz cycles starts down-counting to 0. As soon as the specific timer expires, the JEITA FSM starts-over again, to refresh the battery temperature status. | 0x0 |
| 15:10 | R | - | | 0x0 |
| 9:0 | R/W | TBAT_MON_INTERVAL | Timing interval (in ms) for the Battery temperature monitoring. This interval determines how often the JEITA FSM will be checking and potentially refreshing the Battery temperature status, by selecting accordingly the proper level (Hot, Cold, Warm, Cool or Normal), based on the feedback of the two battery temperature comparators being present in the Charger's analogue circuitry (one for the Hot level and one for Cold, Cool and Warm, to support JEITA). Note: The specific bit-field should be always set to a non-zero value. | 0x63 |

Table 900: CHARGER_TBAT_COMP_TIMER_REG (0x5004044C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|--|-------|
| 25:16 | R | TBAT_COMP_TIMER | Returns the main battery temperature comparator's timer, used for the latching of the comparator's output. The output of the comparator is used by the JEITA FSM, to determine the current battery temperature pack's status. | 0x0 |
| 15:10 | R | - | | 0x0 |
| 9:0 | R/W | TBAT_COMP_SETTLING | Settling time (specified in us) for the main battery temperature comparator, checking for the "COOL", "COLD" and "WARM" levels. The charger's digital block uses a dedicated timer to sample the specific comparator's output. The comparator's output is latched as soon as the timer expires, reaching 0. Then, the timer is reloaded with the settling time | 0x63 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | value and starts-over, down-counting to 0. Note: The specific bit-field should be always set to a non-zero value. | |

Table 901: CHARGER_THOT_COMP_TIMER_REG (0x50040450)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|---|-------|
| 25:16 | R | THOT_COMP_TIME_R | Returns the battery temperature comparator's timer dedicated for the "Hot" level. | 0x0 |
| 15:10 | R | - | | 0x0 |
| 9:0 | R/W | THOT_COMP_SETTLING | Charger's battery temperature comparator settling time (specified in us), specifically for the Hot temperature zone. The charger's digital block uses a dedicated timer to sample the specific comparator's output. The comparator's output is latched as soon as the timer expires, reaching 0. Then, the timer is reloaded with the settling time value and starts-over again Note: The specific bit-field should be always set to a non-zero value. | 0x63 |

Table 902: CHARGER_PWR_UP_TIMER_REG (0x50040454)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|---|-------|
| 25:16 | R | CHARGER_PWR_UP_TIMER | Returns the current value of the charger's power-up timer, running with the 1Khz clock. Note: The specific timer is reset to the value programmed to CHARGER_PWR_UP_SETTLING bit-field, when the Charger's analogue circuitry has been enabled, after being disabled initially. By setting CHARGER_CTRL_REG[CHARGER_ENABLE] to '0', the analogue part is disabled and in order to be properly enable, SW has to wait for 1ms (one 1Khz clock period) time. The latter is needed to ensure that the power-up timer's control signals in the Charger's digital part will be cleared when the analogue part is again enabled, so that a proper new start-up of the Charger's FSM is possible. | 0x0 |
| 15:10 | R | - | | 0x0 |
| 9:0 | R/W | CHARGER_PWR_UP_SETTLING | This bit-field determines the charger's power-up (settling) time, required for the analogue circuitry of the charger. As soon as the charger is powered-on by setting the CHARGER_ENABLE bit-field of CHARGER_CTRL_REG, the charger's FSM loads a dedicated timer with this value and waits for this timer to expire, before proceeding to the next states. Note: The specific bit-field should be always set to a non-zero value. | 0x63 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 903: CHARGER_STATE_IRQ_MASK_REG (0x50040458)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------------------------|--|-------|
| 11 | R/W | CV_TO_PRECHAR GE_IRQ_EN | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from CV_CHARGE to PRE_CHARGE state. | 0x0 |
| 10 | R/W | CC_TO_PRECHAR GE_IRQ_EN | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from CC_CHARGE to PRE_CHARGE state. | 0x0 |
| 9 | R/W | CV_TO_CC_IRQ_E N | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from CV_CHARGE to CC_CHARGE state. | 0x0 |
| 8 | R/W | TBAT_STATUS_UP DATE_IRQ_EN | When set, this bit-field enables the generation of the Charger's state IRQ as soon as the battery temperature status is refreshed by the Charger's Battery temperature monitor (JEITA) FSM. As soon as the specific FSM checks the current battery temperature level, it notifies the main Charger FSM that it has run and that the Battery temperature pack state is checked (and potentially refreshed with a new status). | 0x0 |
| 7 | R/W | TBAT_PROT_TO_P RECHARGE_IRQ_E N | When set, this bit-field enables the Charger's state IRQ generation as soon as the Charger's FSM switches from the Battery temperature protection state (TBAT_PROT) to PRE_CHARGE, resuming charging. | 0x0 |
| 6 | R/W | TDIE_PROT_TO_P RECHARGE_IRQ_E N | When set, this bit-field enables the Charger's state IRQ generation as soon as the Charger's FSM switches from the Die temperature protection state (TDIE_PROT) to PRE_CHARGE, resuming charging. | 0x0 |
| 5 | R/W | EOC_TO_PRECHA RGE_IRQ_EN | When set, this bit-field enables the Charger's State IRQ generation as soon as the Charger's FSM switches from END_OF_CHARGE again to PRE_CHARGE state. This happens when the Vbat voltage level is detected to be below the Replenish level set. | 0x0 |
| 4 | R/W | CV_TO_EOC_IRQ_ EN | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from CV_CHARGE to END_OF_CHARGE state. | 0x0 |
| 3 | R/W | CC_TO_EOC_IRQ_ EN | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from CC_CHARGE to END_OF_CHARGE state. | 0x0 |
| 2 | R/W | CC_TO_CV_IRQ_E N | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from CC_CHARGE to CV_CHARGE state. | 0x0 |
| 1 | R/W | PRECHARGE_TO_ CC_IRQ_EN | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from PRE_CHARGE to CC_CHARGE state.. | 0x0 |
| 0 | R/W | DISABLED_TO_PR ECHARGE_IRQ_EN | When set, this bit-field enables the IRQ generation as soon as the Charger's FSM switches from DISABLED to PRE_CHARGE state. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 904: CHARGER_ERROR_IRQ_MASK_REG (0x5004045C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------------|---|-------|
| 6 | R/W | TBAT_ERROR_IRQ_EN | When set, it enables the generation of Battery temperature IRQs. The IRQ is generated as soon as the JEITA FSM detects that the battery temperature is either in the "Hot" or in the "Cold" temperature region, by sampling the respective comparators' output. | 0x0 |
| 5 | R/W | TDIE_ERROR_IRQ_EN | When set, it enables the generation of Die temperature error IRQs. The IRQ is generated as soon as a Die temperature error is captured, so as soon as the Charger's FSM moves to the TDIE_PROT state. For this to happen, the Die temperature comparator should indicate that Die temperature has exceeded the limit defined in CHARGER_TEMPSET_PARAM_REG.TDIE_MAX. | 0x0 |
| 4 | R/W | VBAT_OVP_ERROR_IRQ_EN | When set, it enables the generation of VBAT_OVP IRQs. The IRQ is generated as soon as the dedicated Vbat comparator shows that Vbat has exceeded the OVP level and the Charger's FSM has switched to the respective error state ("ERROR"). | 0x0 |
| 3 | R/W | TOTAL_CHARGE_TIMEOUT_IRQ_EN | When set, it enables the total charge timeout IRQs. The IRQ is generated as soon as the Charger's global charge timer expires, reaching 0. | 0x0 |
| 2 | R/W | CV_CHARGE_TIMEOUT_IRQ_EN | When set, it enables the CV charge timeout IRQs. The IRQ is generated as soon as the Charger's state timer expires, reaching 0 when the FSM is in the CV_CHARGE state. | 0x0 |
| 1 | R/W | CC_CHARGE_TIMEOUT_IRQ_EN | When set, it enables the CC charge timeout IRQs. The IRQ is generated as soon as the Charger's state timer, expires, reaching 0. | 0x0 |
| 0 | R/W | PRECHARGE_TIMEOUT_IRQ_EN | When set, it enables the Pre-Charge timeout IRQs. The IRQ is generated as soon as the Charger's state timer expires, reaching 0. | 0x0 |

Table 905: CHARGER_STATE_IRQ_STATUS_REG (0x50040460)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|--|-------|
| 11 | R | CV_TO_PRECHARGE_IRQ | 0 = No transition of the Charger's FSM from CV_CHARGE to PRE_CHARGE state has been captured 1 = Charger's FSM has switched from CV_CHARGE to PRE_CHARGE state | 0x0 |
| 10 | R | CC_TO_PRECHARGE_IRQ | 0 = No transition of the Charger's FSM from CC_CHARGE to PRE_CHARGE state has been captured 1 = Charger's FSM has switched from CC_CHARGE to PRE_CHARGE state | 0x0 |
| 9 | R | CV_TO_CC_IRQ | 0 = No transition of the Charger's FSM from CV_CHARGE to CC_CHARGE state has been captured 1 = Charger's FSM has switched from CV_CHARGE to CC_CHARGE state | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------------|---|-------|
| 8 | R | TBAT_STATUS_UP_DATE_IRQ | 0 = No battery temperature status update event has been captured 1 = Battery temperature pack's status has been checked and refreshed by the Charger's Battery temperature monitor FSM. Thus, the new status of the battery temperature should be checked by SW. | 0x0 |
| 7 | R | TBAT_PROT_TO_PRECHARGE_IRQ | 0 = No transition of the Charger's FSM from TBAT_PROT to PRE_CHARGE state has been captured 1 = Charger's FSM has switched from TBAT_PROT to PRE_CHARGE state, resuming charging after having recovered from a battery temperature error. | 0x0 |
| 6 | R | TDIE_PROT_TO_PRECHARGE_IRQ | 0 = No transition of the Charger's FSM from TDIE_PROT to PRE_CHARGE state has been captured 1 = Charger's FSM has switched from TDIE_PROT to PRE_CHARGE state, resuming charging after having recovered from a Die temperature error. | 0x0 |
| 5 | R | EOC_TO_PRECHARGE_IRQ | 0 = No transition of the Charger's FSM from END_OF_CHARGE to PRE_CHARGE state has been captured 1 = Charger's FSM has switched from END_OF_CHARGE to PRE_CHARGE state | 0x0 |
| 4 | R | CV_TO_EOC_IRQ | 0 = No transition of the Charger's FSM from CV_CHARGE to END_OF_CHARGE state has been captured 1 = Charger's FSM has switched from CV_CHARGE to END_OF_CHARGE state | 0x0 |
| 3 | R | CC_TO_EOC_IRQ | 0 = No transition of the Charger's FSM from CC_CHARGE to END_OF_CHARGE state has been captured 1 = Charger's FSM has switched from CC_CHARGE to END_OF_CHARGE state | 0x0 |
| 2 | R | CC_TO_CV_IRQ | 0 = No transition of the Charger's FSM from CC_CHARGE to CV_CHARGE state has been captured 1 = Charger's FSM has switched from CC_CHARGE to CV_CHARGE state | 0x0 |
| 1 | R | PRECHARGE_TO_CC_IRQ | 0 = No transition of the Charger's FSM from PRE_CHARGE to CC_CHARGE state has been captured 1 = Charger's FSM has switched from PRE_CHARGE to CC_CHARGE state | 0x0 |
| 0 | R | DISABLED_TO_PRECHARGE_IRQ | 0 = No transition of the Charger's FSM from DISABLED to PRE_CHARGE state has been captured 1 = Charger's FSM has switched from DISABLED to PRE_CHARGE state | 0x0 |

Table 906: CHARGER_ERROR_IRQ_STATUS_REG (0x50040464)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|---|-------|
| 6 | R | TBAT_ERROR_IRQ | 0 = No Battery temperature error IRQ event is | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------------|---|-------|
| | | | <p>captured, so charging may continue</p> <p>1 = A Battery temperature error IRQ event has been captured, declaring that the Charger's FSM has moved to the respective error state (TBAT_PROT).</p> <p>Note : The status bit is updated automatically when the Battery temperature is detected to be either in the HOT or in the COLD zone, regardless of the state of the respective IRQ mask bit.</p> | |
| 5 | R | TDIE_ERROR_IRQ | <p>0 = No Die temperature error IRQ events have been captured, so charging may continue</p> <p>1 = A Die temperature error IRQ event is captured, declaring that the Charger's FSM has switched to the respective error state (TDIE_PROT) and charging will be automatically stopped.</p> <p>Note : The status bit is updated automatically when a Die temperature error is detected, thus when the die temperature is found to have exceeded the programmed level, regardless of the state of the respective IRQ mask bit. The same applies to all the rest of the bits of CHARGER_ERROR_IRQ_STATUS_REG.</p> | 0x0 |
| 4 | R | VBAT_OVP_ERROR_IRQ | <p>0 = Vbat has not exceeded the Over-Voltage Protection (OVP) level, so charging may continue</p> <p>1 = Vbat has exceeded the Over-Voltage level, thus an OVP error event has been captured. The Charger's FSM switches to the respective error state (ERROR) as soon as the OVP event is captured by the digital part of the Charger and charging will be automatically stopped.</p> | 0x0 |
| 3 | R | TOTAL_CHARGE_TIMEOUT_IRQ | <p>0 = Total charge time counter has not yet reached the maximum charge time (set in CHARGER_TOTAL_CHARGE_TIME_REG)</p> <p>1 = Total charge time counter has reached the maximum charge time programmed. The Charger's FSM will move to the respective error state (ERROR) and charging will be automatically stopped, as soon as the specific event is captured.</p> | 0x0 |
| 2 | R | CV_CHARGE_TIMEOUT_IRQ | <p>0 = State charge time counter has not yet reached the maximum CV charge time (set in CHARGER_CV_CHARGE_TIME_REG)</p> <p>1 = Total charge time counter has reached the maximum CV charge time programmed. The Charger's FSM will move to the respective error state (ERROR) and charging will be automatically stopped, as soon as the specific event is captured.</p> | 0x0 |
| 1 | R | CC_CHARGE_TIMEOUT_IRQ | <p>0 = State charge time counter has not yet reached the maximum CC charge time (set in CHARGER_CC_CHARGE_TIME_REG)</p> <p>1 = Total charge time counter has reached the maximum CC charge time programmed. The Charger's FSM will move to the respective error state (ERROR) and charging will be automatically stopped, as soon as the specific event is captured.</p> | 0x0 |
| 0 | R | PRECHARGE_TIMEOUT_IRQ | <p>0 = State charge time counter has not yet reached the maximum Pre-charge time (set in CHARGER_PRECHARGE_TIME_REG)</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 1 = Total charge time counter has reached the maximum Pre-charge time programmed. The Charger's FSM will move to the respective error state (ERROR) and charging will be automatically stopped, as soon as the specific event is captured. | |

Table 907: CHARGER_STATE_IRQ_CLR_REG (0x50040468)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--|---|-------|
| 11 | R0/W | CV_TO_PRECHAR GE_IRQ_CLR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 10 | R0/W | CC_TO_PRECHAR GE_IRQ_CLR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 9 | R0/W | CV_TO_CC_IRQ_C LR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 8 | R0/W | TBAT_STATUS_UP DATE_IRQ_CLR | Writing a 1 will reset the Battery temperature status update IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 7 | R0/W | TBAT_PROT_TO_P RECHARGE_IRQ_C LR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 6 | R0/W | TDIE_PROT_TO_P RECHARGE_IRQ_C LR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 5 | R0/W | EOC_TO_PRECHA RGE_IRQ_CLR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 4 | R0/W | CV_TO_EOC_IRQ_ CLR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 3 | R0/W | CC_TO_EOC_IRQ_ CLR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 2 | R0/W | CC_TO_CV_IRQ_C LR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 1 | R0/W | PRECHARGE_TO_ CC_IRQ_CLR | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 0 | R0/W | DISABLED_TO_PR ECHARGE_IRQ_CL R | Writing a 1 will reset the respective Charger's State IRQ status bit ; writing a 0 will have no effect | 0x0 |

Table 908: CHARGER_ERROR_IRQ_CLR_REG (0x5004046C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------------------|--|-------|
| 6 | R0/W | TBAT_ERROR_IRQ _CLR | Writing a 1 will reset the respective Charger's Error IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 5 | R0/W | TDIE_ERROR_IRQ_ CLR | Writing a 1 will reset the respective Charger's Error IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 4 | R0/W | VBAT_OVP_ERROR _IRQ_CLR | Writing a 1 will reset the respective Charger's Error IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 3 | R0/W | TOTAL_CHARGE_T IMEOUT_IRQ_CLR | Writing a 1 will reset the respective Charger's Error IRQ status bit ; writing a 0 will have no effect | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------------|--|-------|
| 2 | R0/W | CV_CHARGE_TIME_OUT_IRQ_CLR | Writing a 1 will reset the respective Charger's Error IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 1 | R0/W | CC_CHARGE_TIME_OUT_IRQ_CLR | Writing a 1 will reset the respective Charger's Error IRQ status bit ; writing a 0 will have no effect | 0x0 |
| 0 | R0/W | PRECHARGE_TIME_OUT_IRQ_CLR | Writing a 1 will reset the respective Charger's Error IRQ status bit ; writing a 0 will have no effect | 0x0 |

