NOT FOR PUBLIC RELEASE

# RTL8762AR-CG
# RTL8762AG-CG
# RTL8762AJ-CG
# RTL8762AK-CG

## BLUETOOTH LOW ENERGY SOC

## DATASHEET

**(CONFIDENTIAL: Development Partners Only)**

**Rev. 1.3**
**26 May 2017**
**Track ID: JATR-8275-15**

**Realtek Semiconductor Corp.**
**No. 2, Innovation Road II, Hsinchu Science Park, Hsinchu 300, Taiwan**
**Tel.: +886-3-578-0211. Fax: +886-3-577-6047**
**www.realtek.com**

## COPYRIGHT



## DISCLAIMER

Realtek provides this document 'as is', without warranty of any kind. Realtek may make improvements and/or changes in this document or in the product described in this document at any time. This document could include technical inaccuracies or typographical errors.

## TRADEMARKS

Realtek is a trademark of Realtek Semiconductor Corporation. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

## USING THIS DOCUMENT

This document is intended for the software engineer's reference and provides detailed programming information.

Though every effort has been made to ensure that this document is current and accurate, more information may have become available subsequent to the production of this guide.

## REVISION HISTORY

| Revision | Release Date | Summary |
|---|---|---|
| 1.0 | 2016/07/15 | First release. |
| 1.1 | 2016/07/25 | Added RTL8762AJ data. |
| 1.2 | 2016/09/09 | Added section 8 Clock Management, page 16.<br>Added section 9 Power Management Unit (PMU), page 19.<br>Corrected minor typing errors. |
| 1.3 | 2017/05/26 | Added section 11.5 RTX LDO Characteristics, page 107.<br>Added section 11.6 Synthesizer LDO Characteristics, page 107.<br>Added section 11.7 Retention LDO Characteristics, page 108.<br>Corrected minor typing errors. |

# Table of Contents

# List of Tables

# List of Figures

# 1. General Description

## *1.1. Overview*

The RTL8762AG/RTL8762AR/RTL8762AJ/RTL8762AK (hereafter referred to as the RTL8762A) are ultra-low-power system on-chip solutions for Bluetooth low energy applications that combines the excellent performance of a leading RF transceiver with a low-power ARM® Cortex™-M0, 256KB eFlash, 80KB RAM, and rich powerful supporting features and peripherals.

The embedded ARM® Cortex™-M0 32-bit CPU features a 16-bit instruction set with 32-bit extensions (Thumb-2® technology) that delivers high-density code with a small memory footprint. By using a single-cycle 32-bit multiplier, a 3-stage pipeline, and a Nested Vector Interrupt Controller (NVIC), the ARM® Cortex™-M0 CPU makes program execution simple and highly efficient.

In the RTL8762A, we support the Serial Wire Debug (SWD) interface provided as part of the Debug Access Port (DAP), in conjunction with the Basic Branch Buffer (BBB). This offers a flexible and powerful mechanism for non-intrusive program code debugging. Users can easily add breakpoints in the code and perform single-step debugging.

The RTL8762A memory architecture (see Figure 1, page 2) includes 80KB of data RAM that can be divided into three portions, e.g., 16KB for buffer RAM, 24KB for on-RAM, and 40KB for off-RAM. Specifically, the available memory for APPs includes 9KB in on-RAM and 18KB in off-RAM. Each on-RAM and off-RAM can be divided into static and dynamic areas. The area size can be adjusted via eFUSE.
The on-RAM can maintain data when the system is in Deep Low Power State (DLPS) mode (Data in off-RAM is lost when the system enters DLPS mode). The static area refers to the global variables in the program. The dynamic area refers to the direct or indirect call of the OS's pvPortMalloc allocated space, such as xQueueCreate, xTaskCreate, etc.

The RTL8762A also integrates a sigma-delta ADC, programmable gain amplifier, and microphone bias circuit for voice command application. The RTL8762A embeds IR transceiver, hardware keyscan, and Quad-decoder on a single IC, and is provided in a QFN package.

Package and pin differences between the AG/AR/AJ/AK versions are described in this document where they arise.

## 1.2. RTL8762A Memory Architecture

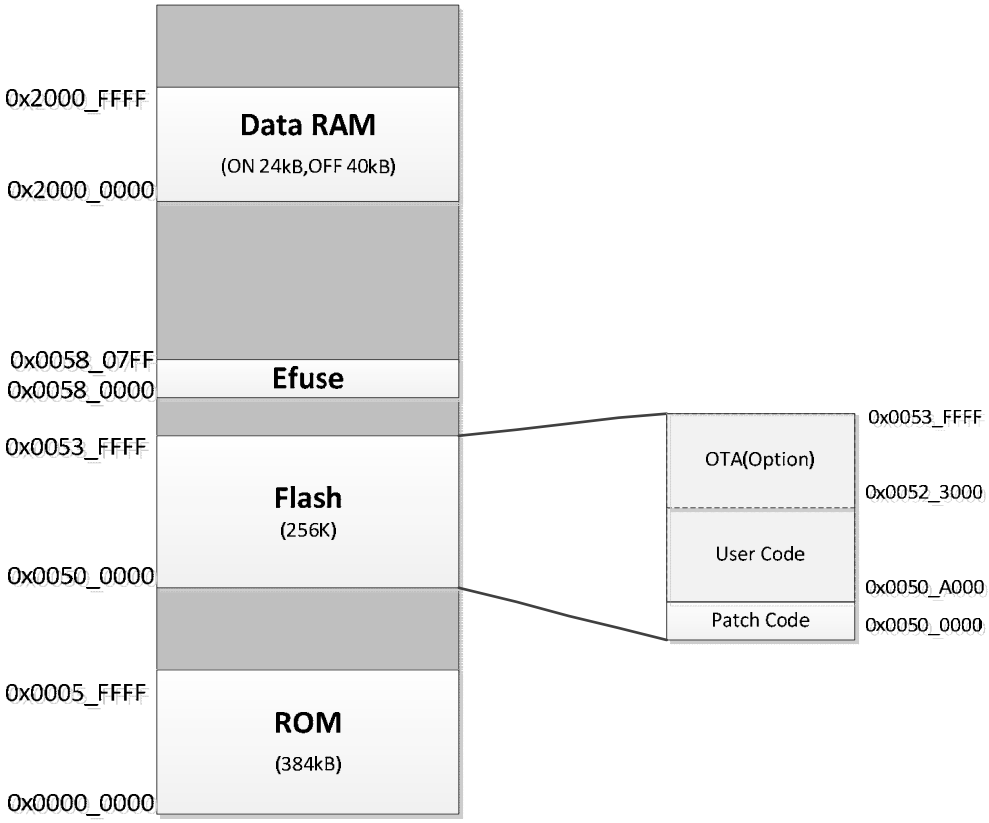


**Figure 1. RTL8762A Memory Architecture**

# 2. Features

### General

- Ultra low power consumption with intelligent PMU
- Supports Bluetooth 4.2 core specification
- Integrated MCU to execute Bluetooth protocol stack
- Supports multiple level Low Energy states
- Supports LE L2CAP Connection Oriented Channel Support
- Supports LE low duty directed advertising
- Supports LE data length extension feature
- Supports OTA (Over-the-Air) programming mechanism for firmware upgrade
- Supports external 32KHz XTAL for low power mode
- Supports GAP, ATT/GATT, SMP, L2CAP
- Generic Applications for GAP Central, Peripheral, Observer and Broadcaster Roles

### Platform

- ARM Cortex-M0 (Maximum 53MHz)
- 256KB embedded flash
- 80KB RAM
- 2KB eFUSE
- Supports AES128/192/256 encrypt/decrypt engine

### Bluetooth Transceiver

- Fast AGC control to improve receiving dynamic range
- Supports Bluetooth Low Energy PHY

### Peripheral Interfaces

- Flexible General Purpose IOs
  - RTL8762AR: 15 GPIOs
  - RTL8762AG: 16 GPIOs
  - RTL8762AJ: 23 GPIOs
  - RTL8762AK: 37 GPIOs
- Three configurable LED pins
- Hardware Keyscan and Quad-decoder
- Embedded IR transceiver
- Real-Time Counters (RTC)
- Supports generic 4-wire SPI master/slave
- Supports Low power comparator
- Timers x 8
- I2C x 2
- PWM x 4
- UART x 3
- Supports 40MHz XTAL
- Supports audio ADC for voice command application

### Package

- RTL8762AR: 32-pin 5x5mm QFN
- RTL8762AG: 32-pin 5x5mm QFN
- RTL8762AJ: 40-pin 5x5mm QFN
- RTL8762AK: 56-pin 7x7mm QFN

# 3. Applications

- TV Remote Controller
- LE HID
- Beacon
- Home Automation
- Key Fob
- Wristband
- Wearable Device

# 4. Block Diagrams

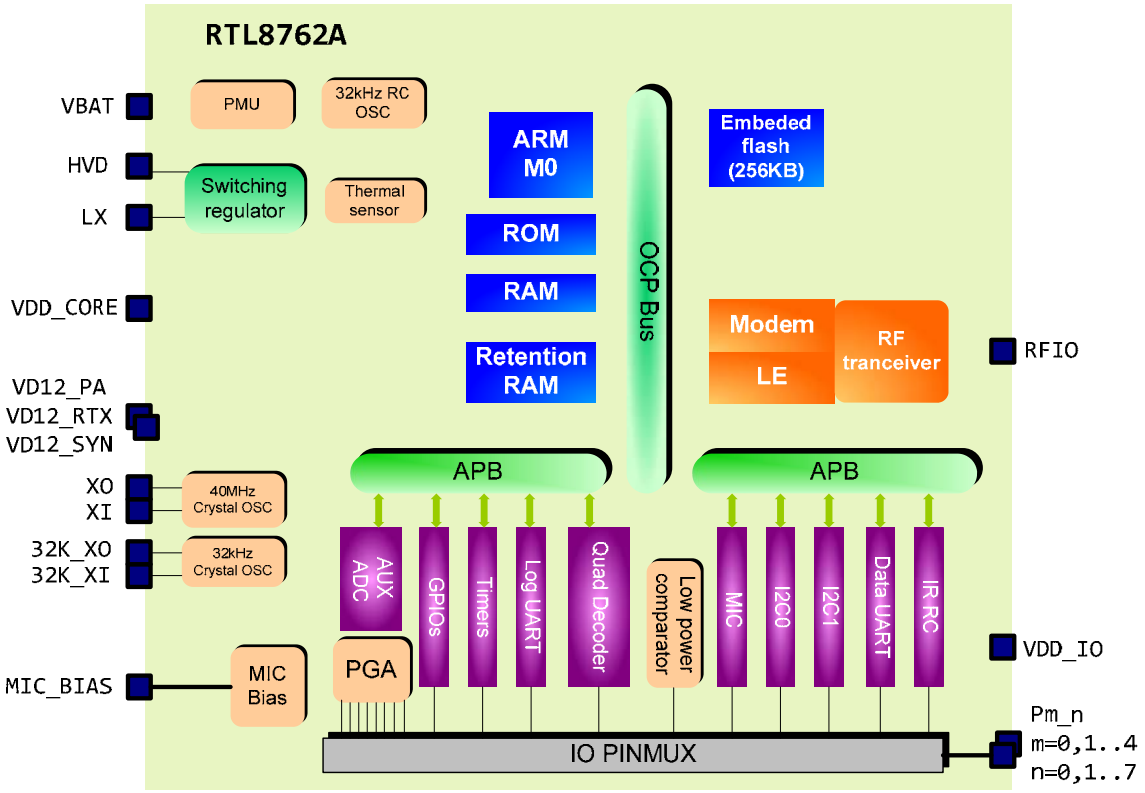


**Figure 2. Block Diagram**

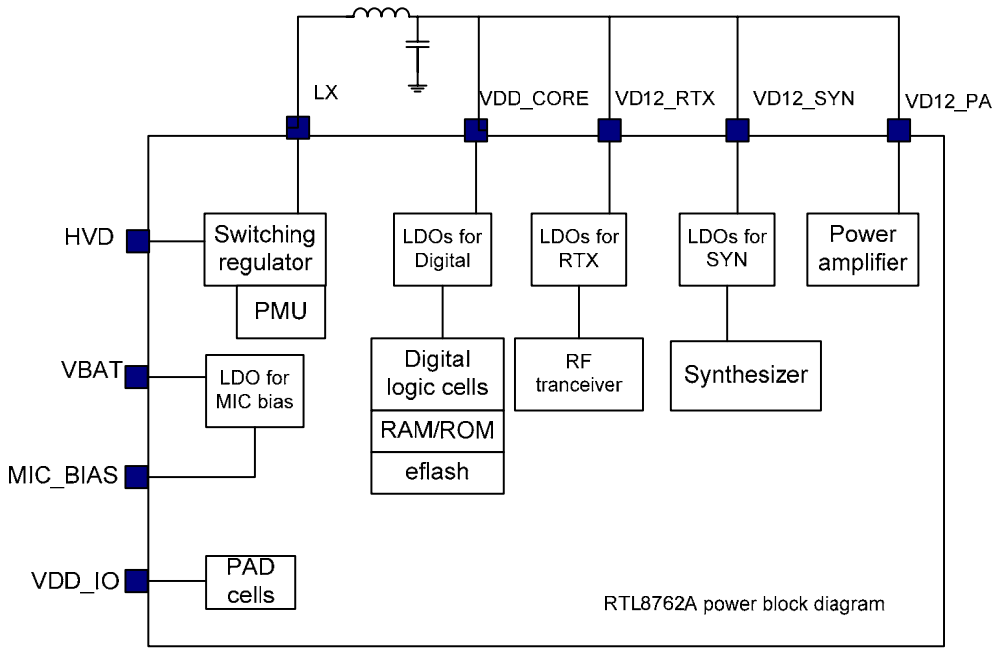


**Figure 3. Power Block Diagram**

# 5. Pin Assignments

## 5.1. RTL8762AR Pin Assignments

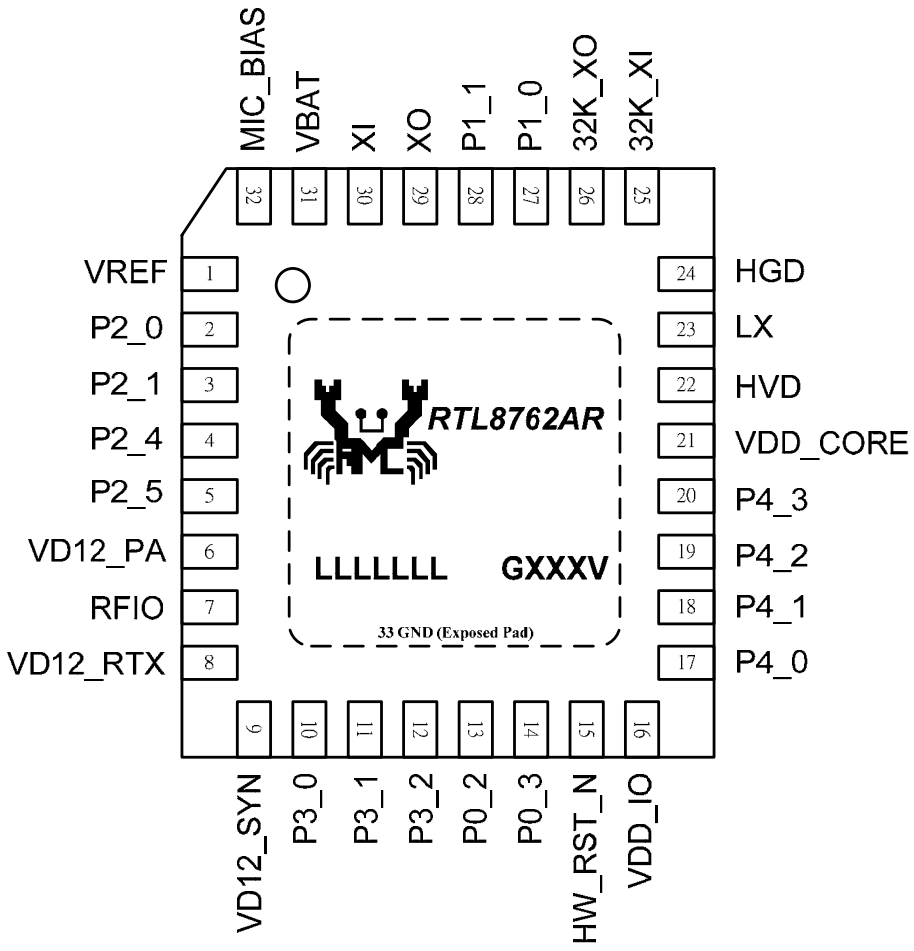


**Figure 4. RTL8762AR Pin Assignments**

## 5.2. Package Identification

Green package is indicated by the 'G' in GXXXV (Figure 4).