42.29 Clock Generation Controller Registers

Table 909: Register map CRG

| Address | Register | Description |
|------------|------------------------|---|
| 0x50000000 | CLK_AMBA_REG | HCLK, PCLK, divider and clock gates |
| 0x50000010 | CLK_RADIO_REG | Radio PLL control register |
| 0x50000014 | CLK_CTRL_REG | Clock control register |
| 0x50000018 | CLK_TMR_REG | Clock control for the timers |
| 0x5000001C | CLK_SWITCH2XTAL_REG | Switches clock from RC32M to XTAL32M |
| 0x50000020 | PMU_CTRL_REG | Power Management Unit control register |
| 0x50000024 | SYS_CTRL_REG | System Control register |
| 0x50000028 | SYS_STAT_REG | System status register |
| 0x5000003C | CLK_RC32K_REG | 32 kHz RC oscillator register |
| 0x50000040 | CLK_XTAL32K_REG | 32 kHz XTAL oscillator register |
| 0x50000044 | CLK_RC32M_REG | Fast RC control register |
| 0x50000048 | CLK_RCX_REG | RCX-oscillator control register |
| 0x5000004C | CLK_RTCDIV_REG | Divisor for RTC 100Hz clock |
| 0x50000050 | BANDGAP_REG | bandgap trimming |
| 0x50000054 | VBUS_IRQ_MASK_REG | IRQ masking |
| 0x50000058 | VBUS_IRQ_CLEAR_REG | Clear pending IRQ register |
| 0x50000060 | BOD_CTRL_REG | Brown Out Detection control register |
| 0x50000064 | BOD_LVL_CTRL0_REG | |
| 0x50000068 | BOD_LVL_CTRL1_REG | |
| 0x5000006C | BOD_LVL_CTRL2_REG | |
| 0x50000070 | P0_PAD_LATCH_REG | Control the state retention of the GPIO ports |
| 0x50000074 | P0_SET_PAD_LATCH_REG | Control the state retention of the GPIO ports |
| 0x50000078 | P0_RESET_PAD_LATCH_REG | Control the state retention of the GPIO ports |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|-------------|-------------------------|---|
| 0x50000007C | P1_PAD_LATCH_REG | Control the state retention of the GPIO ports |
| 0x500000080 | P1_SET_PAD_LATCH_REG | Control the state retention of the GPIO ports |
| 0x500000084 | P1_RESET_PAD_LAT_CH_REG | Control the state retention of the GPIO ports |
| 0x500000090 | BOD_STATUS_REG | |
| 0x500000094 | POR_VBAT_CTRL_REG | Controls the POR on VBAT |
| 0x500000098 | POR_PIN_REG | Selects a GPIO pin for POR generation |
| 0x50000009C | POR_TIMER_REG | Time for POR to happen |
| 0x5000000A0 | LDO_VDDD_HIGH_CTRL_REG | LDO control register |
| 0x5000000A4 | BIAS_VREF_SEL_REG | |
| 0x5000000BC | RESET_STAT_REG | Reset status register |
| 0x5000000C0 | RAM_PWR_CTRL_REG | Control power state of System RAMS |
| 0x5000000CC | SECURE_BOOT_REG | Controls secure booting |
| 0x5000000D4 | DISCHARGE_RAIL_REG | Immediate rail resetting. There is no LDO/DCDC gating |
| 0x5000000EC | ANA_STATUS_REG | Analog Signals Status Register |
| 0x5000000F0 | POWER_CTRL_REG | Power control register |
| 0x5000000F4 | PMU_SLEEP_REG | Configures the sleep/wakeup strategy |
| 0x5000000F8 | PMU_TRIM_REG | LDO trimming register |
| 0x50010000 | CLK_FREQ_TRIM_REG | Xtal frequency trimming register. |
| 0x50010010 | TRIM_CTRL_REG | Control trimming of the XTAL32M |
| 0x50010018 | XTALRDY_CTRL_REG | Control register for XTALRDY IRQ |
| 0x5001001C | XTALRDY_STAT_REG | Difference between XTAL_OK and XTALRDY_IRQ in LP clock cycles |
| 0x50010030 | XTAL32M_CTRL0_REG | Control register for XTAL32M |
| 0x50010034 | XTAL32M_CTRL1_REG | Control register for XTAL32M |
| 0x50010038 | XTAL32M_CTRL2_REG | Control register for XTAL32M |
| 0x5001003C | XTAL32M_CTRL3_REG | Control register for XTAL32M |
| 0x50010040 | XTAL32M_CTRL4_REG | Control register for XTAL32M |
| 0x50010050 | XTAL32M_STAT0_REG | Status register for XTAL32M |
| 0x50010054 | XTAL32M_STAT1_REG | Status register for XTAL32M |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|--------------------|---|
| 0x50010060 | PLL_SYS_CTRL1_REG | System PLL control register 1. |
| 0x50010064 | PLL_SYS_CTRL2_REG | System PLL control register 2. |
| 0x50010068 | PLL_SYS_CTRL3_REG | System PLL control register 3. |
| 0x50010070 | PLL_SYS_STATUS_REG | System PLL status register. |
| 0x50020904 | CLK_COM_REG | Peripheral divider register |
| 0x50020908 | SET_CLK_COM_REG | Peripheral divider register SET register. Reads back 0x0000 |
| 0x5002090C | RESET_CLK_COM_REG | Peripheral divider register RESET register. Reads back 0x0000 |
| 0x50030C04 | CLK_PER_REG | Peripheral divider register |
| 0x50030C08 | SET_CLK_PER_REG | Peripheral divider register SET register, reads 0x0000 |
| 0x50030C0C | RESET_CLK_PER_REG | Peripheral divider register RESET register, reads 0x0000 |
| 0x50030C40 | PCM_DIV_REG | PCM divider and enables |
| 0x50030C44 | PCM_FDIV_REG | PCM fractional division register |
| 0x50030C48 | PDM_DIV_REG | PDM divider and enables |
| 0x50030C4C | SRC_DIV_REG | SRC divider and enables |
| 0x50040500 | CLK_SYS_REG | Peripheral divider register |
| 0x50040504 | BATCHCHECK_REG | |

Table 910: CLK_AMBA_REG (0x50000000)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 15 | R/W | QSPI2_ENABLE | Clock enable for QSPI RAM controller | 0x0 |
| 14:13 | R/W | QSPI2_DIV | QSPI divider 00 = divide by 1 01 = divide by 2 10 = divide by 4 11 = divide by 8 | 0x0 |
| 12 | R/W | QSPI_ENABLE | Clock enable for QSPI controller | 0x0 |
| 11:10 | R/W | QSPI_DIV | QSPI divider 00 = divide by 1 01 = divide by 2 10 = divide by 4 11 = divide by 8 | 0x0 |
| 9 | R/W | OTP_ENABLE | Clock enable for OTP controller | 0x0 |
| 8 | R/W | TRNG_CLK_ENABLE | Clock enable for TRNG block | 0x0 |
| 7 | R/W | - | | 0x0 |
| 6 | R/W | AES_CLK_ENABLE | Clock enable for AES crypto block | 0x0 |
| 5:4 | R/W | PCLK_DIV | APB interface clock, Cascaded with HCLK: | 0x2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| | | | 00 = divide hclk by 1 01 = divide hclk by 2 10 = divide hclk by 4 11 = divide hclk by 8 | |
| 3 | R/W | - | | 0x0 |
| 2:0 | R/W | HCLK_DIV | AHB interface and microprocessor clock. Source clock divided by: 000 = divide hclk by 1 001 = divide hclk by 2 010 = divide hclk by 4 011 = divide hclk by 8 1xx = divide hclk by 16 | 0x2 |

Table 911: CLK_RADIO_REG (0x50000010)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|---|-------|
| 5 | R/W | RFCU_ENABLE | Enable the RF control Unit clock | 0x0 |
| 4 | R/W | CMAC_SYNCH_RESET | Force synchronous reset to CMAC core and Sleep Timer. Its effective only when both Radio and Timer Power Domains are powered and the clocks are enabled. CMAC CPU and CMAC registers, including the retained ones, will be reset. It should be kept in reset for enough time to make sure that it will be captured by CMAC, Low Power and APB clocks. | 0x1 |
| 3 | R/W | CMAC_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 2 | R/W | CMAC_CLK_ENABLE | Enables the clock | 0x0 |
| 1:0 | R/W | CMAC_DIV | Division factor for CMAC 0x0 = divide by 1 0x1 = divide by 2 0x2 = divide by 4 0x3 = divide by 8 | 0x0 |

Table 912: CLK_CTRL_REG (0x50000014)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|---|-------|
| 15 | R | RUNNING_AT_PLL96M | Indicates that the PLL96MHz clock is used as clock, and may not be switched off | 0x0 |
| 14 | R | RUNNING_AT_XTAL32M | Indicates that the XTAL32M clock is used as clock, and may not be switched off | 0x0 |
| 13 | R | RUNNING_AT_RC32M | Indicates that the RC32M clock is used as clock | 0x1 |
| 12 | R | RUNNING_AT_LP | Indicates that either the LP_CLK is being used as | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| | | CLK | clock | |
| 11:7 | R | - | | 0x0 |
| 6 | R/W | - | | 0x1 |
| 5 | R/W | - | | 0x0 |
| 4 | R/W | USB_CLK_SRC | Selects the USB source clock 0 : PLL clock, divided by 2 1 : HCLK | 0x0 |
| 3:2 | 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 GPIO to FUNC_GPIO) | 0x0 |
| 1:0 | R/W | SYS_CLK_SEL | Selects the clock source. 0x0 : XTAL32M 0x1 : RC32M 0x2 : The Low Power clock is used 0x3 : The PLL96Mhz is used | 0x1 |

Table 913: CLK_TMR_REG (0x50000018)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|---|-------|
| 2 | R/W | TMR2_PWM_AON_MODE | Maps Timer2_pwm onto P1_06. This state is preserved during deep sleep, to allow PWM output on the pad during deep sleep. | 0x0 |
| 1 | R/W | TMR_PWM_AON_MODE | Maps Timer1_pwm onto P1_01 This state is preserved during deep sleep, to allow PWM output on the pad during deep sleep. | 0x0 |
| 0 | R/W | WAKEUPCT_ENABLE | Enables the clock | 0x0 |

Table 914: CLK_SWITCH2XTAL_REG (0x5000001C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|---|-------|
| 0 | W | SWITCH2XTAL | When writing to this register, the clock switch will happen from RC32M to XTAL32M. If any other clock is selected than RC32M, the selection is discarded. | 0x0 |

Table 915: PMU_CTRL_REG (0x50000020)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|--|-------|
| 8 | R/W | ENABLE_CLKLESS | Selects the clockless sleep mode. Wakeup is done asynchronously. When set to '1', the lp_clk is stopped during deep | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| | | | sleep, until a wakeup event (not debounced) is detected by the WAKUPCT block. When set to '0', the lp_clk continues running, so the MAC counters keep on running. This mode cannot be combined with regulated sleep, so keep SLEEP_TIMER=0 when using ENABLE_CLKLESS. | |
| 7 | R/W | RETAIN_CACHE | Selects the retainability of the cache block during deep sleep. '1' is retainable, '0' is power gated | 0x0 |
| 6 | R/W | SYS_SLEEP | Put the System powerdomain (PD_SYS) in powerdown. If this bit is '1', and there is no pending IRQ in the PDC for the M33, the PD_SYS will be switched off. Wakeup should be handled by the PDC. | 0x0 |
| 5 | R/W | RESET_ON_WAKEUP | Perform a Hardware Reset after waking up. Booter will be started. | 0x0 |
| 4 | R/W | MAP_BANDGAP_EN | Setting this bit will: -map bandgap_enable to P0_25 -map (wokenup OR cmac_slp_timer_expire) to P0_16 | 0x0 |
| 3 | R/W | COM_SLEEP | Put the Communications powerdomain (PD_COM) in powerdown | 0x1 |
| 2 | R/W | TIM_SLEEP | Put the Timers Powerdomain (PD_TIM) in powerdown. | 0x1 |
| 1 | R/W | RADIO_SLEEP | Put the digital part of the radio, including CMAC (PD_RAD) in powerdown | 0x1 |
| 0 | R/W | PERIPH_SLEEP | Put the peripherals power domain (PD_PER) in powerdown | 0x1 |

Table 916: SYS_CTRL_REG (0x50000024)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|---|-------|
| 15 | W | SW_RESET | Writing a '1' to this bit will generate a SW_RESET. | 0x0 |
| 14:11 | R | - | | 0x0 |
| 10 | R/W | CACHERAM_MUX | Controls accessibility of Cache RAM: 0: the cache controller is bypassed, the cacheRAM is visible in the memory space 1: the cache controller is enabled, the cacheRAM is not visible anymore in the memory space | 0x0 |
| 9 | R/W | TIMEOUT_DISABLE | 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 | 0x0 |
| 8 | - | - | | 0x0 |
| 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 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| | | | and SW_CLK can be used as gpio ports. | |
| 6 | - | - | | 0x0 |
| 5 | R/W | - | | 0x1 |
| 4 | R/W | QSPI_INIT | Enables QSPI initialization after wakeup | 0x0 |
| 3 | R/W | REMAP_INTVECT | 0: normal operation 1: If ARM is in address range 0 to 0x1FF then the address is remapped to SYS-RAM 0x0080.0000 to 0x0080.01FF. This allows to put the interrupt vector table to be placed in RAM while executing from QSPI. | 0x0 |
| 2:0 | R/W | REMAP_ADR0 | Controls which memory is located at address 0x0000 for execution. 0x0: ROM 0x1: OTP un-cached 0x2: QSPI FLASH cached (see also the CACHE_FLASH_REG.FLASH_REGION.* descriptions) Note 1: When REMAP_ADR0=0x2, address 0x0 is mapped to FLASH_REGION_BASE + FLASH_REGION_OFFSET<<2. Note 2: When REMAP_ADR0=0x2, the CPU can <u>only</u> access the Flash region [FLASH_REGION_BASE + FLASH_REGION_OFFSET<<2, FLASH_REGION_SIZE] from the 0x16000000 address range. The complete Flash can be accessed via the 0x36000000 address range but only uncached. 0x3: RAMS un-cached 0x4: QSPI FLASH un-cached (for verification only) 0x5: SYSRAM2 (for testing purposes only) 0x6: Cache Data RAM un-cached (CACHERAM_MUX=0, for testing purposes only) Note 1: DWord (64 bits) access is not supported by the Cache Data RAM interface in mirrored mode (only 32, 16 and 8 bits). Note 2: DMA access is not supported by the Cache Data RAM interface when REMAP_ADR0=0x6. | 0x0 |

Table 917: SYS_STAT_REG (0x50000028)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| 13 | R | POWER_IS_UP | Indicates that the Startup statemachine is finished, and all power regulation is in order. In UltraFastWakeups mode, the SW needs to wait for this signal before starting any heavy traffic. | 0x1 |
| 12 | R | DBG_IS_ACTIVE | Indicates that a debugger is attached. | 0x0 |
| 11 | R | COM_IS_UP | Indicates that PD_COM is functional | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| 10 | R | COM_IS_DOWN | Indicates that PD_COM is in power down | 0x1 |
| 9 | R | TIM_IS_UP | Indicates that PD_TIM is functional | 0x0 |
| 8 | R | TIM_IS_DOWN | Indicates that PD_TIM is in power down | 0x1 |
| 7 | R | MEM_IS_UP | Indicates that PD_MEM is functional | 0x1 |
| 6 | R | MEM_IS_DOWN | Indicates that PD_MEM is in power down | 0x0 |
| 5 | R | SYS_IS_UP | Indicates that PD_SYS is functional | 0x1 |
| 4 | R | SYS_IS_DOWN | Indicates that PD_SYS is in power down | 0x0 |
| 3 | R | PER_IS_UP | Indicates that PD_PER is functional | 0x0 |
| 2 | R | PER_IS_DOWN | Indicates that PD_PER is in power down | 0x1 |
| 1 | R | RAD_IS_UP | Indicates that PD_RAD is functional | 0x0 |
| 0 | R | RAD_IS_DOWN | Indicates that PD_RAD is in power down | 0x1 |

Table 918: CLK_RC32K_REG (0x5000003C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|---|-------|
| 4:1 | R/W | RC32K_TRIM | 0000 = lowest frequency 0111 = default 1111 = highest frequency | 0x7 |
| 0 | R/W | RC32K_ENABLE | Enables the 32kHz RC oscillator | 0x1 |

Table 919: CLK_XTAL32K_REG (0x50000040)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------------|--|-------|
| 9 | - | - | | 0x0 |
| 8 | - | - | | 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 32kHz 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. Preferred setting will be provided by Dialog | 0x3 |
| 0 | R/W | XTAL32K_ENABLE | Enables the 32kHz XTAL oscillator | 0x0 |