## 5.3. RTL8762AG Pin Assignments

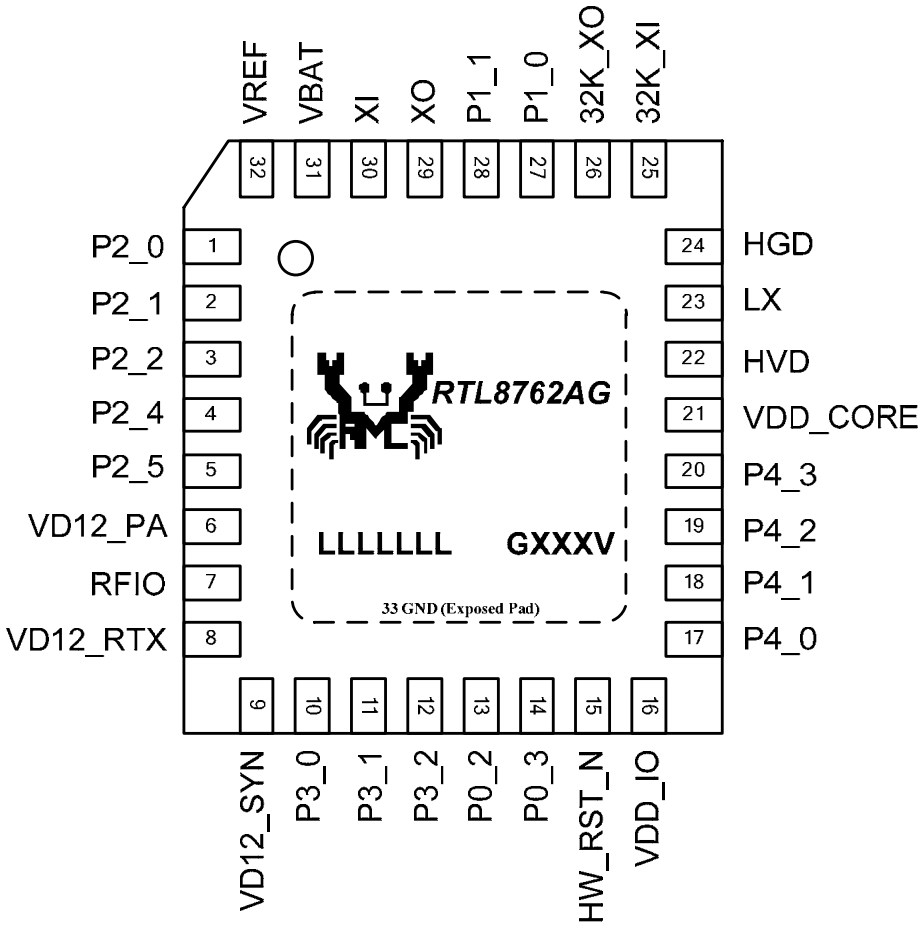


**Figure 5. RTL8762AG Pin Assignments**

## 5.4. Package Identification

Green package is indicated by the 'G' in GXXXV (Figure 5).

## 5.5. RTL8762AJ Pin Assignments

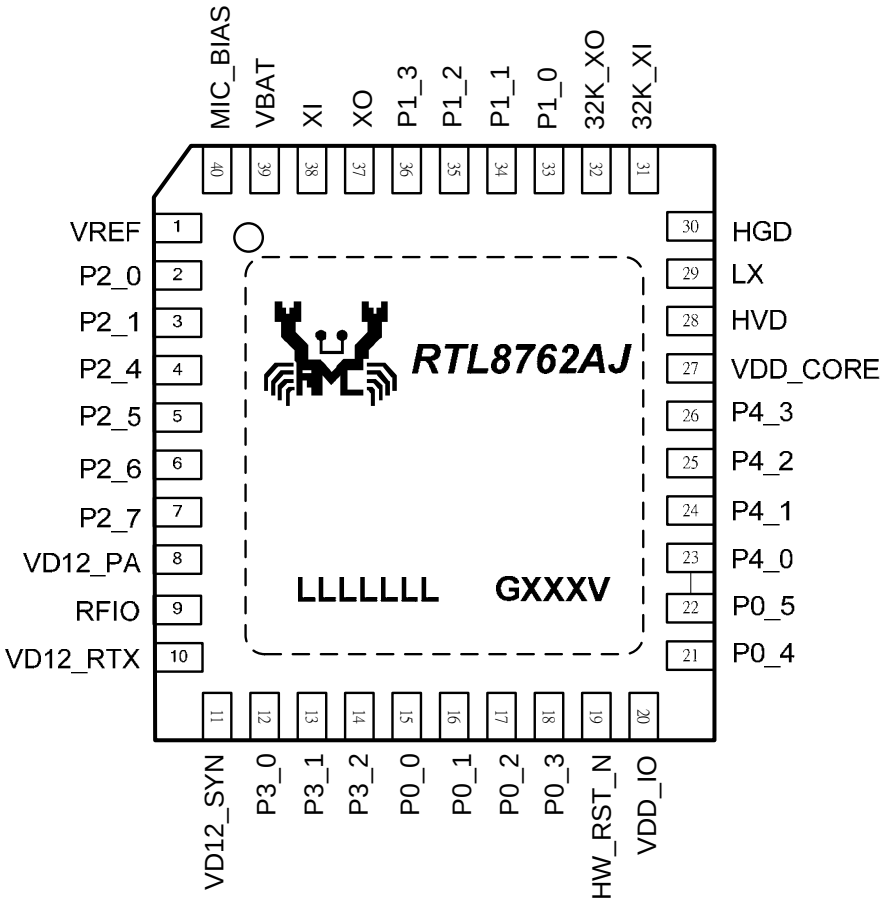


**Figure 6. RTL8762AJ Pin Assignments**

## 5.6. Package Identification

Green package is indicated by the 'G' in GXXXV (Figure 6).

## 5.7. RTL8762AK Pin Assignments

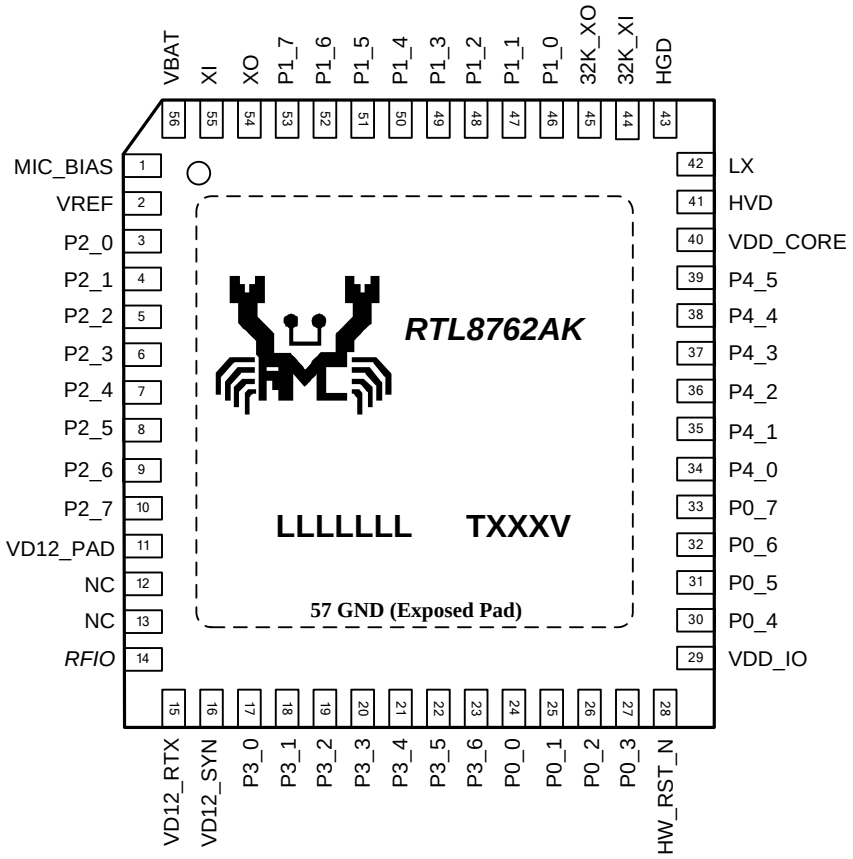


**Figure 7. RTL8762AK Pin Assignments**

## 5.8. Package Identification

Green package is indicated by the 'G' in GXXXV (Figure 7).