Table 920: CLK_RC32M_REG (0x50000044)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|-------------|-------|
| 25 | - | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------|--|-------|
| 24:22 | R/W | - | | 0x0 |
| 21:20 | R/W | RC32M_INIT_RANGE | Course frequency adjustment | 0x1 |
| 19:12 | R/W | RC32M_INIT_DEL | Fine frequency adjustment | 0x80 |
| 11:9 | R/W | RC32M_INIT_DTCF | Fine duty-cycle adjustment. 0x0: minimum 0x2: default 0x4: maximum 0x5 until 0x7: oscillator does not work. | 0x2 |
| 8:5 | R/W | RC32M_INIT_DTC | Course duty-cycle adjustment. 0x0: minimum 0x5: default 0xA: maximum 0xB until 0xF: oscillator does not work | 0x5 |
| 4:1 | R/W | RC32M_BIAS | Bias adjustment | 0x7 |
| 0 | R/W | RC32M_ENABLE | Enables the 32MHz RC oscillator | 0x0 |

Table 921: CLK_RCX_REG (0x50000048)

| 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 922: CLK_RTC DIV_REG (0x5000004C)

| 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 100Hz generation | 0x147 |
| 9:0 | R/W | RTC_DIV_FRAC | Fractional divisor part for RTC 100Hz generation. if RTC_DIV_DENOM=1, <RTC_DIV_FRAC> out of | 0x2A8 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 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> | |

Table 923: BANDGAP_REG (0x50000050)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------------|--|-------|
| 12 | R/W | BANDGAP_ENABLE_CLAMP | Enables a supply clamp inside the bandgap that improves PSRR. Should be enabled by software after cold boot. | 0x0 |
| 11:6 | R/W | BGR_ITRIM | Current trimming for bias | 0x0 |
| 5 | R/W | SYSRAM_LPMX | RAM 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 0.81V, it is necessary to hold this pin high to maintain data retention. | 0x1 |
| 4:0 | R/W | BGR_TRIM | Trim register for bandgap | 0x0 |

Table 924: VBUS_IRQ_MASK_REG (0x50000054)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|--|-------|
| 1 | R/W | VBUS_IRQ_EN_RISE | Setting this bit to '1' enables VBUS_IRQ generation when the VBUS starts to ramp above threshold | 0x0 |
| 0 | R/W | VBUS_IRQ_EN_FALL | Setting this bit to '1' enables VBUS_IRQ generation when the VBUS starts to fall below threshold | 0x0 |

Table 925: VBUS_IRQ_CLEAR_REG (0x50000058)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|---|-------|
| 15:0 | W | VBUS_IRQ_CLEAR | Writing any value to this register will reset the VBUS_IRQ line | 0x0 |

Table 926: BOD_CTRL_REG (0x50000060)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 17 | - | - | | 0x0 |
| 16 | R/W | BOD_V14_RST_EN | If set, generate power-on reset on channel V14 | 0x1 |
| 15 | R/W | BOD_V18F_RST_EN | If set, generate power-on reset on channel V18F | 0x1 |
| 14 | R/W | BOD_VDD_RST_EN | If set, generate power-on reset on channel VDD | 0x1 |
| 13 | R/W | BOD_V18P_RST_EN | If set, generate power-on reset on channel V18P | 0x1 |
| 12 | R/W | BOD_V18_RST_EN | If set, generate power-on reset on channel V18 | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------|--|-------|
| 11 | R/W | BOD_V30_RST_EN | If set, generate power-on reset on channel V30 | 0x1 |
| 10 | R/W | BOD_VBAT_RST_EN | If set, generate power-on reset on channel VBAT | 0x1 |
| 9 | R/W | BOD_V14_EN | Enable brown-out detection for channel V14 | 0x0 |
| 8 | R/W | BOD_V18F_EN | Enable brown-out detection for channel V18F | 0x0 |
| 7 | R/W | BOD_VDD_EN | Enable brown-out detection for channel VDD | 0x1 |
| 6 | R/W | BOD_V18P_EN | Enable brown-out detection for channel V18P | 0x0 |
| 5 | R/W | BOD_V18_EN | Enable brown-out detection for channel V18 | 0x0 |
| 4 | R/W | BOD_V30_EN | Enable brown-out detection for channel V30 | 0x0 |
| 3 | R/W | BOD_VBAT_EN | Enable brown-out detection for channel VBAT | 0x0 |
| 2 | R/W | BOD_STATUS_CLEAR | Clears the brownout status register | 0x0 |
| 1:0 | R/W | BOD_CLK_DIV | Brown-out detector clock divider. 0x0: BOD_CLK/1 0x1: BOD_CLK/2 0x2: BOD_CLK/4 0x3: BOD_CLK/8 (BOD_CLK = 1MHz) | 0x0 |

Table 927: BOD_LVL_CTRL0_REG (0x50000064)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|---|-------|
| 26:18 | R/W | BOD_LVL_V18 | Brown-out detection level for V18; disable the bod channel before adjusting the level setting. $VTH_{BOD} = 1.2 * (BOD_LVL+1)/192$ | 0x107 |
| 17:9 | R/W | BOD_LVL_V30 | Brown-out detection level for V30; disable the bod channel before adjusting the level setting. $VTH_{BOD} = 1.2 * (BOD_LVL+1)/192$ | 0x107 |
| 8:0 | R/W | BOD_LVL_VBAT | Brown-out detection level for VBAT; disable the bod channel before adjusting the level setting. $VTH_{BOD} = 1.5 * (BOD_LVL+1)/192$ | 0xAF |

Table 928: BOD_LVL_CTRL1_REG (0x50000068)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------|--|-------|
| 24:17 | R/W | BOD_LVL_VDD_RESET | Brown-out detection level for VDD in sleep; disable the bod channel before adjusting the level setting. $VTH_{BOD} = 1.2 * (BOD_LVL+1)/192$ | 0x6F |
| 16:9 | R/W | BOD_LVL_VDD_ON | Brown-out detection level for VDD in active; disable the bod channel before adjusting the level setting. $VTH_{BOD} = 1.2 * (BOD_LVL+1)/192$ | 0x80 |
| 8:0 | R/W | BOD_LVL_V18P | Brown-out detection level for V18P; disable the bod channel before adjusting the level setting. $VTH_{BOD} = 1.2 * (BOD_LVL+1)/192$ | 0x107 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 929: BOD_LVL_CTRL2_REG (0x5000006C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 17:9 | R/W | BOD_LVL_V14 | Brown-out detection level for V14; disable the bod channel before adjusting the level setting. VTH_BOD = 1.2 * (BOD_LVL+1)/192 | 0xC7 |
| 8:0 | R/W | BOD_LVL_V18F | Brown-out detection level for V18F; disable the bod channel before adjusting the level setting. VTH_BOD = 1.2 * (BOD_LVL+1)/192 | 0x107 |

Table 930: P0_PAD_LATCH_REG (0x50000070)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------------|
| 31:0 | R/W | P0_LATCH_EN | Direct write to the individual pad latching signals. Latches the control signals of the pads for state retention in powerdown mode. 0 = Control signals are retained 1 = Latch is transparent, pad can be recontrolled | 0xFFFF FFFF |

Table 931: P0_SET_PAD_LATCH_REG (0x50000074)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 31:0 | RWS | P0_SET_LATCH_E_N | Direct Set of the marked bits. Reading returns 0x0. | 0x0 |

Table 932: P0_RESET_PAD_LATCH_REG (0x50000078)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------------|---|-------|
| 31:0 | RW1C | P0_RESET_LATCH_EN | Direct Reset of the marked bits. Reading returns 0x0. | 0x0 |

Table 933: P1_PAD_LATCH_REG (0x5000007C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-----------|
| 22:0 | R/W | P1_LATCH_EN | Direct write to the individual pad latching signals. Latches the control signals of the pads for state retention in powerdown mode. 0 = Control signals are retained 1 = Latch is transparent, pad can be recontrolled | 0x7FFF FF |

Table 934: P1_SET_PAD_LATCH_REG (0x50000080)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 22:0 | RWS | P1_SET_LATCH_E_N | Direct Set of the marked bits. Reading returns 0x0. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 935: P1_RESET_PAD_LATCH_REG (0x50000084)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------------|---|-------|
| 22:0 | RW1C | P1_RESET_LATCH _EN | Direct Reset of the marked bits. Reading returns 0x0. | 0x0 |

Table 936: BOD_STATUS_REG (0x50000090)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|--|-------|
| 6 | R | BOD_V14 | 1: below trigger level (BOD event) 0: above trigger level | 0x0 |
| 5 | R | BOD_V18F | 1: below trigger level (BOD event) 0: above trigger level | 0x0 |
| 4 | R | BOD_VDD | 1: below trigger level (BOD event) 0: above trigger level | 0x0 |
| 3 | R | BOD_V18P | 1: below trigger level (BOD event) 0: above trigger level | 0x0 |
| 2 | R | BOD_V18 | 1: below trigger level (BOD event) 0: above trigger level | 0x0 |
| 1 | R | BOD_V30 | 1: below trigger level (BOD event) 0: above trigger level | 0x0 |
| 0 | R | BOD_VBAT | 1: below trigger level (BOD event) 0: above trigger level | 0x0 |

Table 937: POR_VBAT_CTRL_REG (0x50000094)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|--|-------|
| 13 | R/W | POR_VBAT_MASK_N | Enables propagation of the generated POR | 0x1 |
| 12 | R/W | POR_VBAT_ENABLE | Enables generation of the POR | 0x1 |
| 11:8 | R/W | POR_VBAT_HYST_LOW | Controls hysteresis of POR. 20mV per step. Must be set to 0x2 when thres_ctrl_low is set to 0xf. | 0x2 |
| 7:4 | R/W | POR_VBAT_THRES_HIGH | High-side (PTAT) threshold contribution: Level --> Threshold 0x0 --> 1.25V 0x1 --> 1.27V 0x2 --> 1.29V 0x3 --> 1.31V 0x4 --> 1.44V | 0x6 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------|--|-------|
| | | | 0x5 --> 1.49V 0x6 --> 1.53V 0x7 --> 1.58V 0x8 --> 1.63V 0x9 --> 1.68V 0xA --> 1.73V 0xB --> 1.78V 0xC --> 1.83V 0xD --> 1.87V 0xE --> 1.92V 0xF --> 1.97V | |
| 3:0 | R/W | POR_VBAT_THRES_LOW | Low-side (CTAT) threshold contribution Level --> Threshold 0xC --> 1.25V 0xC --> 1.27V 0xC --> 1.29V 0xC --> 1.31V 0x0 --> 1.44V 0x1 --> 1.49V 0x2 --> 1.53V 0x3 --> 1.58V 0x4 --> 1.63V 0x5 --> 1.68V 0x6 --> 1.73V 0x7 --> 1.78V 0x8 --> 1.83V 0x9 --> 1.87V 0xA --> 1.92V 0xB --> 1.97V 0xF --> 1.63V; use only with POR_VBAT_THRES_LOW=0x6 and POR_VBAT_THRES_HYST=0x2 | 0xF |

Table 938: POR_PIN_REG (0x500000098)

| 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 is always active High | 0x0 |
| 6 | R/W | - | | 0x0 |
| 5:0 | R/W | POR_PIN_SELECT | 0x00: P0_00 ... 0x1f: P0_31 0x20: P1_00 ... | 0x3F |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 0x36: P1_22 0x37 to 0x3E: reserved 0x3F: POR generation disabled | |

Table 939: POR_TIMER_REG (0x5000009C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------|---|-------|
| 6:0 | R/W | POR_TIME | Time for the PORReset to happen. Formula: Time = POR_TIME x 4096 x RC32 clock period Default value: ~3 seconds | 0x18 |

Table 940: LDO_VDDD_HIGH_CTRL_REG (0x500000A0)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------------------|---|-------|
| 5:4 | R/W | - | | 0x0 |
| 3 | R/W | LDO_VDDD_HIGH_LOW_ZOUT_DISABLE | Disables the low Zout switch. The low Zout switch pulls the output of the LDO to ground. When 0, the output of the LDO is pulled to ground when the LDO is disabled. When 1, the output of the LDO remains floating when the LDO is disabled. | 0x0 |
| 2 | R/W | LDO_VDDD_HIGH_STATIC_LOAD_ENABLE | Enables a static load of approx. 10 uA at the output of the LDO VDDD_HIGH. | 0x0 |
| 1 | R/W | LDO_VDDD_HIGH_ENABLE | 0: LDO VDDD_HIGH off, 1: LDO VDDD_HIGH on. | 0x0 |
| 0 | R/W | LDO_VDDD_HIGH_VREF_HOLD | 0: Indicates that the reference input is tracked, 1: Indicates that the reference input is sampled. | 0x0 |

Table 941: BIAS_VREF_SEL_REG (0x500000A4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------------|---|-------|
| 7:4 | R/W | BIAS_VREF_RF2_SEL | same coding as BIAS_VREF_RF1_SEL. | 0xB |
| 3:0 | R/W | BIAS_VREF_RF1_SEL | Vref_code Vref_Voltage (mV) 0:900 1:930 2:960 3:990 4:1020 5:1050 6:1080 7:1110 8:1140 9:1170 | 0xB |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 10:1200 11:1230 12:1260 13:1290 14:1320 15:1350 | |

Table 942: RESET_STAT_REG (0x500000BC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|--|-------|
| 5 | R/W | CMAC_WDOGRESET_STAT | Indicates that a CMAC-Watchdog timeout has happened. Note that it is also set when a PORReset has happened. | 0x1 |
| 4 | R/W | SWD_HWRESET_STAT | Indicates that a write to SWD_RESET_REG has happened. Note that it is also set when a PORReset has happened. | 0x1 |
| 3 | R/W | WDOGRESET_STAT | Indicates that a Watchdog timeout has happened. Note that it is also set when a PORReset has happened. | 0x1 |
| 2 | R/W | SWRESET_STAT | Indicates that a SW Reset has happened | 0x1 |
| 1 | R/W | HWRESET_STAT | Indicates that a HW Reset has happened | 0x1 |
| 0 | R/W | PORESET_STAT | Indicates that a PowerOn Reset has happened. All bitfields of RESET_STAT_REG should be read (in order to check the source of reset) and then cleared to '0', allowing thus the HW to automatically set to '1' the proper bitfields during the next reset event. | 0x1 |

Table 943: RAM_PWR_CTRL_REG (0x500000C0)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|--|-------|
| 15:14 | R/W | RAM8_PWR_CTRL | See description of RAM1_PWR_CTRL. | 0x0 |
| 13:12 | R/W | RAM7_PWR_CTRL | See description of RAM1_PWR_CTRL. | 0x0 |
| 11:10 | R/W | RAM6_PWR_CTRL | See description of RAM1_PWR_CTRL. | 0x0 |
| 9:8 | R/W | RAM5_PWR_CTRL | See description of RAM1_PWR_CTRL. | 0x0 |
| 7:6 | R/W | RAM4_PWR_CTRL | See description of RAM1_PWR_CTRL. | 0x0 |
| 5:4 | R/W | RAM3_PWR_CTRL | See description of RAM1_PWR_CTRL. | 0x0 |
| 3:2 | R/W | RAM2_PWR_CTRL | See description of RAM1_PWR_CTRL. | 0x0 |
| 1:0 | R/W | RAM1_PWR_CTRL | Power state control of the individual RAMs. May only change when the memory isn't accessed. When PD_MEM_IS_UP: 0x0: Normal operation 0x1: Normal operation 0x2: Retained (no access possible) 0x3: Off (memory content corrupted) When PD_MEM_IS_DOWN: | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|--|-------|
| | | | 0x0: Retained 0x1: Off (memory content corrupted) 0x2: Retained 0x3: Off (memory content corrupted) | |

Table 944: SECURE_BOOT_REG (0x500000CC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------------------|---|-------|
| 7 | R/W | PROT_QSPI_KEY_READ | This bit will permanently disable CPU read capability at OTP offset 0x00000B00 and for the complete segment | 0x0 |
| 6 | R/W | PROT_QSPI_KEY_WRITE | This bit will permanently disable ANY write capability at OTP offset 0x00000B00 and for the complete segment | 0x0 |
| 5 | R/W | PROT_AES_KEY_R_EAD | This bit will permanently disable CPU read capability at OTP offset 0x00000A00 and for the complete segment | 0x0 |
| 4 | R/W | PROT_AES_KEY_W_RITE | This bit will permanently disable ANY write capability at OTP offset 0x00000A00 and for the complete segment | 0x0 |
| 3 | R/W | PROT_SIG_KEY_W_RITE | This bit will permanently disable ANY write capability at OTP offset 0x000008C0 and for the complete segment | 0x0 |
| 2 | R/W | FORCE_CMAC_DE_BUGGER_OFF | This bit will permanently disable the CMAC debugger | 0x0 |
| 1 | R/W | FORCE_DEBUGGER_OFF | Follows the respective OTP flag value. Its value is updated by the BootROM code. 1: The system debugger SWD is totally disabled. 0: The system debugger is enabled with DEBUGGER_ENABLE | 0x0 |
| 0 | R/W | SECURE_BOOT | Follows the respective OTP flag value. Its value is updated by the BootROM code. 1: system is a secure system supporting secure boot 0: system is not supporting secure boot | 0x0 |

Table 945: DISCHARGE_RAIL_REG (0x500000D4)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|---|-------|
| 2 | R/W | RESET_V18P | 1: Enables immediate discharging of the V18P rail. Note that the source is not disabled. 0: disable immediate discharging of the V18P rail. This bit is ORed with the automatic function controlled by PMU_RESET_RAIL_REG.RESET_V18P | 0x0 |
| 1 | R/W | RESET_V18 | 1: Enables immediate discharging of the V18 rail. Note that the source is not disabled. 0: disable immediate discharging of the V18 rail. This bit is ORed with the automatic function controlled by | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------|---|-------|
| | | | PMU_RESET_RAIL_REG.RESET_V18 | |
| 0 | R/W | RESET_V14 | 1: Enables immediate discharging of the V14 rail. Note that the source is not disabled. 0: disable immediate discharging of the V14 rail. This bit is ORed with the automatic function controlled by PMU_RESET_RAIL_REG.RESET_V14 | 0x0 |

Table 946: ANA_STATUS_REG (0x500000EC)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|--|-------|
| 14 | R | COMP_VBUS_HIGH | COMP_VBUS_HIGH = 1 -> VBUS > 4V | 0x0 |
| 13 | R | COMP_VBUS_LOW | COMP_VBUS_LOW = 1 -> VBUS > 3.4V | 0x0 |
| 12 | R | COMP_VBAT_HIGH | COMP_VBAT_HIGH = 1 -> VBAT > 2.5V | 0x0 |
| 11 | R | COMP_VBAT_LOW | COMP_VBAT_LOW = 1 -> VBAT > 1.667V | 0x0 |
| 10 | R | COMP_VDD_OK | COMP_VDD_OK = 1 -> VDD > 1.125V | 0x0 |
| 9 | R | VBUS_AVAILABLE | High when VBUS > (VBAT + 150 mV). Hysteresis is approx. 40 mV | 0x0 |
| 8 | R | BANDGAP_OK | When high bandgap is active | 0x0 |
| 7 | R | LDO_3V0_VBAT_OK | When high LDO_VBAT is active | 0x0 |
| 6 | R | LDO_3V0_VBUS_OK | When high LDO_VBUS is active | 0x0 |
| 5 | R | LDO_1V8P_OK | When high LDO_IO2 is active | 0x0 |
| 4 | R | LDO_1V8_OK | When high LDO_IO is active | 0x0 |
| 3 | R | LDO_RADIO_OK | When high LDO_RADIO is active | 0x0 |
| 2 | R | LDO_CORE_OK | When high LDO_CORE(LDO1V2) is active | 0x0 |
| 1 | R | LDO_VDD_HIGH_OK | When high the ADC LDO is active. This LDO also supplies part of the LRA | 0x0 |
| 0 | R | BOD_VIN_NOK | General output of the BOD to indicate that one of the monitored inputs is below the trigger-level. | 0x0 |

Table 947: POWER_CTRL_REG (0x500000F0)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|---|-------|
| 31:29 | R/W | VDD_SLEEP_LEVEL | Level setting for VDD rail when using sleep LDO 0x0: 0.75 V 0x1: 0.80 V 0x2: 0.85 V 0x3: 0.90 V 0x4: 0.95 V 0x5: 1.00 V 0x6: N.A. 0x7: N.A. | 0x0 |
| 28:25 | R/W | VDD_CLAMP_LEVEL | Level setting for VDD when using clamp | 0x4 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------------|---|-------|
| | | L | Typical output voltages (not regulated): 0x0: 1037 mV 0x1: 1005 mV 0x2: 978 mV 0x3: 946 mV 0x4: 1120 mV 0x5: 1089 mV 0x6: 1058 mV 0x7: 1030 mV 0x8: 952 mV 0x9: 918 mV 0xA: 889 mV 0xB: 861 mV 0xC: 862 mV 0xD: 828 mV 0xE: 798 mV 0xF: 706 mV (changed to be lower than ldo_core_ret level) | |
| 24 | R/W | CLAMP_3V0_VBAT_ENABLE | Enables (1) or disables (0) clamp that can supply V30 from VBAT | 0x0 |
| 23 | R/W | V18_LEVEL | Level setting for V18 rail 0x0: 1.2 V 0x1: 1.8 V | 0x1 |
| 22:20 | R/W | V14_LEVEL | Level setting for V14 rail 0x0: 1.20 V 0x1: 1.25 V 0x2: 1.30 V 0x3: 1.35 V 0x4: 1.40 V 0x5: 1.45 V 0x6: 1.50 V 0x7: 1.55 V | 0x4 |
| 19:18 | R/W | V30_LEVEL | Level setting for V30 rail 0x0: 3.0 V 0x1: 3.45 V 0x2: 3.3 V 0x3: 3.3 V | 0x0 |
| 17:16 | R/W | VDD_LEVEL | Level setting for VDD rail 0x0: 0.9 V 0x1: 1.0 V 0x2: 1.1 V 0x3: 1.2 V | 0x3 |
| 15 | R/W | LDO_3V0_REF | Selects reference source for V30 LDOs 0x0: VDD rail 0x1: Bandgap output | 0x0 |
| 14 | R/W | LDO_CORE_RET_ENABLE_SLEEP | Enables (1) or disables (0) LDO_CORE_RET in sleep mode | 0x1 |
| 13 | R/W | LDO_CORE_RET_E | Enables (1) or disables (0) LDO_CORE_RET in | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------------|---|-------|
| | | NABLE_ACTIVE | active mode | |
| 12 | R/W | LDO_CORE_ENABLER | Enables (1) or disables (0) LDO_CORE | 0x1 |
| 11 | R/W | LDO_3V0_RET_ENABLE_SLEEP | Enables (1) or disables (0) LDO_3V0_RET in sleep mode | 0x1 |
| 10 | R/W | LDO_3V0_RET_ENABLE_ACTIVE | Enables (1) or disables (0) LDO_3V0_RET in active mode | 0x0 |
| 9:8 | R/W | LDO_3V0_MODE | Controls for LDO_3V0 0x0: Disabled 0x1: LDO_VBAT enabled 0x2: LDO_VBUS enabled 0x3: Automatic selection of LDO_VBAT or LDO_VBUS | 0x3 |
| 7 | R/W | LDO_RADIO_ENABLE | Enables (1) or disables (0) LDO_RADIO | 0x1 |
| 6 | R/W | LDO_1V8_RET_ENABLE_SLEEP | Enables (1) or disables (0) LDO_1V8_RET in sleep mode | 0x1 |
| 5 | R/W | LDO_1V8_RET_ENABLE_ACTIVE | Enables (1) or disables (0) LDO_1V8_RET in active mode | 0x0 |
| 4 | R/W | LDO_1V8_ENABLE | Enables (1) or disables (0) LDO_1V8 | 0x1 |
| 3 | R/W | SW_1V8F_ENABLE_FORCE | Forces switch between V18P and V18F rails on | 0x1 |
| 2 | R/W | LDO_1V8P_RET_ENABLE_SLEEP | Enables (1) or disables (0) LDO_1V8P_RET in sleep mode | 0x1 |
| 1 | R/W | LDO_1V8P_RET_ENABLE_ACTIVE | Enables (1) or disables (0) LDO_1V8P_RET in active mode | 0x0 |
| 0 | R/W | LDO_1V8P_ENABLE | Enables (1) or disables (0) LDO_1V8P | 0x1 |

Table 948: PMU_SLEEP_REG (0x500000F4)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|---|-------|
| 19 | R/W | - | | 0x0 |
| 18 | R/W | CLAMP_VDD_WKUP_MAX | Forces the VDD clamp voltage to its maximum value when waking up from sleep. | 0x0 |
| 17 | R/W | ULTRA_FAST_WAKEUP | Allows the core to start running on the RC32M while the PMU is still waiting for supplies to settle to the final value. Only use in combination with FAST_WAKEUP and 0.9 V on VDD during sleep. | 0x0 |
| 16 | R/W | FAST_WAKEUP | Speeds up the wakeup process by enabling all LDOs simultaneously instead of in staggered order. Only use if all voltages have been retained during sleep. | 0x1 |
| 15:12 | R/W | BOD_SLEEP_INTEGRAL | This is a value defining the interval every which Brown Out Detection is activated to check on the power rails voltage. The value represents BG_REFRESH_INTERVALS | 0x0 |
| 11:0 | R/W | BG_REFRESH_INTERVAL | This is a value defining the interval every which the Bandgap will be activated for refresh. The value | 0x80 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | represents ticks of lp_clk/64 e.g. 30,5 us * 64 = 1,9 ms. | |

Table 949: PMU_TRIM_REG (0x500000F8)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------|--|-------|
| 15:12 | R/W | LDO_1V8_TRIM | Trim setting for LDO_1V8 Unsigned binary notation, trim range $\pm 10\%$ | 0x8 |
| 11:8 | R/W | LDO_1V8P_TRIM | Trim setting for LDO_1V8P Unsigned binary notation, trim range $\pm 10\%$ | 0x8 |
| 7:4 | R/W | LDO_SUPPLY_VBAT_TRIM | Trim setting for LDO_SUPPLY_VBAT Sign-magnitude notation, trim range $\pm 10\%$ | 0x0 |
| 3:0 | R/W | LDO_SUPPLY_VBUS_TRIM | Trim setting for LDO_SUPPLY_VBUS Sign-magnitude notation, trim range $\pm 10\%$ | 0x0 |

Table 950: CLK_FREQ_TRIM_REG (0x50010000)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------|---|-------|
| 29:20 | R/W | XTAL32M_START | Xtal frequency trimming register - START phase of startup 0x2BF = lowest frequency (high load capacitance) 0x000 = highest frequency (low load capacitance) Cload = $5.0p + 6.09p \times \text{XTAL32M_TRIM}/0x2BF$ - this includes the PCB parasitic capacitances of the reference desing | 0x2BF |
| 19:10 | R/W | XTAL32M_RAMP | Xtal frequency trimming register - RAMP phase of startup. 0x2BF = lowest frequency (high load capacitance) 0x000 = highest frequency (low load capacitance) Cload = $5.0p + 6.09p \times \text{XTAL32M_TRIM}/0x2BF$ - this includes the PCB parasitic capacitances of the reference desing | 0x2BF |
| 9:0 | R/W | XTAL32M_TRIM | Xtal frequency trimming register. 0x2BF = lowest frequency (high load capacitance) 0x000 = highest frequency (low load capacitance) Cload = $5.0p + 6.09p \times \text{XTAL32M_TRIM}/0x2BF$ - this includes the PCB parasitic capacitances of the reference desing | 0x170 |

Table 951: TRIM_CTRL_REG (0x50010010)

| 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 reaches this value, the signal XTAL_SETTLE_READY will be set | 0x5 |
| 7:6 | R/W | XTAL_TRIM_SELECT | Select which source controls the XTAL trimming 0b00: xtal counter. Starts | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|---|-------|
| | | | <p>CLK_FREQ_TRIM_REG[XTAL32M_START] after COUNT_N * 32 xtal pulses trim is changed to CLK_FREQ_TRIM_REG[XTAL32M_TRIM].</p> <p>0b01: xtal OK filter. Starts with CLK_FREQ_TRIM_REG[XTAL32M_START], when xtal is ramping is changed to CLK_FREQ_TRIM_REG[XTAL32M_TRIM].</p> <p>0b10: statically forced off. Only uses CLK_FREQ_TRIM_REG[XTAL32M_TRIM].</p> <p>0b11: xtal OK filter, 2 stage. Starts with CLK_FREQ_TRIM_REG[XTAL32M_START] switches to CLK_FREQ_TRIM_REG[XTAL32M_RAMP] after timeout (sw1='1', XTAL32M_CTRL0_REG[XTAL32M_SW_DELAY]), and switches to CLK_FREQ_TRIM_REG[XTAL32M_TRIM] when sw2='1'.</p> | |
| 5:0 | R/W | XTAL_COUNT_N | <p>Defines the number of XTAL cycles to be counted, before the xtal trimming is applied, in steps of 32.</p> <p>0x01: 32</p> <p>0x02: 64</p> <p>0x3f: 2016</p> | 0x22 |

Table 952: XTALRDY_CTRL_REG (0x50010018)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|--|-------|
| 8 | R/W | XTALRDY_CLK_SEL | XTALRDY IRQ timer clock selection: 0: 32KHz 1: 256kHz | 0x1 |
| 7:0 | R/W | XTALRDY_CNT | Number of 32kHz or 256kHz cycles between the crystal is enabled, and the XTALRDY_IRQ is fired. Frequency set by XTALRDY_CLK_SEL. 0x00: no interrupt | 0x0 |

Table 953: XTALRDY_STAT_REG (0x5001001C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 15:8 | R | XTALRDY_COUNT | Current value of IRQ counter | 0x0 |
| 7:0 | R | XTALRDY_STAT | Value of IRQ counter when trimming is switched from RAMP to TRIM | 0x0 |

Table 954: XTAL32M_CTRL0_REG (0x50010030)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------------------|-----------------------------------|-------|
| 31 | - | - | | 0x0 |
| 30 | R/W | XTAL32M_DXTAL_SYSPLL_ENABLE | Enables DXTAL for the system PLL. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------------|---|-------|
| 29 | - | - | | 0x0 |
| 28:26 | - | - | | 0x2 |
| 25 | - | - | | 0x1 |
| 24:22 | - | - | | 0x6 |
| 21 | - | - | | 0x0 |
| 20:18 | - | - | | 0x0 |
| 17:15 | R/W | XTAL32M_CORE_C UR_SET | Core current trim setting. 0x0: min current ... 0x3: default current ... 0x6 max amplitude 0x7 is equal to 0x3. | 0x5 |
| 14 | - | - | | 0x1 |
| 13 | - | - | | 0x1 |
| 12 | - | - | | 0x0 |
| 11 | - | - | | 0x0 |
| 10 | - | - | | 0x1 |
| 9 | - | - | | 0x1 |
| 8 | - | - | | 0x0 |
| 7 | - | - | | 0x1 |
| 6 | - | - | | 0x0 |
| 5 | - | - | | 0x1 |
| 4 | - | - | | 0x1 |
| 3 | R/W | XTAL32M_RCOSC_ CALIBRATE | Request an RC-oscillator calibration. If set, calibration will be started after xtal startup. | 0x0 |
| 2 | - | - | | 0x1 |
| 1 | R/W | XTAL32M_RCOSC_ XTAL_DRIVE | Enable drive of crystal by RCOSC, needed for fast startup | 0x0 |
| 0 | R/W | XTAL32M_CXCOMP_ _ENABLE | Enable the shunt-capacitance compensation amplifier circuit (OSF BOOST). | 0x0 |

Table 955: XTAL32M_CTRL1_REG (0x50010034)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------------------|--|-------|
| 31 | - | - | | 0x0 |
| 30:28 | R/W | XTAL32M_STARTU P_TDISCHARGE | Discharge time. 0x0: disable 0x1: 8 us 0x2: 4 us 0x3: 2 us 0x4: 1 us 0x5: 1/2 us | 0x7 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------------|---|-------|
| | | | 0x6: 1/8 us 0x7: 1/32 us | |
| 27 | - | - | | 0x0 |
| 26:24 | R/W | XTAL32M_STARTUP_TSETTLE | Settle time. 0x0: 16 us 0x1: 8 us 0x2: 4 us 0x3: 2 us 0x4: 1 us 0x5: 1/2 us 0x6: 1/4 us 0x7: 1/8 us | 0x5 |
| 23 | R/W | XTAL32M_XTAL_ENABLE | Enable xtal (startup) or enable xtal block (software control mode) - testing only, to enable xtal, use PDC. | 0x0 |
| 22:13 | R/W | XTAL32M_STARTUP_TDRIVE LSB | LSB part of the sequence drive time. From 0 to 32us with steps of 1/32us. | 0x1F |
| 12:8 | R/W | XTAL32M_DRIVE_CYCLES | Number of sequences to drive at startup. | 0x8 |
| 7:5 | R/W | XTAL32M_STARTUP_TDRIVE | Drive time of the sequence. 0x0: 32 us 0x1: 16 us 0x2: 8 us 0x3: 4 us 0x4: 2 us 0x5: 1 us 0x6: 1/2 us 0x7: 1/4 us | 0x5 |
| 4:0 | R/W | XTAL32M_RCOSC_SYNC_DELAY_TRIM | Synchronization mode delay trim.. | 0x4 |