# 6. Pin Descriptions

The following signal type codes are used in the tables:

I: Input O: Output

P: Power

## 6.1. RF Interface

**Table 1. RF Interface**

| Symbol | Type | Pin No | | | | Description |
|---|---|---|---|---|---|---|
| - | - | AR | AG | AJ | AK | - |
| RFIO | I | 7 | 7 | 9 | 14 | BT RX signal/BT TX signal (low power mode) |

## 6.2. XTAL and System Interface

**Table 2. XTAL and System Interface**

| Symbol | Type | Pin No | | | | Description |
|---|---|---|---|---|---|---|
| - | - | AR | AG | AJ | AK | - |
| 32K_XI | I | 25 | 25 | 31 | 44 | 32k crystal input or external 32k clock input |
| 32K_XO | O | 26 | 26 | 32 | 45 | 32k crystal output |
| XI | I | 30 | 30 | 38 | 55 | 40MHz crystal input or external 40MHz clock input |
| XO | O | 29 | 29 | 37 | 54 | 40MHz crystal output |
| HW_RST_N | I | 15 | 15 | 19 | 28 | Hardware reset pin; low active |

## 6.3. General Purpose IOs

**Table 3. General Purpose IOs**

| Symbol | Type | Pin No | | | | Description |
|---|---|---|---|---|---|---|
| - | - | AR | AG | AJ | AK | - |
| P0_0 | IO | - | - | 15 | 24 | General purpose IO (GPIO); refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down. |
| P0_1 | IO | - | - | 16 | 25 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down. |

| Symbol | Type | Pin No | | | | Description |
|---|---|---|---|---|---|---|
| - | - | AR | AG | AJ | AK | - |
| P0_2 | IO | 13 | 13 | 17 | 26 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>20mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down. |
| P0_3 | IO | 14 | 14 | 18 | 27 | LOG_UART TX.<br>Erase eFlash trigger.<br>Power on trap; pull-up to erase eFlash content when powered on. (default PAD internal pull-down).<br>Note: We recommend this pin is connected via a resistor to ground. |
| P0_4 | IO | - | - | 21 | 30 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>20mA driving capability.<br>With internal pull-up and pull-down. |
| P0_5 | IO | - | - | 22 | 31 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>20mA driving capability.<br>With internal pull-up and pull-down. |
| P0_6 | IO | - | - | - | 32 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P0_7 | IO | - | - | - | 33 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P1_0 | IO | 27 | 27 | 33 | 46 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down.<br>SWDIO (default). |
| P1_1 | IO | 28 | 28 | 34 | 47 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down.<br>SWDCLK (default). |
| P1_2 | IO | - | - | 35 | 48 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P1_3 | IO | - | - | 36 | 49 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P1_4 | IO | - | - | - | 50 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P1_5 | IO | - | - | - | 51 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P1_6 | IO | - | - | - | 52 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |

| Symbol | Type | Pin No | | | | Description |
|---|---|---|---|---|---|---|
| - | - | AR | AG | AJ | AK | - |
| P1_7 | IO | - | - | - | 53 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P2_0 | IO | 2 | 1 | 2 | 3 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>AUXADC input 0. |
| P2_1 | IO | 3 | 2 | 3 | 4 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>AUXADC input 1. |
| P2_2 | IO | - | 3 | - | 5 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down.<br>AUXADC input 2. |
| P2_3 | IO | - | - | - | 6 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down.<br>AUXADC input 3. |
| P2_4 | IO | 4 | 4 | 4 | 7 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>AUXADC input 4. |
| P2_5 | IO | 5 | 5 | 5 | 8 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>AUXADC input 5. |
| P2_6 | IO | - | - | 6 | 9 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down.<br>AUXADC input 6. |
| P2_7 | IO | - | - | 7 | 10 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down.<br>AUXADC input 7. |
| P3_0 | IO | 10 | 10 | 12 | 17 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>HCI_UART_TX (default). |

| Symbol | Type | Pin No | | | | Description |
|---|---|---|---|---|---|---|
| - | - | AR | AG | AJ | AK | - |
| P3_1 | IO | 11 | 11 | 13 | 18 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>HCI_UART_RX (default). |
| P3_2 | IO | 12 | 12 | 14 | 19 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>8mA driving capability. |
| P3_3 | IO | - | - | - | 20 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P3_4 | IO | - | - | - | 21 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P3_5 | IO | - | - | - | 22 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P3_6 | IO | - | - | - | 23 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |
| P4_0 | IO | 17 | 17 | 23 | 34 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down. |
| P4_1 | IO | 18 | 18 | 24 | 35 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down. |
| P4_2 | IO | 19 | 19 | 25 | 36 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down. |
| P4_3 | IO | 20 | 20 | 26 | 37 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With wakeup function.<br>With internal pull-up and pull-down.<br>Low power comparator input. |
| P4_4 | IO | - | - | - | 38 | General purpose IO; refer to Table 8, Pin Multiplexer, page 20.<br>8mA driving capability.<br>With internal pull-up and pull-down. |

## 6.4. Power Pins

Table 4. Power Pins

| Symbol | Type | Pin No | | | | Description |
|---|---|---|---|---|---|---|
| - | - | AR | AG | AJ | AK | - |
| VREF | P | 1 | 32 | 1 | 2 | ADC reference voltage (decouple) |
| VD12_PA | P | 6 | 6 | 8 | 11 | Supply 1.2V power for PA |
| VD12_RTX | P | 8 | 8 | 10 | 15 | Supply 1.2V power for RF transceiver |
| VD12_SYN | P | 9 | 9 | 11 | 16 | Supply 1.2V power for synthesizer |
| VDD_IO | P | 16 | 16 | 20 | 29 | Supply 1.8V~3.3V power for digital IO PADs |
| VDD_CORE | P | 21 | 21 | 27 | 40 | Supply 1.2V power for digital core |
| HVD | P | 22 | 22 | 28 | 41 | Supply 1.8V~3.3V power for Switching regulator input |
| LX | P | 23 | 23 | 29 | 42 | Switching regulator output |
| HGD | P | 24 | 24 | 30 | 43 | Ground for switching regulator |
| VBAT | P | 31 | 31 | 39 | 56 | Battery voltage input |
| MIC_BIAS | P | 32 | - | 40 | 1 | Microphone bias |

# 7. Bluetooth Radio

## 7.1. RF Transceiver

The RTL8762A includes an embedded GFSK RF transceiver with ultra-low power consumption and full compliance with the Bluetooth low energy wireless system. The block diagram is shown in Figure 8.

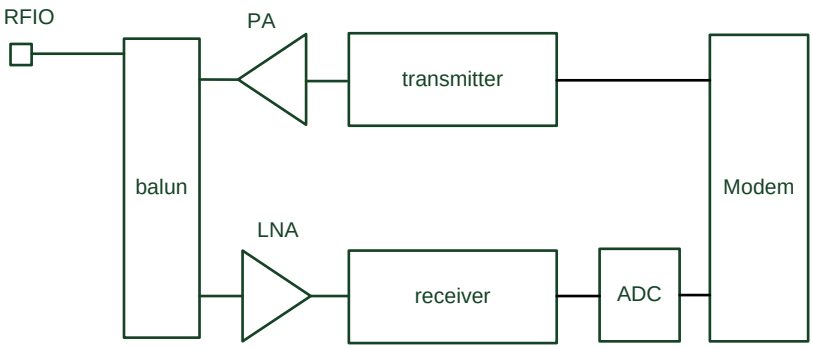


**Figure 8. RF Transceiver Block Diagram**

## 7.2. Modem

In the transmit path, the modem combines with the RF transmitter to generate a GFSK signal. In the receiver path, the modem receives a baseband GFSK signal from an analog to digital converter (ADC), and decodes the bit data via channel filtering, synchronizing, and demodulating.

An RF automatic calibration scheme is implemented in the modem to compensate for transistor characteristic variations in the CMOS process, and for ambient temperature differences.

## 7.3. Transmitter

The transmitter convert baseband signals to 2.4GHz unlicensed Industrial, Scientific and Medical (ISM) band GFSK modulated signals. The up-converted GFSM signal is amplified by the integrated power amplifier.

## 7.4. Front-End

To minimize external BOM requirements, the RTL8762A is single-ended RF mode and TX/RX path sharing the same RFIO pin with an integrated balun. For antenna matching and harmonic signal reduction, a PI matching network is required in the RF path.

# 8. Clock Management

For optimal power consumption and performance, the RTL8762A offers high and low frequency clocks. The high frequency clock is generated by an external 40MHz crystal oscillator (XTAL) or internal oscillator. The low frequency clock is generated by a 32.768kHz XTAL.

In normal mode the high frequency clock is kept running to provide clock to the CPU, Bluetooth core, and the peripheral block. In low power mode the high frequency clock is turned off for power saving. The 32kHz low frequency oscillator/XTAL remains on to provide clock to the RTC (Real Time Counter), BT core, and PMU.

## *8.1. 40MHz XTAL Oscillator*

The PCB connection between the 40MHz XTAL and RTL8762A is shown in Figure 9. The 40MHz XTAL specification is shown in Table 5, page 17. The external capacitor is selected via the following equation:

$$CL = \frac{(C1 + Cxi)(C2 + Cxo)}{(C1 + Cxi + C2 + Cxo)} + Cstray$$

Where CL is the load capacitance of the 40MHz XTAL.

Cxi and Cxo are internal trimming capacitors. A typical value for Cxi/Cxo in the RTL8762A is 6.4pF. The tuning range of Cxi/Cxo is 0pF~12.8pF.

Cstray is parasitic capacitance from package and PCB routing effects. In a standard case the capacitance is from 2~5pF, but the actual value of Cstray is strongly dependent on PCB layout and the package size of the XTAL. For example, when CL=12pF crystal is selected, C1 is equal to C2 and Cxi is equal to Cxo, and Cstray is estimated as 4pF.

C1+Cx1=16pF; in a typical setting Cxi is 6.4pF, and C1, C2 is 16-6.4~=10pF. For precise frequency accuracy, fine-tuning Cxi/Cxo is recommended during the mass production procedure.

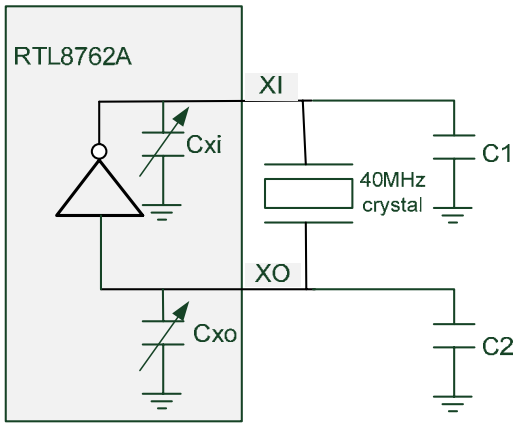


**Figure 9. 40MHz Crystal Oscillator Schematic**

Table 5. 40MHz XTAL Specification

| Parameter | Minimum | Typical | Maximum |
|---|---|---|---|
| Frequency (MHz) | - | 40 | - |
| Frequency tolerance (ppm) | - | - | ±10 |
| Frequency stability (ppm) | - | - | ±10 |
| Load capacitance (pF) | - | 12 | - |
| Drive Level (μW) | - | - | 300 |
| Equivalent Series Resistance (Ohm) | - | - | 25 |
| Insulation Resistance (MOhm) | 500 | - | - |

## *8.2. 32.768KHz XTAL Oscillator*

The RTL8762A uses a 32.768kHz XTAL oscillator as a sleep clock in low power mode. The PCB connection between the 32.768kHz XTAL and RTL8762A is shown in Figure 10. The 32.768kHz XTAL specification is shown in Table 6, page 18.

There is a fixed 7pF capacitor (Cx) and a trimming capacitor (Cxi/Cxo) with a value from 0pF to 12.8pF in the RTL8762A. Due to the embedded Cx, C1 and C2 are not required when a crystal load capacitor (CL) of 7pF is selected. The calculated value of Cxi, Cxo, C1, and C2 is shown in the following equation:

$$CL = \frac{(C1 + Cx + Cxi)(C2 + Cx + Cxo)}{(C1 + 2Cx + Cxi + C2 + Cxo)} + Cstray$$

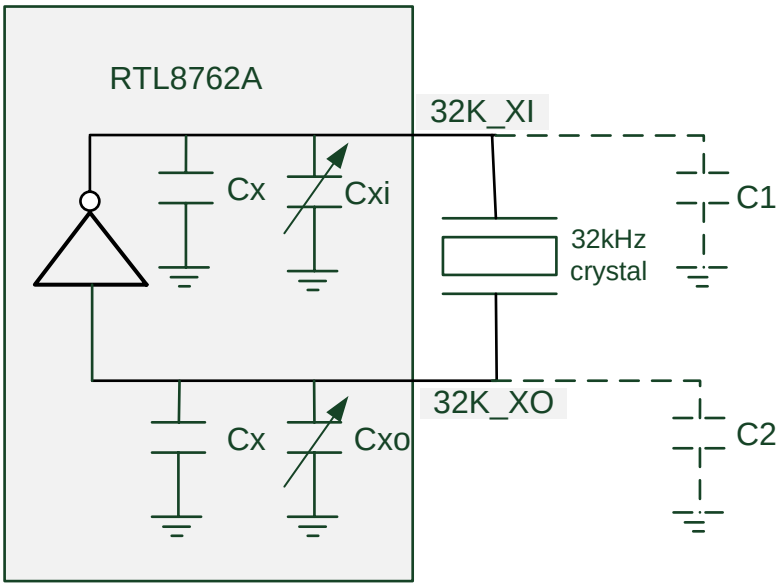


Figure 10. 32kHz Crystal Oscillator Schematic

**Table 6. 32KHz XTAL Specification**

| Parameter | Minimum | Typical | Maximum |
|---|---|---|---|
| Frequency (KHz) | - | 32.768 | - |
| Frequency tolerance (ppm) | - | - | ±20 |
| Load capacitance (pF) | - | 7 | - |
| Drive Level (μW) | - | - | 0.5 |
| Equivalent Series Resistance (KOhm) | - | - | 90 |
| Insulation Resistance (MOhm) | 500 | - | - |

# 9. Power Management Unit (PMU)

The RTL8762A is supplied with 1.8V to 3.3V by a single power source. For more flexibility of peripheral usage, IO voltage (VDDIO) can be different from VBAT (but VDDIO should be less than or equal to VBAT). There is a high-efficiency BUCK regulator to provide power to the digital core circuit and radio circuit.

The RTL8762A defines three PMU power states for various conditions.

**Active Mode:** All clock and power is turned on. All functions operate in this mode.

**Deep LPS Mode**: High-speed clock and core domain power is turned off. The CPU stops running. Data can be retained in retention SRAM.

**Power Down Mode:** Except in an 'always-on' power domain, all clock sources and power are turned off. Power down mode can only be woken by GPIO pins.

# 10. Peripheral Interface Descriptions

The RTL8762A series peripheral descriptions are shown in the table below.

**Table 7. Peripheral Interface Descriptions**

| Physical Address | IP Function |
|---|---|
| 0x4000_0000 - 0x4000_0FFF | SYS Control |
| 0x4000_0100 - 0x4000_012F | RTC |
| 0x4000_0364 - 0x4000_0387 | PWM |
| 0x4000_1000 - 0x4000_17FF | GPIO |
| 0x4000_2000 - 0x4000_2FFF | Timer |
| 0x4000_3000 - 0x4000_33FF | Internal use |
| 0x4000_4000 - 0x4000_402B | Quad Decoder |
| 0x4000_4030 - 0x4000_4053 | 2-wire SPI |
| 0x4000_6000 - 0x4000_6FFF | Key Scan |
| 0x4000_8000 - 0x4000_8FFF | Internal use |
| 0x4001_0000 - 0x4001_0FFF | ADC |
| 0x4004_0000 - 0x4004_03FF | Data UART |
| 0x4004_1000 - 0x4004_1FFF | IR RC |
| 0x4004_2000 - 0x4004_23FF | SPI_0 |
| 0x4004_2400 - 0x4004_27FF | SPI_1 |
| 0x4004_4000 - 0x4004_43FF | I2C_0 |
| 0x4004_4400 - 0x4004_47FF | I2C_1 |
| 0x4006_0000 - 0x4006_07FF | GDMA |

## 10.1. Pin Multiplexer

All GPIO pins in the RTL8762A are configurable via the built-in pin multiplexer. Table 8 shows all GPIO pin configurations. Figure 11, page 21 shows the PINMUX and GPIO PADs control path. In the RTL8762A, all pins have an internal pull-up and pull-down resistor for controlling GPIO_PU and GPIO_PD.