Table 956: XTAL32M_CTRL2_REG (0x50010038)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------------------|--|-------|
| 31:24 | - | - | | 0x0 |
| 23:22 | - | - | | 0x0 |
| 21:14 | R/W | XTAL32M_RCOSC_TRIM_SNS | Trim sensitivity used during calibration of the RC-oscillator. $\text{XTAL_RCOSC_TRIM_SNS} = 128 / (\text{M}_0 * \text{S})$ $\text{M}_0 = 8196$ $\text{S} = \text{sensitivity of RCOSC} (\sim 128 \text{ ppm/LSB})$ Using the internal frequency counter $\text{S} = (\text{FreqDetHigh} - \text{FreqDetLow}) / (\text{M} * \text{D})$ FreqDetHigh, FreqDetLow: the frequency counter reading (XTAL32M_STAT0_REG.FREQ_DET_OUT) at trim setting T0 and T1 | 0x79 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------------------|--|-------|
| | | | (XTAL32M_CTRL3_REG.XTAL32M_RCOSC_TRIM) D = T0 - T1 M -> see XTAL32M_CTRL2_REG.XTAL32M_FREQ_DET_LEN XTAL_RCOSC_TRIM_SNS = 64 * D / (FreqDetHigh-FreqDetLow), when XTAL32M_FREQ_DET_LEN = 1 | |
| 13:12 | R/W | XTAL32M_CXCOMP_PHI_TRIM | Phase correction for cxcomp circuit. | 0x0 |
| 11:3 | R/W | XTAL32M_CXCOMP_TRIM_CAP | Size of shunt capacitance compensation cap | 0x0 |
| 2:0 | - | - | | 0x7 |

Table 957: XTAL32M_CTRL3_REG (0x5001003C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------------|---|-------|
| 31 | - | - | | 0x0 |
| 30 | R/W | XTAL32M_RCOSC_TRIM_STROBE | Force RC-oscillator trim setting. | 0x0 |
| 29 | - | - | | 0x0 |
| 28 | - | - | | 0x0 |
| 27 | - | - | | 0x0 |
| 26 | - | - | | 0x0 |
| 25 | - | - | | 0x0 |
| 24 | - | - | | 0x0 |
| 23 | - | - | | 0x0 |
| 22 | R/W | XTAL32M_FREQ_DETECT_START | Force start frequency detector. | 0x0 |
| 21 | - | - | | 0x0 |
| 20 | - | - | | 0x0 |
| 19 | - | - | | 0x1 |
| 18 | R/W | XTAL32M_SW_CTRL_MODE | Enable all the software overrides, the state-machine will remain in IDLE. | 0x0 |
| 17:14 | R/W | XTAL32M_RCOSC_BAND_SELECT | Set RCOSC band select - apply with RCOSC_TRIM_STROBE | 0x5 |
| 13:4 | R/W | XTAL32M_RCOSC_TRIM | Set RCOSC trim (fine) - apply with RCOSC_TRIM_STROBE | 0x225 |
| 3 | - | - | | 0x0 |
| 2 | - | - | | 0x1 |
| 1 | - | - | | 0x0 |
| 0 | - | - | | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 958: XTAL32M_CTRL4_REG (0x50010040)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|-------------|-------|
| 31:24 | - | - | | 0x0 |
| 23 | - | - | | 0x0 |
| 22 | - | - | | 0x0 |
| 21 | - | - | | 0x0 |
| 20:19 | - | - | | 0x0 |
| 18:17 | - | - | | 0x0 |
| 16 | - | - | | 0x0 |
| 15 | - | - | | 0x0 |
| 14 | - | - | | 0x0 |
| 13 | - | - | | 0x0 |
| 12 | - | - | | 0x0 |
| 11 | - | - | | 0x0 |
| 10 | - | - | | 0x0 |
| 9 | - | - | | 0x0 |
| 8:5 | - | - | | 0x0 |
| 4 | - | - | | 0x0 |
| 3 | - | - | | 0x0 |
| 2 | - | - | | 0x0 |
| 1 | - | - | | 0x0 |
| 0 | - | - | | 0x0 |

Table 959: XTAL32M_STAT0_REG (0x50010050)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------------------------------|--|-------|
| 31:28 | R | XTAL32M_RCOSC_BAND_SELECT_STATUS | Currently selected band. | 0x0 |
| 27:18 | - | - | | 0x0 |
| 17 | - | - | | 0x0 |
| 16 | - | - | | 0x0 |
| 15 | R | XTAL32M_RCOSC_CALIBRATION_DONE | Signals that the calibration phase has been completed. | 0x0 |
| 14 | - | - | | 0x0 |
| 13:0 | - | - | | 0x0 |

Table 960: XTAL32M_STAT1_REG (0x50010054)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------|-------------|-------|
| 31:27 | - | - | | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------------|--|-------|
| 26 | - | - | | 0x0 |
| 25 | - | - | | 0x0 |
| 24 | - | - | | 0x0 |
| 23 | - | - | | 0x0 |
| 22:11 | - | - | | 0x0 |
| 10 | - | - | | 0x0 |
| 9 | - | - | | 0x0 |
| 8 | - | - | | 0x0 |
| 7:4 | R | XTAL32M_CAL_STA TE | Current state of the calibration state-machine. 0x0: CAL_DELAY - Delay for the biasing to settle 0x1: CAL_SYNC - Enables synchronization circuit 0x2: CAL_FREQDET - Enables the frequency detector 0x3: CAL_OFFSET - Evaluate OFFSET trimresult 0x4: CAL_MULT_SHIFT - Evaluate (delta) trimresult 0x5: CAL_TRIM - Generate new trimvalue 0x6: CAL_BAND_UPDWN - Delay after band change 0x7: CAL_END - Calibration ended 0x8: CAL_IDLE - Calibration IDLE (default state) | 0x0 |
| 3:0 | R | XTAL32M_STATE | Current state of the startup state-machine. 0x0: XTAL_WAIT_LDO - Allow for settling of the biasing 0x1: XTAL_DRIVE - Crystal is driven by rcosc 0x2: XTAL_DISCHARGE - Discharge loadcaps 0x3: XTAL_SETTLE - Allows for settling of the xtal signal 0x4: XTAL_SYNC - Restart RCOSC, and synchronize 0x5: XTAL_OVERLOAD_DETECT - amplitude detection mode by overload bit 0x6: XTAL_OVERLOAD_BLANK_SETTLE - settling delay for amplitude regulator when drive ends 0x7: XTAL_OVERLOAD_BLANK - blank glitch from enabling gm_current 0x8: XTAL_RUN - Startup ended 0x9: XTAL_SAMPLE - Delay before sample amplitude control 0xa: XTAL_SW2_MASK - Delay for masked SW2 0xb: XTAL_IDLE - Idle state (default) | 0x0 |

Table 961: PLL_SYS_CTRL1_REG (0x50010060)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|---|-------|
| 15 | R/W | - | | 0x0 |
| 14 | R/W | PLL_SEL_MIN_CUR | 0: VCO current read from min_current <5:0>, | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------------|--|-------|
| | | _INT | 1: VCO current is internally determined with a calibration algorithm. | |
| 13:12 | - | - | | 0x2 |
| 11 | R/W | PLL_PRE_DIV | PLL input divider (1: Indicates divide by 2). | 0x1 |
| 10:4 | R/W | PLL_N_DIV | PLL loop divider N (x means divide by x, 0 means divide by 1) | 0x6 |
| 3 | R/W | LDO_PLL_VREF_H_OLD | 0: Indicates that the reference input is tracked, 1: Indicates that the reference input is sampled. | 0x0 |
| 2 | R/W | LDO_PLL_ENABLE | 0: LDO PLL off, 1: LDO PLL on. | 0x0 |
| 1 | R/W | PLL_EN | 0: Power down 1: PLL on | 0x0 |
| 0 | R/W | - | | 0x0 |

Table 962: PLL_SYS_CTRL2_REG (0x50010064)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-------------|-------------|-------|
| 31:16 | R | - | | 0x0 |
| 15 | R/W | PLL_RECALIB | Recalibrate | 0x0 |
| 14:10 | - | - | | 0x3 |
| 9:5 | R/W | - | | 0x0 |
| 4:0 | - | - | | 0x0 |

Table 963: PLL_SYS_CTRL3_REG (0x50010068)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|---|-------|
| 15:13 | - | - | | 0x4 |
| 12 | - | - | | 0x0 |
| 11 | - | - | | 0x0 |
| 10 | - | - | | 0x0 |
| 9 | - | - | | 0x0 |
| 8 | - | - | | 0x0 |
| 7 | R/W | PLL_TEST_VCTR | 1: map loopfilter voltage on external pin <tbd> | 0x0 |
| 6:1 | R/W | PLL_MIN_CURRENT | VCO current trimming. | 0x38 |
| 0 | - | - | | 0x0 |

Table 964: PLL_SYS_STATUS_REG (0x50010070)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|---|-------|
| 15 | R | LDO_PLL_OK | 1: Indicates that LDO PLL is in regulation. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|--|-------|
| 14:12 | R | - | | 0x0 |
| 11 | R | PLL_CALIBRATION_END | Indicates that calibration has finished. | 0x0 |
| 10:5 | R | PLL_BEST_MIN_CU_R | Calibrated VCO current. | 0x0 |
| 4:1 | R | - | | 0x0 |
| 0 | R | PLL_LOCK_FINE | 1: PLL locked | 0x0 |

Table 965: CLK_COM_REG (0x50020904)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 17:16 | R/W | LCD_EXT_CLK_SEL | Select LCD external clock speed. 0x0: 1 Hz 0x1: 62.5 Hz 0x2: 125 Hz 0x3: off | 0x3 |
| 15:14 | R/W | SNC_DIV | Division factor for SNC, w.r.t. pclk setting 0x0 = divide by 1 0x1 = divide by 2 0x2 = divide by 4 0x3 = divide by 8 | 0x0 |
| 13 | R/W | - | | 0x0 |
| 12 | R/W | I2C2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 11 | R/W | I2C2_ENABLE | Enables the clock | 0x0 |
| 10 | R/W | I2C_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 9 | R/W | I2C_ENABLE | Enables the clock | 0x0 |
| 8 | R/W | SPI2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 7 | R/W | SPI2_ENABLE | Enables the clock | 0x0 |
| 6 | R/W | SPI_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 5 | R/W | SPI_ENABLE | Enables the clock | 0x0 |
| 4 | R/W | UART3_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 3 | R/W | UART3_ENABLE | Enables the clock | 0x0 |
| 2 | R/W | UART2_CLK_SEL | Selects the clock source 1 = DIV1 clock | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|-------------------|-------|
| | | | 0 = DIVN clock | |
| 1 | R/W | UART2_ENABLE | Enables the clock | 0x0 |
| 0 | R/W | UART_ENABLE | Enables the clock | 0x0 |

Table 966: SET_CLK_COM_REG (0x50020908)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 17:16 | RWS | LCD_EXT_CLK_SEL | Select LCD external clock speed. 0x0: 1 Hz 0x1: 62.5 Hz 0x2: 125 Hz 0x3: off | 0x0 |
| 15:14 | RWS | SNC_DIV | Division factor for SNC, w.r.t. pclk setting 0x0 = divide by 1 0x1 = divide by 2 0x2 = divide by 4 0x3 = divide by 8 | 0x0 |
| 13 | RWS | - | | 0x0 |
| 12 | RWS | I2C2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 11 | RWS | I2C2_ENABLE | Enables the clock | 0x0 |
| 10 | RWS | I2C_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 9 | RWS | I2C_ENABLE | Enables the clock | 0x0 |
| 8 | RWS | SPI2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 7 | RWS | SPI2_ENABLE | Enables the clock | 0x0 |
| 6 | RWS | SPI_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 5 | RWS | SPI_ENABLE | Enables the clock | 0x0 |
| 4 | RWS | UART3_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 3 | RWS | UART3_ENABLE | Enables the clock | 0x0 |
| 2 | RWS | UART2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 1 | RWS | UART2_ENABLE | Enables the clock | 0x0 |
| 0 | RWS | UART_ENABLE | Enables the clock | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 967: RESET_CLK_COM_REG (0x5002090C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 17:16 | RW1C | LCD_EXT_CLK_SEL | Select LCD external clock speed. 0x0: 1 Hz 0x1: 62.5 Hz 0x2: 125 Hz 0x3: off | 0x0 |
| 15:14 | RW1C | SNC_DIV | Division factor for SNC, w.r.t. pclk setting 0x0 = divide by 1 0x1 = divide by 2 0x2 = divide by 4 0x3 = divide by 8 | 0x0 |
| 13 | RW1C | - | | 0x0 |
| 12 | RW1C | I2C2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 11 | RW1C | I2C2_ENABLE | Enables the clock | 0x0 |
| 10 | RW1C | I2C_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 9 | RW1C | I2C_ENABLE | Enables the clock | 0x0 |
| 8 | RW1C | SPI2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 7 | RW1C | SPI2_ENABLE | Enables the clock | 0x0 |
| 6 | RW1C | SPI_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 5 | RW1C | SPI_ENABLE | Enables the clock | 0x0 |
| 4 | RW1C | UART3_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 3 | RW1C | UART3_ENABLE | Enables the clock | 0x0 |
| 2 | RW1C | UART2_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 1 | RW1C | UART2_ENABLE | Enables the clock | 0x0 |
| 0 | RW1C | UART_ENABLE | Enables the clock | 0x0 |

Table 968: CLK_PER_REG (0x50030C04)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 12:8 | R/W | MC_TRIG_DIV | Trigger divider for the motor controller | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| | | | 0x0: divide LP_CLK by 1 0x1: divide LP_CLK by 2 ... 0x1F: divide LP_CLK by 32 | |
| 7:3 | R/W | MC_CLK_DIV | Clock divider for the motor controller slot. The slots are clocked on (a PCLK synchronized version of) the LP clock, and can be further divided by this divider: 0x0: divide LP clock by 1 0x1: divide LP clock by 2 ... 0x1F: divide LP clock by 32 | 0x0 |
| 2 | R/W | MC_CLK_EN | Enables the clock | 0x0 |
| 1 | R/W | LRA_CLK_EN | Enables the clock | 0x0 |
| 0 | R/W | GPADC_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock/ 2 | 0x0 |

Table 969: SET_CLK_PER_REG (0x50030C08)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------|--|-------|
| 12:8 | RWS | MC_TRIG_DIV | Trigger divider for the motor controller 0x0: divide LP_CLK by 1 0x1: divide LP_CLK by 2 ... 0x1F: divide LP_CLK by 32 | 0x0 |
| 7:3 | RWS | MC_CLK_DIV | Clock divider for the motor controller slot. The slots are clocked on (a PCLK synchronized version of) the LP clock, and can be further divided by this divider: 0x0: divide LP clock by 1 0x1: divide LP clock by 2 ... 0x1F: divide LP clock by 32 | 0x0 |
| 2 | RWS | MC_CLK_EN | Enables the clock | 0x0 |
| 1 | RWS | LRA_CLK_EN | Enables the clock | 0x0 |
| 0 | RWS | GPADC_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock/ 2 | 0x0 |

Table 970: RESET_CLK_PER_REG (0x50030C0C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 12:8 | RW1C | MC_TRIG_DIV | Trigger divider for the motor controller 0x0: divide LP_CLK by 1 0x1: divide LP_CLK by 2 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| | | | ... 0x1F: divide LP_CLK by 32 | |
| 7:3 | RW1C | MC_CLK_DIV | Clock divider for the motor controller slot. The slots are clocked on (a PCLK synchronized version of) the LP clock, and can be further divided by this divider: 0x0: divide LP clock by 1 0x1: divide LP clock by 2 ... 0x1F: divide LP clock by 32 | 0x0 |
| 2 | RW1C | MC_CLK_EN | Enables the clock | 0x0 |
| 1 | RW1C | LRA_CLK_EN | Enables the clock | 0x0 |
| 0 | RW1C | GPADC_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock/ 2 | 0x0 |

Table 971: PCM_DIV_REG (0x50030C40)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 13 | R/W | PCM_SRC_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 12 | R/W | CLK_PCM_EN | Enable for the internally generated PCM clock The PCM_DIV must be set before or together with CLK_PCM_EN. | 0x0 |
| 11:0 | R/W | PCM_DIV | PCM clock divider. Minimum value is 0x2. | 0x0 |

Table 972: PCM_FDIV_REG (0x50030C44)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|---|-------|
| 15:0 | R/W | PCM_FDIV | These bits define the fractional division part of the PCM clock. The left most '1' defines the denominator, the number of '1' bits define the numerator. E.g. 0x0110 means 2/9, with a distribution of 1.0001.0000 0xeeee means 13/16, with a distribution of 1111.1110.1110.1110 | 0x0 |

Table 973: PDM_DIV_REG (0x50030C48)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|--|-------|
| 9 | R/W | PDM_MASTER_MODE | Master mode selection 0: slave mode 1: master mode | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| 8 | R/W | CLK_PDM_EN | Enable for the internally generated PDM clock The PDM_DIV must be set before or together with CLK_PDM_EN. | 0x0 |
| 7:0 | R/W | PDM_DIV | PDM clock divider | 0x0 |

Table 974: SRC_DIV_REG (0x50030C4C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| 8 | R/W | CLK_SRC_EN | Enable for the internally generated SRC clock The SRC_DIV must be set before or together with CLK_SRC_EN. | 0x0 |
| 7:0 | R/W | SRC_DIV | SRC clock divider | 0x0 |

Table 975: CLK_SYS_REG (0x50040500)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|--|-------|
| 5 | R/W | CLK_CHG_EN | Enables the clocks for the charger FSM block | 0x0 |
| 4 | R/W | LCD_RESET_REQ | Generates a SW reset towards the LCD controller. | 0x0 |
| 3:2 | R/W | - | | 0x0 |
| 1 | R/W | LCD_CLK_SEL | Selects the clock source 1 = DIV1 clock 0 = DIVN clock | 0x0 |
| 0 | R/W | LCD_ENABLE | Enables the clock | 0x0 |