**Table 8. Pin Multiplexer**

| [Decimal Number]: [Pin Function] | | | |
|---|---|---|---|
| 0: Idle mode | 25: qdec_phase_a_z | 50: SPI2W_DATA (master only) | 75: KEY_ROW_0 |
| 1: HCI_UART_TX | 26: qdec_phase_b_z | 51: SPI2W_CLK (master only) | 76: KEY_ROW_1 |
| 2: HCI_UART_RX | 27: LOG_UART_TX | 52: SPI2W_CS (master only) | 77: KEY_ROW_2 |
| 3: HCI_UART_CTS | 28: LOG_UART_RX | 53: SWD_CLK | 78: KEY_ROW_3 |
| 4: HCI_UART_RTS | 29: IRDA_TX | 54: SWD_DIO | 79: KEY_ROW_4 |
| 5: I2C0_CLK | 30: IRDA_RX | 55: KEY_COL_0 | 80: KEY_ROW_5 |
| 6: I2C0_DAT | 31: RTC_CLK_OUT | 56: KEY_COL_1 | 81: KEY_ROW_6 |
| 7: Not used | 32: DATA_UART_TX | 57: KEY_COL_2 | 82: KEY_ROW_7 |
| 8: Not used | 33: DATA_UART_RX | 58: KEY_COL_3 | 83: DMIC_DATA |
| 9: I2C1_CLK | 34: DATA_UART_CTS | 59: KEY_COL_4 | 84: DMIC_CLK |
| 10: I2C1_DAT | 35: DATA_UART_RTS | 60: KEY_COL_5 | 85: DW_GPIOA |
| 11: Not used | 36: SPI1_SS_N_0 (master only) | 61: KEY_COL_6 | - |
| 12: Not used | 37: SPI1_SS_N_1 (master only) | 62: KEY_COL_7 | - |
| 13: timer_pwm0 | 38: SPI1_SS_N_2 (master only) | 63: KEY_COL_8 | - |
| 14: timer_pwm1 | 39: SPI1_CLK (master only) | 64: KEY_COL_9 | - |
| 15: timer_pwm2 | 40: SPI1_MO (master only) | 65: KEY_COL_10 | - |
| 16: timer_pwm3 | 41: SPI1_MI (master only) | 66: KEY_COL_11 | - |
| 17: timer_etet0 | 42: SPI0_SS_N_0 (slave) | 67: KEY_COL_12 | - |
| 18: timer_ete1 | 43: SPI0_CLK (slave) | 68: KEY_COL_13 | - |
| 19: timer_ete2 | 44: SPI0_SO (slave) | 69: KEY_COL_14 | - |
| 20: timer_ete3 | 45: SPI0_SI (slave) | 70: KEY_COL_15 | - |
| 21: qdec_phase_a_x | 46: SPI0_SS_N_0 (master only) | 71: KEY_COL_16 | - |
| 22: qdec_phase_b_x | 47: SPI0_CLK (master only) | 72: KEY_COL_17 | - |
| 23: qdec_phase_a_y | 48: SPI0_MO (master only) | 73: KEY_COL_18 | - |
| 24: qdec_phase_b_y | 49: SPI0_MI (master only) | 74: KEY_COL_19 | - |

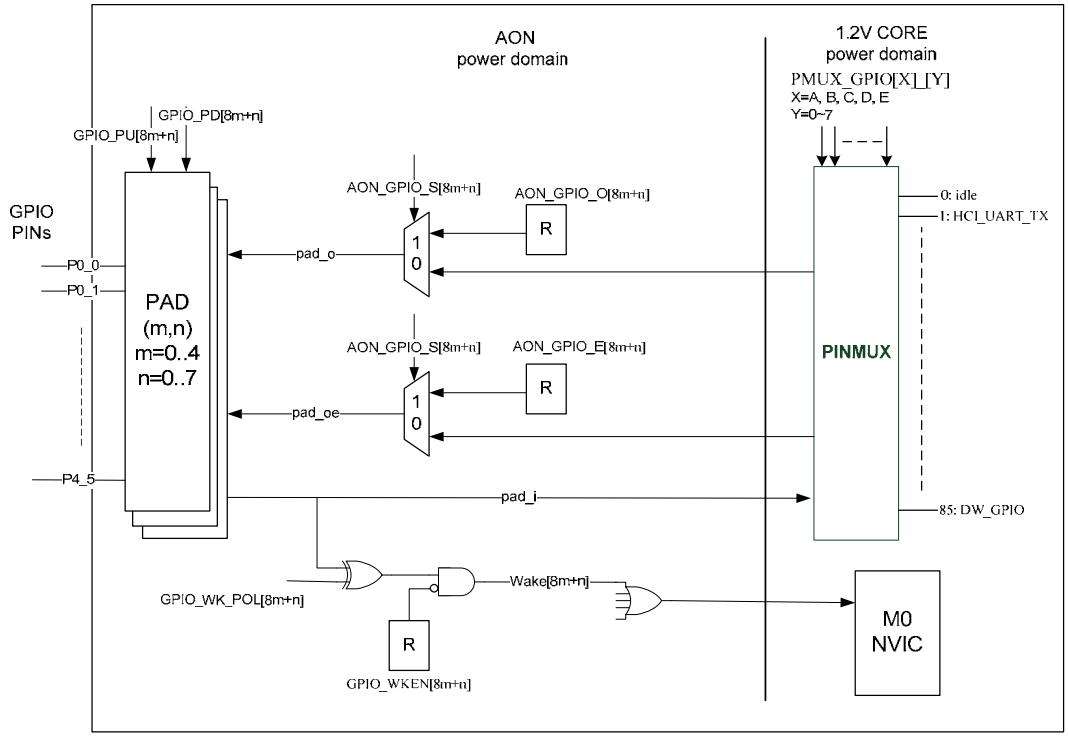


**Figure 11. PINMUX and GPIO PADs Control Path**

**Table 9. Pin Multiplexer (Base Address: 0x4000_0000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_GPIOA0~3** | | | | | |
| 0x280 | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | d27 | PMUX_GPIOA_3 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[3] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | 0 | PMUX_GPIOA_2 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[2] |
| | [15] | - | - | - | Reserved |
| | [14:8] | R/W | 0 | PMUX_GPIOA_1 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[1] |
| | [7] | - | - | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [6:0] | R/W | 0 | PMUX_GPIOA_0 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[0] |
| **REG_GPIOA4~7** | | | | | |
| 0x284 | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | 0 | PMUX_GPIOA_7 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[7] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | 0 | PMUX_GPIOA_6 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[6] |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | 0 | PMUX_GPIOA_5 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[5] |
| | [7] | R/W | 0 | - | Reserved |
| | [6:0] | R/W | 0 | PMUX_GPIOA_4 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[4] |
| **REG_GPIOB0~3** | | | | | |
| 0x288 | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | 0 | PMUX_GPIOB_3 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[11] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | 0 | PMUX_GPIOB_2 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[10] |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | d53 | PMUX_GPIOB_1 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[9] |
| | [7] | R/W | 0 | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [6:0] | R/W | d54 | PMUX_GPIOB_0 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[8] |
| **REG_GPIOB4~7** | | | | | |
| 0x28C | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | 0 | PMUX_GPIOB_7 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[15] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | 0 | PMUX_GPIOB_6 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[14] |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | 0 | PMUX_GPIOB_5 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[13] |
| | [7] | R/W | 0 | - | Reserved |
| | [6:0] | R/W | 0 | PMUX_GPIOB_4 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[12] |
| **REG_GPIOC0~3** | | | | | |
| 0x290 | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | 0 | PMUX_GPIOC_3 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[19] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | 0 | PMUX_GPIOC_2 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[18] |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | 0 | PMUX_GPIOC_1 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[17] |
| | [7] | R/W | 0 | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [6:0] | R/W | 0 | PMUX_GPIOC_0 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[16] |
| **REG_GPIOC4~7** | | | | | |
| 0x294 | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | 0 | PMUX_GPIOC_7 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[23] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | 0 | PMUX_GPIOC_6 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[22] |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | d5 | PMUX_GPIOC_5 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[21] |
| | [7] | R/W | 0 | - | Reserved |
| | [6:0] | R/W | d6 | PMUX_GPIOC_4 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[20] |
| **REG_GPIOD0~3** | | | | | |
| 0x298 | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | d4 | PMUX_GPIOD_3 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[27] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | d3 | PMUX_GPIOD_2 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[26] |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | d2 | PMUX_GPIOD_1 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[25] |
| | [7] | R/W | 0 | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [6:0] | R/W | d1 | PMUX_GPIOD_0 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[24] |
| **REG_GPIOD4~7** | | | | | |
| 0x29C | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | 0 | PMUX_GPIOD_7 | Refer to Table 8 Pin Multiplexer, page 20.<br>DW_GPIO function not supported |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | 0 | PMUX_GPIOD_6 | Refer to Table 8 Pin Multiplexer, page 20.<br>DW_GPIO function not supported |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | 0 | PMUX_GPIOD_5 | Refer to Table 8 Pin Multiplexer, page 20.<br>DW_GPIO function not supported |
| | [7] | R/W | 0 | - | Reserved |
| | [6:0] | R/W | 0 | PMUX_GPIOD_4 | Refer to Table 8 Pin Multiplexer, page 20.<br>DW_GPIO function not supported |
| **REG_GPIOE0~3** | | | | | |
| 0x2A0 | [31] | R/W | 0 | - | Reserved |
| | [30:24] | R/W | d42 | PMUX_GPIOE_3 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[31] |
| | [23] | R/W | 0 | - | Reserved |
| | [22:16] | R/W | d45 | PMUX_GPIOE_2 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[30] |
| | [15] | R/W | 0 | - | Reserved |
| | [14:8] | R/W | d44 | PMUX_GPIOE_1 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[29] |
| | [7] | R/W | 0 | - | Reserved |
| | [6:0] | R/W | d43 | PMUX_GPIOE_0 | Refer to Table 8 Pin Multiplexer, page 20; only differences are listed in the following description.<br>[Decimal number]: [GPIO mode]<br>85: DW_GPIOA[28] |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_GPIOE4~5** | | | | | |
| 0x2A4 | [31:15] | R/W | - | - | Reserved |
| | [14:8] | R/W | 0 | PMUX_GPIOE_5 | Refer to Table 8 Pin Multiplexer, page 20.<br>DW_GPIO function not supported |
| | [7] | R/W | - | - | Reserved |
| | [6:0] | R/W | 0 | PMUX_GPIOE_4 | Refer to Table 8 Pin Multiplexer, page 20.<br>DW_GPIO function not supported |

## *10.2. Real-Time Counter (RTC)*

There are 24-bit counters with four individual comparators. The counter is clocked by an internal 32KHz clock with 12-bit pre-scalar. The comparators output can interrupt the CPU and wake up the chip from DLPS mode. The RTC block diagram is shown below.