Table 976: BATCHECK_REG (0x50040504)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------------|---|-------|
| 7 | R/W | BATCHECK_LOAD_ENABLE | Enable a current load on the battery. | 0x0 |
| 6:4 | R/W | BATCHECK_ILOAD | Set the current load to (ILOAD+1) mA. | 0x0 |
| 3:0 | R/W | BATCHECK_TRIM | Trim the current load with steps of 2.7% from -19.1% to +19.1%. 0: +0.0% , 8: -0% 1: +2.7% , 9: -2.7% 2: +5.5% , 10: -5.5% 3: +8.2% , 11: -8.2% 4: +10.9% , 12: -10.9% 5: +13.6% , 13: -13.6% 6: +16.4% , 14: -16.4% 7: +19.1% , 15: -19.1% | 0x0 |

42.30 Cache Controller Registers

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 977: Register map CACHE

| Address | Register | Description |
|------------|------------------------------|---|
| 0x100C0000 | CACHE_CTRL1_REG | Cache control register 1 |
| 0x100C0004 | CACHE_LNSIZECFG_REG | Cache line size configuration register |
| 0x100C0008 | CACHE_ASSOCCFG_REG | Cache associativity configuration register |
| 0x100C0020 | CACHE_CTRL2_REG | Cache control register 2 |
| 0x100C0028 | CACHE_MRM_HITS_REG | Cache MRM (Miss Rate Monitor) HITS register |
| 0x100C002C | CACHE_MRM_MISSES_REG | Cache MRM (Miss Rate Monitor) MISSES register |
| 0x100C0030 | CACHE_MRM_CTRL_REG | Cache MRM (Miss Rate Monitor) CONTROL register |
| 0x100C0034 | CACHE_MRM_TINT_REG | Cache MRM (Miss Rate Monitor) TIME INTERVAL register |
| 0x100C0038 | CACHE_MRM_MISSES_THRES_REG | Cache MRM (Miss Rate Monitor) THRESHOLD register |
| 0x100C003C | CACHE_MRM_HITS_THRESHOLD_REG | Cache MRM (Miss Rate Monitor) HITS THRESHOLD register |
| 0x100C0040 | CACHE_FLASH_REG | Cache Flash program size and base address register |
| 0x100C0050 | SWD_RESET_REG | SWD HW reset control register |

Table 978: CACHE_CTRL1_REG (0x100C0000)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|---|-------|
| 31:2 | - | - | | 0 |
| 1 | R/W | CACHE_RES1 | Reserved. Always keep 0. | 0 |
| 0 | R0/W | CACHE_FLUSH | Writing a '1' into this bit, flushes the contents of the tag memories which invalidates the content of the cache memory. The read of this bit is always '0'. Note: The flushing of the cache TAG memory takes 0x100 or 0x200 HCLK cycles for a Cache Data RAM size of 8 KB resp. 16 KB. | 0 |

Table 979: CACHE_LNSIZECFG_REG (0x100C0004)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|---|-------|
| 31:2 | - | - | | 0 |
| 1:0 | R/W | CACHE_LINE | Cache line size: 0: 8 bytes, 1: 16 bytes, 2: 32 bytes, 3: reserved. Note: Flush the cache just after the dynamic (run-time) reconfiguration of the cache with an 8 bytes | 0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | cache line size: write the value "01" into the cache control register CACHE_CTRL1_REG just after the write of the value "00" into the cache line size configuration register CACHE_LNSIZECFG_REG. | |

Table 980: CACHE_ASSOCCFG_REG (0x100C0008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-------------|--|-------|
| 31:2 | - | - | | 0 |
| 1:0 | R/W | CACHE_ASSOC | Cache associativity: 0: 1-way (direct mapped) 1: 2-way 2: 4-way 3: reserved. | 2 |

Table 981: CACHE_CTRL2_REG (0x100C0020)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:11 | - | - | | 0 |
| 10 | R/W | CACHE_CGEN | 0: Cache controller clock gating is not enabled. 1: Cache controller clock gating is enabled (enabling power saving). Note: This bit must be set to '0' (default) when setting the CACHE_FLUSH bit while executing from other than QSPI FLASH cached, e.g. from Booter or SYSRAM. | 0 |
| 9 | R/W | CACHE_WEN | 0: Cache Data and TAG memory read only. 1: Cache Data and TAG memory read/write. The TAG and Data memory are only updated by the cache controller. There is no HW protection to prevent unauthorized access by the ARM. Note: When accessing the memory mapped Cache Data and TAG memory (for debugging purposes) only 32 bits access is allowed to the Cache Data memory and only 16 bits access is allowed to the Cache TAG memory. | 0 |
| 8:0 | R/W | CACHE_LEN | Length of QSPI FLASH cacheable memory. N*64 KByte. N = 0 to 512 (max. of 32 Mbyte). Setting CACHE_LEN=0 disables the cache. Note 1: The max. relevant CACHE_LEN setting depends on the chosen Flash region (program) size. Note 2: The first block (CACHE_LEN=1) includes the memory space specified by CACHE_FLASH_REG[FLASH_REGION_OFFSET]. | 0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 982: CACHE_MRM_HITS_REG (0x100C0028)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------|------------------------------------|-------|
| 31:0 | R/W | MRM_HITS | Contains the amount of cache hits. | 0x0 |

Table 983: CACHE_MRM_MISSES_REG (0x100C002C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--------------------------------------|-------|
| 31:0 | R/W | MRM_MISSES | Contains the amount of cache misses. | 0x0 |

Table 984: CACHE_MRM_CTRL_REG (0x100C0030)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------------------|---|-------|
| 31:5 | - | - | | 0 |
| 4 | R/W | MRM_IRQ_HITS_THRES_STATUS | 0: No interrupt is generated. 1: Interrupt (pulse-sensitive) is generated because the number of cache hits reached the programmed threshold (threshold != 0). | 0 |
| 3 | R/W | MRM_IRQ_MISSES_THRES_STATUS | 0: No interrupt is generated. 1: Interrupt (pulse-sensitive) is generated because the number of cache misses reached the programmed threshold (threshold != 0). | 0 |
| 2 | R/W | MRM_IRQ_TINT_STATUS | 0: No interrupt is generated. 1: Interrupt (pulse-sensitive) is generated because the time interval counter reached the end (time interval != 0). | 0 |
| 1 | R/W | MRM_IRQ_MASK | 0: Disables interrupt generation. 1: Enables interrupt generation. Note: The Cache MRM generates a pulse-sensitive interrupt towards the ARM processor, | 0 |
| 0 | R/W | MRM_START | 0: Freeze the "misses/hits" counters and reset the time interval counter to the programmed value in CACHE_MRM_TINT_REG. 1: Enables the counters. Note: In case CACHE_MRM_CTRL_REG[MRM_START] is set to '1' and CACHE_MRM_TINT_REG (!=0) is used for the MRM interrupt generation, the time interval counter counts down (on a fixed reference clock of 16 MHz) until it's '0'. At that time CACHE_MRM_CTRL_REG[MRM_START] will be reset automatically to '0' by the MRM hardware and the MRM interrupt will be generated. | 0 |

Table 985: CACHE_MRM_TINT_REG (0x100C0034)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|----------|---|-------|
| 31:19 | - | - | | 0x0 |
| 18:0 | R/W | MRM_TINT | Defines the time interval for the monitoring in 32 MHz clock cycles. See also the description of CACHE_MRM_CTRL_REG[MRM_IRQ_TINT_STA] | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | TUS]. Note: When MRM_TINT=0 (unrealistic value), no interrupt will be generated. | |

Table 986: CACHE_MRM_MISSES_THRES_REG (0x100C0038)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|--|-------|
| 31:0 | R/W | MRM_MISSES_THRES | Defines the misses threshold to trigger the interrupt generation. See also the description of CACHE_MRM_CTRL_REG[MRM_IRQ_MISSES_THRESHOLD_STATUS]. Note: When MRM_MISSES_THRES=0 (unrealistic value), no interrupt will be generated. | 0x0 |

Table 987: CACHE_MRM_HITS_THRES_REG (0x100C003C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|----------------|--|-------|
| 31:0 | R/W | MRM_HITS_THRES | Defines the hits threshold to trigger the interrupt generation. See also the description of CACHE_MRM_CTRL_REG[MRM_IRQ_HITS_THRESHOLD_STATUS]. Note: When MRM_HITS_THRES=0 (unrealistic value), no interrupt will be generated. | 0x0 |

Table 988: CACHE_FLASH_REG (0x100C0040)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|---|--------|
| 31:16 | R/W | FLASH_REGION_BASE | These bits corresponds with the Flash region base address bits [31:16]. Default value is '0x1600'. The Flash region base address bits [31:25] are fixed to '0x16' and bits [17:16] are fixed to '0x0'. These register bits are retained. Note 1: The updated value takes effect only after a software reset. Note 2 The Flash region base address setting depends on the chosen Flash region size. | 0x1600 |
| 15:4 | R/W | FLASH_REGION_OFFSET | Flash region offset address (in words). This value is added to the Flash (CPU) address bits [13:2]. These register bits are retained. Note 1: The updated value takes effect only after a software reset. | 0x0 |
| 3 | R | - | | 0x0 |
| 2:0 | R/W | FLASH_REGION_SIZE | Flash region size. Default value is '6' (0.5 MBytes). 0 = 32 MBytes, 1 = 16 MBytes, 2 = 8 MBytes, | 0x6 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>3 = 4 MBytes, 4 = 2 MBytes, 5 = 1 MBytes, 6 = 0.5 MBytes, 7 = 0.25 MBytes.</p> <p>These register bits are retained.</p> <p>Note 1: The updated value takes effect only after a software reset.</p> <p>Note 2: See for the max. region (program) size the memory map.</p> | |

Table 989: SWD_RESET_REG (0x100C0050)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------------|---|-------|
| 31:1 | - | - | | 0 |
| 0 | R0/W | SWD_HW_RESET_REQ | <p>0: default.</p> <p>1: HW reset request (from the debugger tool). The register is automatically reset with a HW_RESET. This bit can only be accessed by the debugger software and not by the application.</p> | 0 |

42.31 Audio Unit Registers

Table 990: Register map APU

| Address | Register | Description |
|------------|-----------------|---------------------------|
| 0x50030600 | SRC1_CTRL_REG | SRC1 control register |
| 0x50030604 | SRC1_IN_FS_REG | SRC1 Sample input rate |
| 0x50030608 | SRC1_OUT_FS_REG | SRC1 Sample output rate |
| 0x5003060C | SRC1_IN1_REG | SRC1 data in 1 |
| 0x50030610 | SRC1_IN2_REG | SRC1 data in 2 |
| 0x50030614 | SRC1_OUT1_REG | SRC1 data out 1 |
| 0x50030618 | SRC1_OUT2_REG | SRC1 data out 2 |
| 0x5003061C | APU_MUX_REG | APU mux register |
| 0x50030620 | COEF10_SET1_REG | SRC coefficient 1,0 set 1 |
| 0x50030624 | COEF32_SET1_REG | SRC coefficient 3,2 set 1 |
| 0x50030628 | COEF54_SET1_REG | SRC coefficient 5,4 set 1 |
| 0x5003062C | COEF76_SET1_REG | SRC coefficient 7,6 set 1 |
| 0x50030630 | COEF98_SET1_REG | SRC coefficient 9,8 set 1 |
| 0x50030634 | COEF0A_SET1_REG | SRC coefficient 10 set 1 |
| 0x50030700 | PCM1_CTRL_REG | PCM1 Control register |
| 0x50030704 | PCM1_IN1_REG | PCM1 data in 1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Address | Register | Description |
|------------|---------------|-----------------|
| 0x50030708 | PCM1_IN2_REG | PCM1 data in 2 |
| 0x5003070C | PCM1_OUT1_REG | PCM1 data out 1 |
| 0x50030710 | PCM1_OUT2_REG | PCM1 data out 2 |

Table 991: SRC1_CTRL_REG (0x50030600)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------------|--|-------|
| 31:30 | R/W | SRC_PDM_DO_DEL | PDM_DO output delay line (typical) 0: no delay 1: 8 ns 2: 12 ns 3: 16 ns | 0 |
| 29:28 | R/W | SRC_PDM_MODE | PDM Output mode selection on PDM_DO1 00: No output 01: Right channel (data from SRC1_IN_REG) 10: Left channel (data from SRC2_IN_REG) 11: Left and Right channel | 0 |
| 27:26 | R/W | SRC_PDM_DI_DEL | PDM_DI input delay line (typical) 0: no delay 1: 4 ns 2: 8 ns 3: 12 ns | 0 |
| 25 | R0/W | SRC_OUT_FLOWCLR | Writing a 1 clears the SRC1_OUT Overflow/underflow bits 23-22. No more over/underflow indications while bit is 1. Keep 1 until the over/under flow bit is cleared | 0 |
| 24 | W | SRC_IN_FLOWCLR | Writing a 1 clears the SRC1_IN Overflow/underflow bits 21-20. No more over/underflow indications while bit is 1. Keep 1 until the over/under flow bit is cleared | 0 |
| 23 | R | SRC_OUT_UNFLOW | 1 = SRC1_OUT Underflow occurred | 0 |
| 22 | R | SRC_OUT_OVFLOW | 1 = SRC1_OUT Overflow occurred | 0 |
| 21 | R | SRC_IN_UNFLOW | 1 = SRC1_IN Underflow occurred | 0 |
| 20 | R | SRC_IN_OVFLOW | 1 = SRC1_IN Overflow occurred | 0 |
| 19 | R0/W | SRC_RESYNC | 1 = SRC will restart synchronisation | 0 |
| 18 | R | SRC_OUT_OK | SRC1_OUT Status 0: acquisition in progress 1: acquisition ready (In manual mode this bit is always 1) | 0 |
| 17:16 | R/W | SRC_OUT_US | SRC1_OUT UpSampling IIR filters setting 00: for sample rates up-to 48kHz 01: for sample rates of 96kHz 10: reserved 11: for sample rates of 192kHz | 0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------------|---|-------|
| 15 | - | - | | 0 |
| 14 | R/W | SRC_OUT_CAL_BY_PASS | SRC1_OUT1 upsampling filter bypass 0:Do not bypass 1:Bypass filter | 0 |
| 13 | R/W | SRC_OUT_AMODE | SRC1_OUT1 Automatic Conversion mode 0:Manual mode 1:Automatic mode | 0 |
| 12 | R/W | SRC_PDM_OUT_IN_V | Swap the left and the right output PDM channel | 0 |
| 11 | R/W | SRC_FIFO_DIRECTION | 0 = SRC fifo is used to store samples from memory to SRC 1 = SRC fifo is used to store sample from SRC to memory | 0x0 |
| 10 | R/W | SRC_FIFO_ENABLE | 0 = fifo disable. On each src request, one sample is serviced 1 = fifo enable. Fifo is used to store samples from / to src SRC supports only DMA burst size 4 when fifo is enable else no burst | 0x0 |
| 9 | R/W | SRC_OUT_DSD_MODE | 0 = SRC1 OUT PDM mode 1 = SRC1 OUT DSD mode | 0x0 |
| 8 | R/W | SRC_IN_DSD_MODE | 0: SRC1 IN PDM mode 1: SRC1 IN DSD mode | 0x0 |
| 7 | R/W | SRC_DITHER_DISABLE | Dithering feature 0: Enable 1: Disable | 0 |
| 6 | R | SRC_IN_OK | SRC1_IN status 0: Acquisition in progress 1: Acquisition ready | 0 |
| 5:4 | R/W | SRC_IN_DS | SRC1_IN UpSampling IIR filters setting 00: for sample rates up-to 48kHz 01: for sample rates of 96kHz 10: reserved 11: for sample rates of 192kHz | 0 |
| 3 | R/W | SRC_PDM_IN_INV | Swap the left and the right input PDM channel | 0 |
| 2 | R/W | SRC_IN_CAL_BYPASS | SRC1_IN upsampling filter bypass 0: Do not bypass 1: Bypass filter | 0 |
| 1 | R/W | SRC_IN_AMODE | SRC1_IN Automatic conversion mode 0: Manual mode 1: Automatic mode | 0 |
| 0 | R/W | SRC_EN | SRC1_IN and SRC1_OUT enable 0: disabled 1: enabled | 0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 992: SRC1_IN_FS_REG (0x50030604)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---|-------|
| 31:24 | - | - | | 0 |
| 23:0 | R/W | SRC_IN_FS | <p>SRC_IN Sample rate $SRC_IN_FS = SRC_DIV * 4096 * Sample_rate / 100$ Sample_rate upper limit is 192kHz. For 96kHz and 192kHz SRC_CTRLx_REG[SRC_IN_DS] must be set as shown below: (for SRC_DIV=1) Sample_rate SRC_IN_FS SRC_IN_DS Audio bandwidth 8000 Hz 0x050000 0 4000 Hz 11025 Hz 0x06E400 0 5512 Hz 16000 Hz 0x0A0000 0 8000 Hz 22050 Hz 0x0DC800 0 11025 Hz 32000 Hz 0x140000 0 16000 Hz 44100 Hz 0x1B9000 0 22050 Hz 48000 Hz 0x1E0000 0 24000 Hz 96000 Hz 0x1E0000 1 24000 Hz 192000 Hz 0x1E0000 3 24000 Hz</p> <p>In manual SRC mode, SRC_IN_FS can be set and adjusted to the desired sample rate at any time. In automatic mode the SRC returns the final sample rate as soon as SRC_IN_OK. Note that SRC_DS is not calculated in automatic mode and must be set manually automatic mode with Sample_rate of 96 and 192kHz.</p> | 0 |

Table 993: SRC1_OUT_FS_REG (0x50030608)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|------------|---|-------|
| 31:24 | - | - | | 0 |
| 23:0 | R/W | SRC_OUT_FS | <p>SRC_OUT Sample rate $SRC_OUT_FS = SRC_DIV * 4096 * Sample_rate / 100$ Sample_rate upper limit is 192kHz. For 96kHz and 192kHz SRC_CTRLx_REG[SRC_DS] must be set as shown below: (for SRC_DIV=1) Sample_rate SRC_OUT_FS SRC_OUT_DS Audio bandwidth 8000 Hz 0x050000 0 4000 Hz 11025 Hz 0x06E400 0 5512 Hz 16000 Hz 0x0A0000 0 8000 Hz 22050 Hz 0x0DC800 0 11025 Hz 32000 Hz 0x140000 0 16000 Hz 44100 Hz 0x1B9000 0</p> | 0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------|---|-------|
| | | | <p>22050 Hz 48000 Hz 0x1E0000 0 24000 Hz 96000 Hz 0x1E0000 1 24000 Hz 192000 Hz 0x1E0000 3 24000 Hz</p> <p>In manual SRC mode, SRC_OUT_FS can be set and adjusted to the desired sample rate at any time. In automatic mode the SRC returns the final sample rate as soon as SRC_OUT_OK. Note that SRC_DS is not calculated in automatic mode and must be set manually automatic mode with Sample_rate of 96 and 192kHz.</p> | |

Table 994: SRC1_IN1_REG (0x5003060C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|-------------|-------|
| 31:0 | R/W | SRC_IN | SRC1_IN1 | 0 |

Table 995: SRC1_IN2_REG (0x50030610)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|-------------|-------|
| 31:0 | R/W | SRC_IN | SRC1_IN2 | 0 |

Table 996: SRC1_OUT1_REG (0x50030614)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|-------------|-------|
| 31:0 | R | SRC_OUT | SRC1_OUT1 | 0 |

Table 997: SRC1_OUT2_REG (0x50030618)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|-------------|-------|
| 31:0 | R | SRC_OUT | SRC1_OUT2 | 0 |

Table 998: APU_MUX_REG (0x5003061C)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|--|-------|
| 6 | R/W | PDM1_MUX_IN | PDM1 input mux 0 = SRC1_MUX_IN 1 = PDM input | 0x0 |
| 5:3 | R/W | PCM1_MUX_IN | PCM1 input mux 0 = off 1 = SRC1 output | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-------------|---|-------|
| | | | 2 = PCM output registers | |
| 2:0 | R/W | SRC1_MUX_IN | SRC1 input mux 0 = off 1 = PCM output 2 = SRC1 input registers | 0x0 |

Table 999: COEF10_SET1_REG (0x50030620)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---------------|--------|
| 31:16 | R/W | SRC_COEF1 | coefficient 1 | 0x79A9 |
| 15:0 | R/W | SRC_COEF0 | coefficient 0 | 0x9278 |

Table 1000: COEF32_SET1_REG (0x50030624)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---------------|--------|
| 31:16 | R/W | SRC_COEF3 | coefficient 3 | 0x6D56 |
| 15:0 | R/W | SRC_COEF2 | coefficient 2 | 0x8B41 |

Table 1001: COEF54_SET1_REG (0x50030628)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---------------|--------|
| 31:16 | R/W | SRC_COEF5 | coefficient 5 | 0x9BC5 |
| 15:0 | R/W | SRC_COEF4 | coefficient 4 | 0xBE15 |

Table 1002: COEF76_SET1_REG (0x5003062C)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---------------|--------|
| 31:16 | R/W | SRC_COEF7 | coefficient 7 | 0x8C28 |
| 15:0 | R/W | SRC_COEF6 | coefficient 6 | 0x7E1A |

Table 1003: COEF98_SET1_REG (0x50030630)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|-----------|---------------|--------|
| 31:16 | R/W | SRC_COEF9 | coefficient 9 | 0x92D7 |
| 15:0 | R/W | SRC_COEF8 | coefficient 8 | 0x75E6 |

Table 1004: COEF0A_SET1_REG (0x50030634)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|----------------|--------|
| 15:0 | R/W | SRC_COEF10 | coefficient 10 | 0x41F2 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 1005: PCM1_CTRL_REG (0x50030700)

| Bit | Mode | Symbol | Description | Reset |
|-------|------|--------------|---|-------|
| 31:20 | R/W | PCM_FSC_DIV | PCM Framesync divider, Values 7-0xFFFF. To divide by N, write N-1. (Minimum value N-1=7 for 8 bits PCM_FSC) Note if PCM_CLK_BIT=1, N must always be even | 0x0 |
| 19:17 | R | - | | 0x0 |
| 16 | R/W | PCM_FSC_EDGE | 0: shift channels 1, 2, 3, 4, 5, 6, 7, 8 after PCM_FSC edge 1: shift channels 1, 2, 3, 4 after PCM_FSC edge shift channels 5, 6, 7, 8 after opposite PCM_FSC edge | 0x0 |
| 15:11 | R/W | PCM_CH_DEL | Channel delay in multiples of 8 bits | 0x0 |
| 10 | R/W | PCM_CLK_BIT | 0:One clock cycle per data bit 1:Two clock cycles per data bit | 0x0 |
| 9 | R/W | PCM_FSCINV | 0: PCM FSC 1: PCM FSC inverted | 0x0 |
| 8 | R/W | PCM_CLKINV | 0:PCM CLK 1:PCM CLK inverted | 0x0 |
| 7 | R/W | PCM_PPOD | 0:PCM DO push pull 1:PCM DO open drain | 0x0 |
| 6 | R/W | PCM_FSCDEL | 0:PCM FSC starts one cycle before MSB bit 1:PCM FSC starts at the same time as MSB bit | 0x0 |
| 5:2 | R/W | PCM_FSCLEN | 0:PCM FSC length equal to 1 data bit N:PCM FSC length equal to N*8 | 0x0 |
| 1 | R/W | PCM_MASTER | 0:PCM interface in slave mode 1:PCM interface in master mode | 0x0 |
| 0 | R/W | PCM_EN | 0:PCM interface disabled 1:PCM interface enabled | 0x0 |

Table 1006: PCM1_IN1_REG (0x50030704)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--------------------|-------|
| 31:0 | R | PCM_IN | PCM1_IN1 bits 31-0 | 0x0 |

Table 1007: PCM1_IN2_REG (0x50030708)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------|--------------------|-------|
| 31:0 | R | PCM_IN | PCM1_IN2 bits 31-0 | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 1008: PCM1_OUT1_REG (0x5003070C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|---------------------|-------------|
| 31:0 | R/W | PCM_OUT | PCM1_OUT1 bits 31-0 | 0xFFFF FFFF |

Table 1009: PCM1_OUT2_REG (0x50030710)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------|---------------------|-------------|
| 31:0 | R/W | PCM_OUT | PCM1_OUT2 bits 31-0 | 0xFFFF FFFF |

42.32 Analog Miscellaneous Registers

Table 1010: Register map ANAMISC

| Address | Register | Description |
|------------|---------------------------------|---|
| 0x50030B10 | CLK_REF_SEL_REG | Select clock for oscillator calibration |
| 0x50030B14 | CLK_REF_CNT_REG | Count value for oscillator calibration |
| 0x50030B18 | CLK_REF_VAL_REG | DIVN reference cycles, lower 16 bits |

Table 1011: CLK_REF_SEL_REG (0x50030B10)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|----------------|--|-------|
| 7:5 | R/W | CAL_CLK_SEL | Select reference clock input to be used in calibration: 0x0 : DIVN clock 0x1 : RC32K 0x2 : RC32M 0x3 : XTAL32K 0x4 : RCOSC 0x5, 0x6, 0x7: Reserved | 0x0 |
| 4 | 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 |
| 3 | R/W | REF_CAL_START | Writing a '1' starts a calibration. This bit is cleared when calibration is finished, and CLK_REF_VAL is ready. | 0x0 |
| 2:0 | R/W | REF_CLK_SEL | Select clock input for calibration: 0x0 : RC32K 0x1 : RC32M 0x2 : XTAL32K 0x3 : RCX 0x4 : RCOSC | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 1012: CLK_REF_CNT_REG (0x50030B14)

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

Table 1013: CLK_REF_VAL_REG (0x50030B18)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:0 | R | XTAL_CNT_VAL | Returns the number of DIVN clock cycles counted during the calibration time, defined with REF_CNT_VAL | 0x0 |

42.33 Crypto-Engine Registers

Table 1014: Register map AES_HASH

| Address | Register | Description |
|------------|----------------------|--|
| 0x30040000 | CRYPTO_CTRL_REG | Crypto Control register |
| 0x30040004 | CRYPTO_START_REG | Crypto Start calculation |
| 0x30040008 | CRYPTO_FETCHADDR_REG | Crypto DMA fetch register |
| 0x3004000C | CRYPTO_LEN_REG | Crypto Length of the input block in bytes |
| 0x30040010 | CRYPTO_DESTADDR_REG | Crypto DMA destination memory |
| 0x30040014 | CRYPTO_STATUS_REG | Crypto Status register |
| 0x30040018 | CRYPTO_CLRIRQ_REG | Crypto Clear interrupt request |
| 0x3004001C | CRYPTO_MREG0_REG | Crypto Mode depended register 0 |
| 0x30040020 | CRYPTO_MREG1_REG | Crypto Mode depended register 1 |
| 0x30040024 | CRYPTO_MREG2_REG | Crypto Mode depended register 2 |
| 0x30040028 | CRYPTO_MREG3_REG | Crypto Mode depended register 3 |
| 0x30040100 | CRYPTO_KEYS_START | Crypto First position of the AES keys storage memory |

Table 1015: CRYPTO_CTRL_REG (0x30040000)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|-----------------|--|-------|
| 17 | R/W | CRYPTO_AES_KEXP | It forces (active high) the execution of the key expansion process with the starting of the AES encryption/decryption process. The bit will be | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-------|------|---------------------|--|-------|
| | | | cleared automatically by the hardware, after the completion of the AES key expansion process. | |
| 16 | R/W | CRYPTO_MORE_IN | 0 : Define that this is the last input block. When the current input is consumed by the crypto engine and the output data is written to the memory, the calculation ends (CRYPTO_INACTIVE goes to one). 1 : The current input data block is not the last. More input data will follow. When the current input is consumed, the engine stops and waits for more data (CRYPTO_WAIT_FOR_IN goes to one). | 0x0 |
| 15:10 | R/W | CRYPTO_HASH_OUT_LEN | The number of bytes minus one of the hash result which will be saved at the memory by the DMA. In relation with the selected hash algorithm the accepted values are: MD5: 0..15 -> 1-16 bytes SHA-1: 0..19 -> 1-20 bytes SHA-256: 0..31 -> 1 - 32 bytes SHA-256/224: 0..27 -> 1- 28 bytes SHA-384: 0..47 -> 1 - 48 bytes SHA-512: 0..63 -> 1 - 64 bytes SHA-512/224: 0..27 -> 1- 28 bytes SHA-512/256: 0..31 -> 1 - 32 bytes | 0x0 |
| 9 | R/W | CRYPTO_HASH_SEL | Selects the type of the algorithm 0 : The encryption algorithm (AES) 1 : A hash algorithm. The exact algorithm is defined by the fields CRYPTO_ALG and CRYPTO_ALG_MD. | 0x0 |
| 8 | R/W | CRYPTO_IRQ_EN | Interrupt Request Enable 0 : The interrupt generation ability is disabled. 1 : The interrupt generation ability is enabled. Generates an interrupt request at the end of operation. | 0x0 |
| 7 | R/W | CRYPTO_ENCDEC | Encryption/Decryption 0 : Decryption 1 : Encryption | 0x0 |
| 6:5 | R/W | CRYPTO_AES_KEY_SZ | The size of AES Key 0x0 : 128 bits AES Key 0x1 : 192 bits AES Key 0x2 : 256 bits AES Key 0x3 : 256 bits AES Key | 0x0 |
| 4 | R/W | CRYPTO_OUT_MD | Output Mode. This field makes sense only when the AES algorithm is selected (CRYPTO_HASH_SEL =0) 0 : Write back to memory all the resulting data 1 : Write back to memory only the final block of the resulting data | 0x0 |
| 3:2 | R/W | CRYPTO_ALG_MD | It defines the mode of operation of the AES algorithm when the controller is configured for an encryption/decryption processing (CRYPTO_HASH_SEL = 0). 0x0 : ECB | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------|--|-------|
| | | | <p>0x1 : ECB 0x2 : CTR 0x3 : CBC</p> <p>When the controller is configured to applies a HASH function, this field selects the desired HASH algorithm with the help of the CRYPTO_ALG.</p> <p>0x0 : HASH algorithms that are based on 32 bits operations 0x1 : HASH algorithms that are based on 64 bits operations 0x2 : Reserved 0x3 : Reserved</p> <p>See also the CRYPTO_ALG field.</p> | |
| 1:0 | R/W | CRYPTO_ALG | <p>Algorithm selection. When CRYPTO_HASH_SEL = 0 the only available choice is the AES algorithm.</p> <p>0x0 : AES 0x1 : Reserved 0x2 : Reserved 0x3 : Reserved</p> <p>When CRYPTO_HASH_SEL = 1, this field selects the desired hash algorithm, with the help of the CRYPTO_ALG_MD field.</p> <p>If CRYPTO_ALG_MD = 0x0 0x0 : MD5 0x1 : SHA-1 0x2 : SHA-256/224 0x3 : SHA-256</p> <p>If CRYPTO_ALG_MD = 0x1 0x0 : SHA-384 0x1 : SHA-512 0x2 : SHA-512/224 0x3 : SHA-512/256</p> | 0x0 |

Table 1016: CRYPTO_START_REG (0x30040004)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|--------------|---|-------|
| 0 | R0/W | CRYPTO_START | Write 1 to initiate the processing of the input data. This register is auto-cleared. | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 1017: CRYPTO_FETCH_ADDR_REG (0x30040008)

| Bit | Mode | Symbol | Description | Reset |
|------|------|-----------------------|---|-------|
| 31:0 | R/W | CRYPTO_FETCH_A DDR | The memory address from where will be retrieved the data that will be processed. The value of this register is updated as the calculation proceeds and the output data are written to the memory. | 0x0 |

Table 1018: CRYPTO_LEN_REG (0x3004000C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|------------|--|-------|
| 23:0 | R/W | CRYPTO_LEN | It contains the number of bytes of input data. If this number is not a multiple of a block size, the data is automatically extended with zeros. The value of this register is updated as the calculation proceeds and the output data are written to the memory. | 0x0 |

Table 1019: CRYPTO_DEST_ADDR_REG (0x30040010)

| Bit | Mode | Symbol | Description | Reset |
|------|------|---------------------|---|-------|
| 31:0 | R/W | CRYPTO_DEST_A DR | Destination address at where the result of the processing is stored. The value of this register is updated as the calculation proceeds and the output data are written to the memory. | 0x0 |

Table 1020: CRYPTO_STATUS_REG (0x30040014)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|------------------------|---|-------|
| 2 | R | CRYPTO_IRQ_ST | The status of the interrupt request line of the CRYPTO block. 0 : There is no active interrupt request. 1 : An interrupt request is pending. | 0x0 |
| 1 | R | CRYPTO_WAIT_FO R_IN | Indicates the situation where the engine waits for more input data. This is applicable when the CRYPTO_MORE_IN= 1, so the input data are fragmented in the memory. 0 : The crypto is not waiting for more input data. 1 : The crypto waits for more input data. The CRYPTO_INACTIVE flag remains to zero to indicate that the calculation is not finished. The supervisor of the CRYPTO must program to the CRYPTO_FETCH_ADDR and CRYPTO_LEN a new input data fragment. The calculation will be continued as soon as the CRYPTO_START register will be written with 1. This action will clear the CRYPTO_WAIT_FOR_IN flag. | 0x0 |
| 0 | R | CRYPTO_INACTIVE | 0 : The CRYPTO is active. The processing is in progress. 1 : The CRYPTO is inactive. The processing has finished. | 0x1 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 1021: CRYPTO_CLRIRQ_REG (0x30040018)

| Bit | Mode | Symbol | Description | Reset |
|-----|------|---------------|---|-------|
| 0 | R0/W | CRYPTO_CLRIRQ | Write 1 to clear a pending interrupt request. | 0x0 |

Table 1022: CRYPTO_MREG0_REG (0x3004001C)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|---|-------|
| 31:0 | R/W | CRYPTO_MREG0 | <p>It contains information that are depended by the mode of operation, when is used the AES algorithm:</p> <p>CBC - IV[31:0]</p> <p>CTR - CTRBLK[31:0]. It is the initial value of the 32 bits counter.</p> <p>At any other mode, the contents of this register has no meaning.</p> | 0x0 |

Table 1023: CRYPTO_MREG1_REG (0x30040020)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:0 | R/W | CRYPTO_MREG1 | <p>It contains information that are depended by the mode of operation, when is used the AES algorithm:</p> <p>CBC - IV[63:32]</p> <p>CTR - CTRBLK[63:32]</p> <p>At any other mode, the contents of this register has no meaning.</p> | 0x0 |

Table 1024: CRYPTO_MREG2_REG (0x30040024)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:0 | R/W | CRYPTO_MREG2 | <p>It contains information that are depended by the mode of operation, when is used the AES algorithm:</p> <p>CBC - IV[95:64]</p> <p>CTR - CTRBLK[95:64]</p> <p>At any other mode, the contents of this register has no meaning.</p> | 0x0 |

Table 1025: CRYPTO_MREG3_REG (0x30040028)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:0 | R/W | CRYPTO_MREG3 | <p>It contains information that are depended by the mode of operation, when is used the AES algorithm:</p> <p>CBC - IV[127:96]</p> <p>CTR - CTRBLK[127:96]</p> <p>At any other mode, the contents of this register has no meaning.</p> | 0x0 |

Multi-core Bluetooth 5.2 SoC family with system PMU

Table 1026: CRYPTO_KEYS_START (0x30040100)

| Bit | Mode | Symbol | Description | Reset |
|------|------|--------------|--|-------|
| 31:0 | W | CRYPTO_KEY_X | CRYPTO_KEY_(0-63) This is the AES keys storage memory. This memory is accessible via AHB slave interface, only when the CRYPTO is inactive (CRYPTO_INACTIVE = 1). | N/A |

43 Ordering Information

Table 1027: Ordering Information (Samples)

| Part Number | Package | Size (mm) | Shipment Form | Pack Quantity |
|------------------|----------|-------------|---------------|---------------|
| DA14691-00000HQ2 | VFBGA86 | 6 x 6 x 0.9 | Reel | 100/1000 |
| DA14695-00000HQ2 | VFBGA86 | 6 x 6 x 0.9 | Reel | 100/1000 |
| DA14697-00000HR2 | VFBGA100 | 5 x 5 x 0.9 | Reel | 100/1000 |
| DA14699-00000HR2 | VFBGA100 | 5 x 5 x 0.9 | Reel | 100/1000 |

Table 1028: Ordering Information (Production)

| Part Number | Package | Size (mm) | Shipment Form | Pack Quantity |
|------------------|----------|-------------|---------------|---------------|
| DA14691-00000HQ2 | VFBGA86 | 6 x 6 x 0.9 | Reel | 4000 |
| DA14695-00000HQ2 | VFBGA86 | 6 x 6 x 0.9 | Reel | 4000 |
| DA14697-00000HR2 | VFBGA100 | 5 x 5 x 0.9 | Reel | 5000 |
| DA14699-00000HR2 | VFBGA100 | 5 x 5 x 0.9 | Reel | 5000 |

Part Number Legend:

DA1469x-RRXXXYYZ

RR: chip revision number

XXX: variant (00T: High Temperature)

YY: package code (HQ: VFBGA86, HR: VFBGA100)

Z: packing method (1: Tray, 2: Reel, A: Mini-Reel)

44 Package information

44.1 Moisture Sensitivity Level (MSL)

The MSL is an indicator for the maximum allowable time period (floor life time) 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 % RH before the solder reflow process.

BGA packages are qualified for MSL 3.

| MSL Level | Floor Life Time |
|-----------|-------------------------|
| MSL 4 | 72 hours |
| MSL 3 | 168 hours |
| MSL 2A | 4 weeks |
| MSL 2 | 1 year |
| MSL 1 | Unlimited at 30°C/85%RH |

44.2 Soldering information

Refer to the JEDEC standard J-STD-020 for relevant soldering information. This document can be downloaded from <http://www.jedec.org>.

44.3 Package outlines

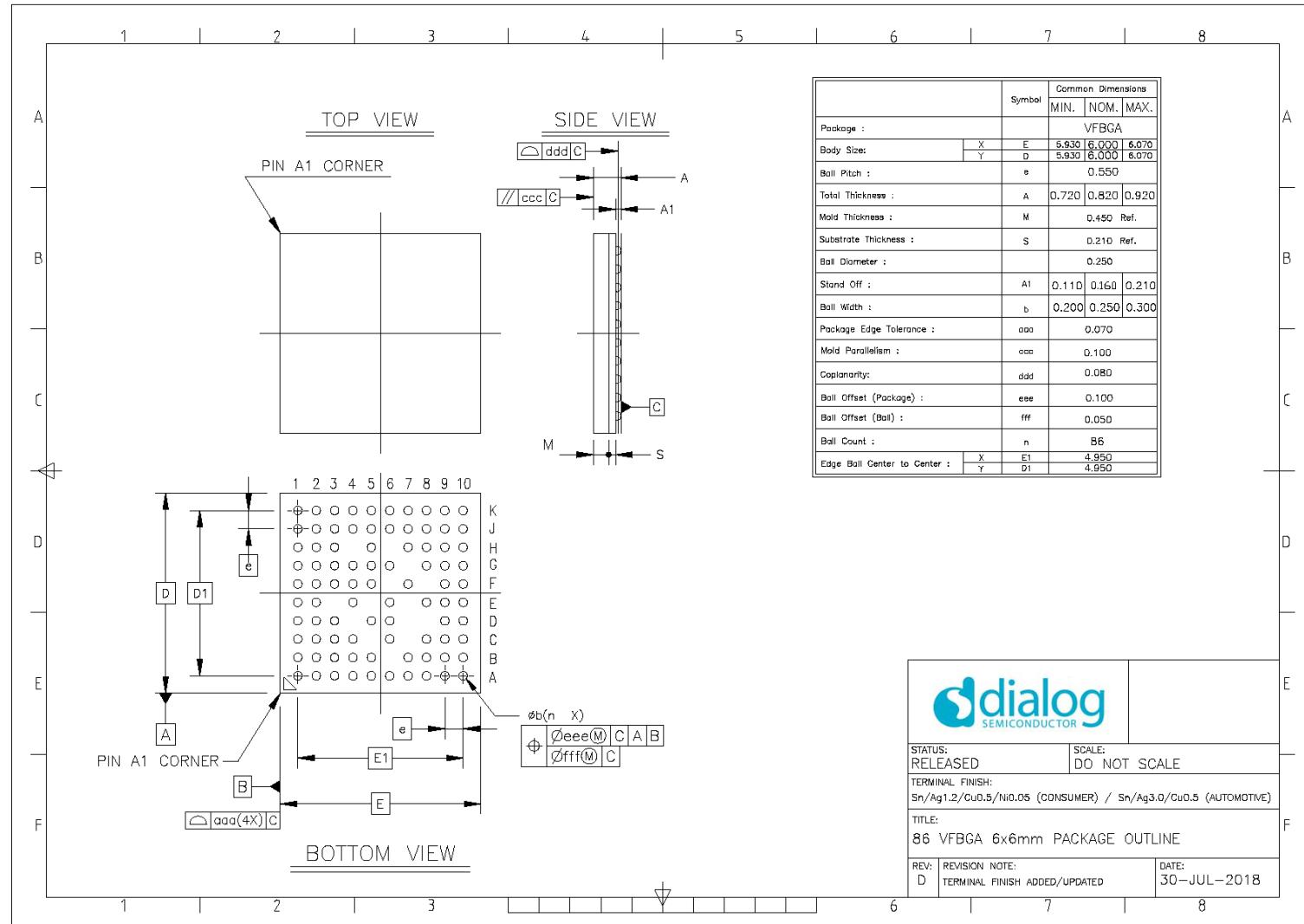


Figure 110: VFBGA86 Package Outline Drawing

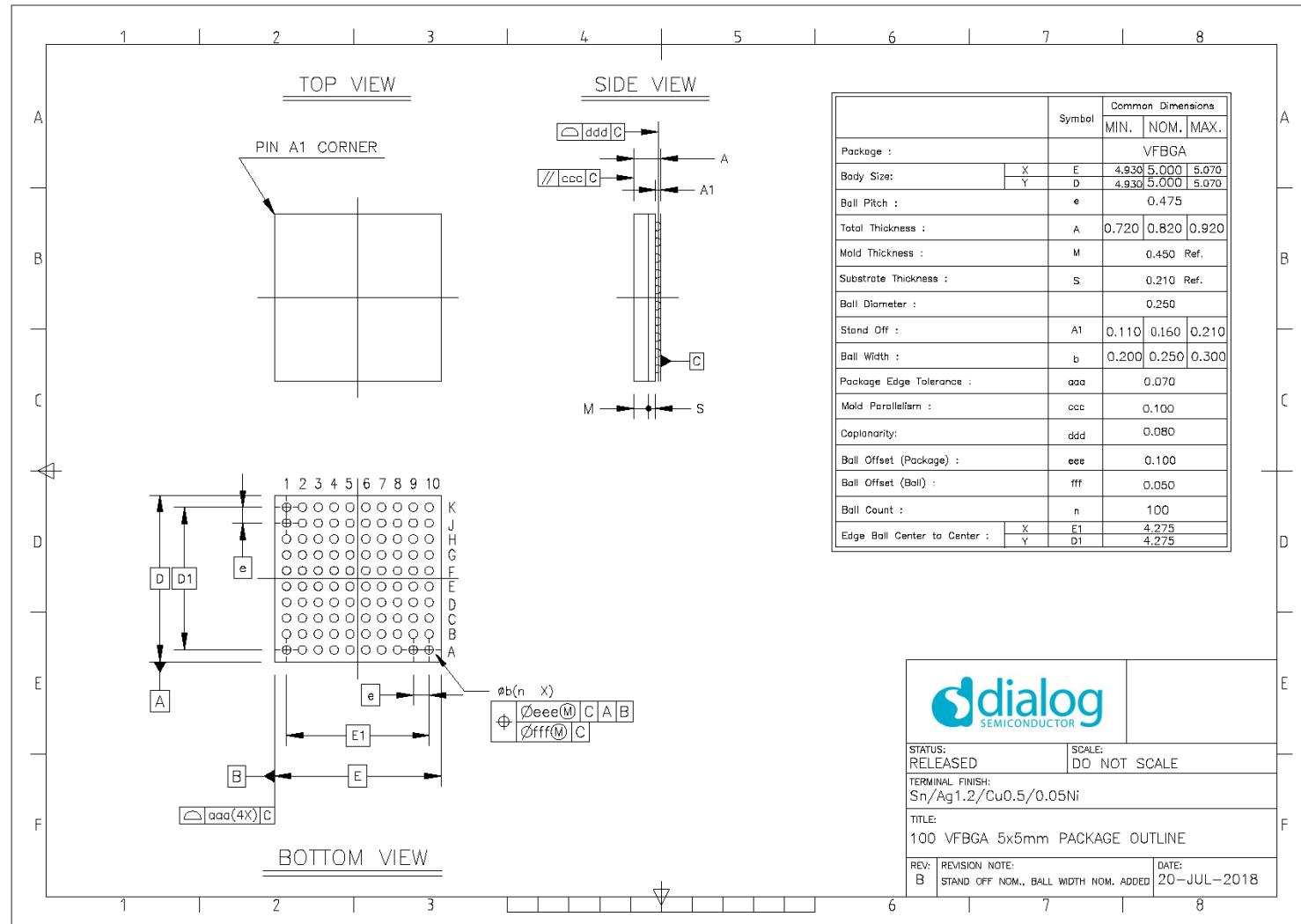


Figure 111: VFBGA100 Package Outline Drawing

Revision History

| Revision | Date | Description |
|---|-------------|--|
| 3.2 | 12-Oct-2020 | <ul style="list-style-type: none"> Editorial corrections Corrected table 138 Corrected chapter 7.2.7 (primary charger detection) values Added rail discharging section in Charger chapter Updated GPIOs being CMAC SWD pins by default in IO chapter Added GPADC LDO dependency to V18P Added DCDC V12/V14 rail characteristics Added typical value for pin leakage in characteristics |
| 3.1 | 18-Oct-2019 | <ul style="list-style-type: none"> Editorial corrections sections 40.2.5 Added new legal text to section 38.4 Section 10.2, Table 88, following regs removed (RTC_EVENT_CTRL_REG, RTC_MOTOR_EVENT_PERIOD_REG, RTC_PDC_EVENT_PERIOD_REG) Table 699 P0_PADPWR_CTRL_REG (0x50020AF4), added text “1.8 V only” to description Corrected Figure 57: Wake-Up Controller Block Diagram Sections 6.3.3 Wake-up Options, Ultra-Fast (The VDD_SLEEP_LEVEL voltage must be 0.9 V.) Section 15.2.1 removed brackets from (Operand2) |
| 3.0 | 26-Jul-2019 | Product status: Final. Datasheet status: Final. |
| Change details: Specifications updated, JIRA items added, text and graphics edited, disclaimer updated. | | |
| 2.0 | 14-Oct-2020 | Product status: Preliminary. Datasheet status: Qualification. |
| | | |

Multi-core Bluetooth 5.2 SoC family with system PMU

Status Definitions

| Revision | Datasheet status | Product status | Definition |
|----------|------------------|----------------|--|
| 1.<n> | Target | Development | This datasheet contains the design specifications for product development. Specifications may be changed in any manner without notice. |
| 2.<n> | Preliminary | Qualification | This datasheet contains the specifications and preliminary characterization data for products in pre-production. Specifications may be changed at any time without notice in order to improve the design. |
| 3.<n> | Final | Production | This datasheet contains the final specifications for products in volume production. The specifications may be changed at any time in order to improve the design, manufacturing and supply. Major specification changes are communicated via Customer Product Notifications. Datasheet changes are communicated via www.dialog-semiconductor.com . |
| 4.<n> | Obsolete | Archived | This datasheet contains the specifications for discontinued products. The information is provided for reference only. |

Disclaimer

Unless otherwise agreed in writing, the Dialog Semiconductor products (and any associated software) referred to in this document are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of a Dialog Semiconductor product (or associated software) can reasonably be expected to result in personal injury, death or severe property or environmental damage. Dialog Semiconductor and its suppliers accept no liability for inclusion and/or use of Dialog Semiconductor products (and any associated software) in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Information in this document is believed to be accurate and reliable. However, Dialog Semiconductor does not give any representations or warranties, express or implied, as to the accuracy or completeness of such information. Dialog Semiconductor furthermore takes no responsibility whatsoever for the content in this document if provided by any information source outside of Dialog Semiconductor.

Dialog Semiconductor reserves the right to change without notice the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications. Notwithstanding the foregoing, for any automotive grade version of the device, Dialog Semiconductor reserves the right to change the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications, in accordance with its standard automotive change notification process.

Applications, software, and semiconductor products described in this document are for illustrative purposes only. Dialog Semiconductor makes no representation or warranty that such applications, software and semiconductor products will be suitable for the specified use without further testing or modification. Unless otherwise agreed in writing, such testing or modification is the sole responsibility of the customer and Dialog Semiconductor excludes all liability in this respect.

Nothing in this document may be construed as a license for customer to use the Dialog Semiconductor products, software and applications referred to in this document. Such license must be separately sought by customer with Dialog Semiconductor.

All use of Dialog Semiconductor products, software and applications referred to in this document is subject to Dialog Semiconductor's [Standard Terms and Conditions of Sale](#), available on the company website (www.dialog-semiconductor.com) unless otherwise stated.

Dialog, Dialog Semiconductor and the Dialog logo are trademarks of Dialog Semiconductor Plc or its subsidiaries. All other product or service names and marks are the property of their respective owners.

© 2020 Dialog Semiconductor. All rights reserved.

RoHS Compliance

Dialog Semiconductor's 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.

Contacting Dialog Semiconductor

| United Kingdom (Headquarters) | North America | Singapore | China (Shenzhen) |
|--|--|--|---|
| Dialog Semiconductor (UK) LTD Phone: +44 1793 757700 | Dialog Semiconductor Inc. Phone: +1 408 845 8500 | Dialog Semiconductor Singapore Phone: +65 64 8499 29 | Dialog Semiconductor China Phone: +86 755 2981 3669 |
| Germany Dialog Semiconductor GmbH Phone: +49 7021 805-0 | Japan Dialog Semiconductor K. K. Phone: +81 3 5425 4567 | Hong Kong Dialog Semiconductor Hong Kong Phone: +852 3769 5200 | China (Shanghai) Dialog Semiconductor China Phone: +86 21 5424 9058 |
| The Netherlands Dialog Semiconductor B.V. Phone: +31 73 640 8822 | Taiwan Dialog Semiconductor Taiwan Phone: +886 281 786 222 | Korea Dialog Semiconductor Korea Phone: +82 2 3469 8200 | |
| Email: enquiry@diasemi.com | Web site: www.dialog-semiconductor.com | | |