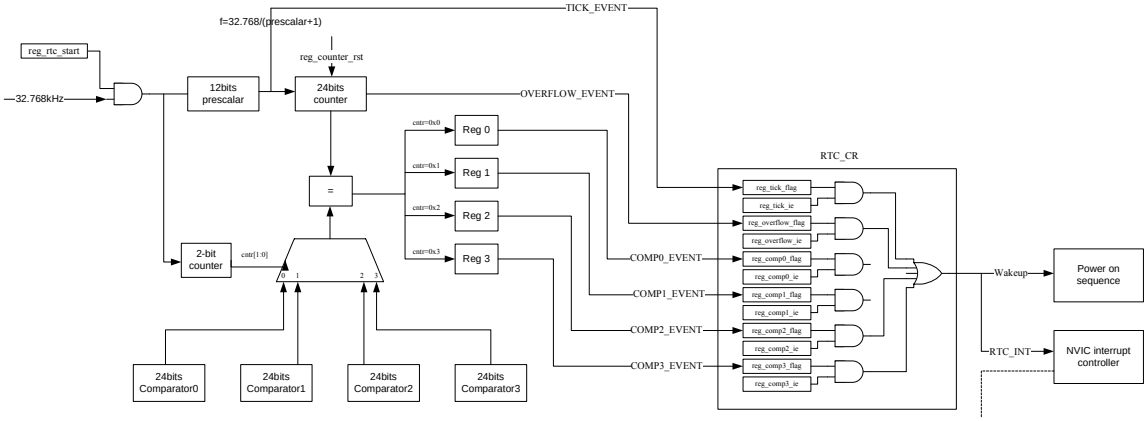


**Figure 12. RTC Block Diagram**

**Table 10. RTC (Base Address: 0x4000_0000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **0x100 RTC Controller Register0 (RTC_CR0)** | | | | | |
| 0x100 | [31] | R/W | 0x0 | reg_rtc_clk_sel | RTC clock source sel<br>0x0: From BTAON (xtal or ring osc)<br>0x1: From 32k sdm |
| | [30] | R/W | 0x0 | reg_rtc_tick_flag_on | Enable or disable RTC tick function<br>0x1: Enable<br>0x0: Disable |
| | [29:20] | - | - | - | Reserved |
| | [19] | R/W | 0x0 | reg_comp3_on | Enable or disable compare3 function<br>0x1: Enable<br>0x0: Disable |
| | [18] | R/W | 0x0 | reg_comp2_on | Enable or disable compare2 function<br>0x1: Enable<br>0x0: Disable |
| | [17] | R/W | 0x0 | reg_comp1_on | Enable or disable compare1 function<br>0x1: Enable<br>0x0: Disable |
| | [16] | R/W | 0x0 | reg_comp0_on | Enable or disable compare0 function<br>0x1: Enable<br>0x0: Disable |
| | [15:8] | - | - | - | Reserved |
| | [7] | R/W | 0x0 | reg_counter_overflow_clk_en | Switch RTC counter clock to 32KHz<br>0x1: Enable<br>0x0: Disable |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [6] | R/W | 0x1 | reg_lpcomp_counter_start | Start or stop LPCOMP counter<br>0x1: Start LPCOMP counter<br>0x0: Stop LPCOMP counter |
| | [5] | - | - | - | Reserved |
| | [4] | R/W | 0x0 | reg_lpcomp_counter_rst | Reset LPCOMP Counter<br>0x1: Reset LPCOMP Counter to 0 |
| | [3] | R/W | 0x0 | reg_div_counter_rst | Reset divider Counter<br>0x1: Reset divider Counter to 0 |
| | [2] | R/W | 0x0 | reg_counter_rst | Reset Counter<br>0x1: Reset Counter to 0 |
| | [1] | - | - | - | Reserved |
| | [0] | R/W | 0x1 | reg_rtc_start | Start or stop RTC<br>0x1: Start RTC.<br>0x0: Stop RTC. |
| **0x104 RTC Controller Register1 (RTC_CR1)** | | | | | |
| 0x104 | [31:22] | - | - | - | Reserved |
| | [21] | R/W | 0x0 | reg_lpcomp_src_int_en | Enable lpcomp out sync signal to CPU interrupt<br>0x1: Enable<br>0x0: Disable |
| | [20] | R/W | 0x0 | reg_rtc_wakeup_en | Enable wakeup signal to BTAON<br>0x1: Enable<br>0x0: Disable |
| | [19] | R/W | 0x0 | reg_comp3_ie | Enable or disable compare3 event<br>0x1: Enable<br>0x0: Disable |
| | [18] | R/W | 0x0 | reg_comp2_ie | Enable or disable compare2 event<br>0x1: Enable<br>0x0: Disable |
| | [17] | R/W | 0x0 | reg_comp1_ie | Enable or disable compare1 event<br>0x1: Enable<br>0x0: Disable |
| | [16] | R/W | 0x0 | reg_comp0_ie | Enable or disable compare0 event<br>0x1: Enable<br>0x0: Disable |
| | [15:5] | - | - | - | Reserved |
| | [4] | W | 0x0 | reg_lpcomp_ie | Enable or disable LPCOMP event<br>0x1: Enable<br>0x0: Disable |
| | [3:2] | - | - | - | Reserved |
| | [1] | R/W | 0x0 | reg_overflow_ie | Enable or disable overflow event<br>0x1: Enable<br>0x0: Disable |
| | [0] | R/W | 0x1 | reg_tick_ie | Enable or disable tick event<br>0x1: Enable<br>0x0: Disable |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **0x108 RTC Status Register (RTC_SR)** | | | | | |
| 0x108 | [31:20] | - | - | - | Reserved |
| | [19] | R/W | 0x0 | reg_comp3_flag | Event Status of Comparator3<br>This flag is set by hardware when a comparator3 match event occurs while reg_comp3_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [18] | R/W | 0x0 | reg_comp2_flag | Event Status of Comparator2<br>This flag is set by hardware when a comparator2 match event occurs while reg_comp2_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [17] | R/W | 0x0 | reg_comp1_flag | Event Status of Comparator1<br>This flag is set by hardware when a comparator1 match event occurs while reg_comp1_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [16] | R/W | 0x0 | reg_comp0_flag | Event Status of Comparator0<br>This flag is set by hardware when a comparator0 match event occurs while reg_comp0_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [15:5] | - | - | - | Reserved |
| | [4] | R/W | 0x0 | reg_lpcomp_flag | Event Status of LPCOMP<br>This flag is set by hardware when a LPCOMP comparator match event occurs while reg_lpcomp_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [3:2] | - | - | - | Reserved |
| | [1] | R/W | 0x0 | reg_overflow_flag | Event Status of Overflow<br>This flag is set by hardware when a counter overflow event occurs while reg_overflw_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [0] | R/W | 0x0 | reg_tick_flag | Event Status of Tick<br>This flag is set by hardware when a counter overflow event occurs while reg_tick_ie is already set.<br>This flag is cleared by software by writing 1. |
| **0x10C RTC_PRESCALER** | | | | | |
| 0x10C | [31:12] | - | - | - | Reserved |
| | [11:0] | R/W | 0x0 | reg_prescaler_val | Prescaler Value<br>12 bit prescaler for COUNTER frequency (32768/(PRESCALER+1)).<br>Must be written when RTC is stopped. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **0x110 RTC_COMP0** | | | | | |
| 0x110 | [31:24] | - | - | - | Reserved |
| | [23:0] | R/W | 0x0 | reg_comp0 | Value of comparator0 |
| **0x114 RTC_COMP1** | | | | | |
| 0x114 | [31:24] | - | - | - | Reserved |
| | [23:0] | R/W | 0x0 | reg_comp1 | Value of comparator1 |
| **0x118 RTC_COMP2** | | | | | |
| 0x118 | [31:24] | - | - | - | Reserved |
| | [23:0] | R/W | 0x0 | reg_comp2 | Value of comparator2 |
| **0x11C RTC_COMP3** | | | | | |
| 0x11C | [31:24] | - | - | - | Reserved |
| | [23:0] | R/W | 0x0 | reg_comp3 | Value of comparator3 |
| **0x120 RTC_LPCOMP_CMP** | | | | | |
| 0x120 | [31:12] | - | - | - | Reserved |
| | [11:0] | R/W | 0x0 | reg_lpcomp_comp | Value of LPCOMP counter's comparator value |
| **0x124 RTC_CNT** | | | | | |
| 0x124 | [31:24] | - | - | - | Reserved |
| | [23:0] | R | 0x0 | reg_counter | Counter Value |
| **0x128 RTC_LPCOMP_CNT** | | | | | |
| 0x128 | [31:12] | - | - | - | Reserved |
| | [11:0] | R | 0x0 | reg_lpcomp_counter | Value of LPCOMP async counter's value |
| **0x12C RTC Status Register (Off Domain)** | | | | | |
| 0x12C | [31] | R/W | 0x1 | reg_rtc_int_ctrl_en | RTC interrupt control enable signal<br>0x1: Enable off-domain RTC intrrupt control<br>0x0: Disable |
| | [30:20] | - | - | - | Reserved |
| | [19] | R/W | 0x0 | reg_comp3_flag | Event Status of Comparator3<br>This flag is set by hardware when a comparator3 match event occurs while reg_comp3_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [18] | R/W | 0x0 | reg_comp2_flag | Event Status of Comparator2<br>This flag is set by hardware when a comparator2 match event occurs while reg_comp2_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [17] | R/W | 0x0 | reg_comp1_flag | Event Status of Comparator1<br>This flag is set by hardware when a comparator1 match event occurs while reg_comp1_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [16] | R/W | 0x0 | reg_comp0_flag | Event Status of Comparator0<br>This flag is set by hardware when a comparator0 match event occurs while reg_comp0_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [15:5] | - | - | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [4] | R/W | 0x0 | reg_lpcomp_flag | Event Status of LPCOMP<br>This flag is set by hardware when a LPCOMP comparator match event occurs while reg_lpcomp_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [3:2] | - | - | - | Reserved |
| | [1] | R/W | 0x0 | reg_overflow_flag | Event Status of Overflow<br>This flag is set by hardware when a counter overflow event occurs while reg_overflw_ie is already set.<br>This flag is cleared by software by writing 1. |
| | [0] | R/W | 0x0 | reg_tick_flag | Event Status of Tick<br>This flag is set by hardware when a counter overflow event occurs while reg_tick_ie is already set.<br>This flag is cleared by software by writing 1. |

## *10.3. PWM*

The RTL8762A supports four channel PWM outputs that are individually clocked by a hardware timer. The clock source of the hardware timer is selected from a 32KHz or 10MHz clock as shown below.

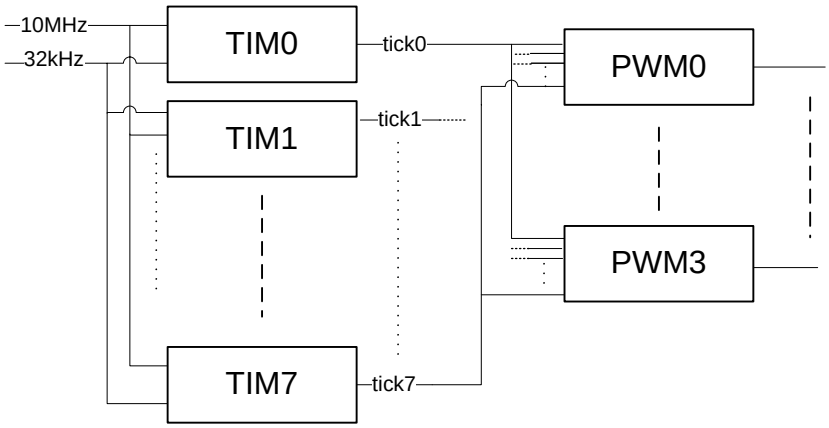


**Figure 13. PWM is Clocked by Hardware Timer**

**Table 11. PWM (Base Address: 0x4000_0000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_PERI_PWM0_CTRL_0** | | | | | |
| 0x364 | [31:9] | R/W | - | - | Reserved |
| | [4:1] | R/W | 2 | BIT_PERI_PWM0_GT_SEL | 0: GTIMER 0<br>1: GTIMER 1<br>…<br>7: GTIMER 7<br>Etc. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [0] | R/W | 0 | BIT_PERI_PWM0_EN | Enable PWM0 |
| **REG_PERI_PWM0_CTRL_1** | | | | | |
| 0x368 | [31:16] | R/W | 0 | BIT_PERI_PWM0_DUTY | The on-duty duration of PWM pulse. |
| | [15:0] | R/W | 0 | BIT_PERI_PWM0_PERIOD | The period of PWM pulse. The time unit is configured from GTIMER specified by BIT_PWM1_GT_SEL field. |
| **REG_PERI_PWM1_CTRL_0** | | | | | |
| 0x36C | [31:9] | R/W | - | - | Reserved |
| | [4:1] | R/W | 2 | BIT_PERI_PWM1_GT_SEL | 0: GTIMER 0<br>1: GTIMER 1<br>…<br>7: GTIMER 7<br>Etc. |
| | [0] | R/W | 0 | BIT_PERI_PWM1_EN | Enable PWM1 |
| **REG_PERI_PWM1_CTRL_1** | | | | | |
| 0x370 | [31:16] | R/W | 0 | BIT_PERI_PWM1_DUTY | The on-duty duration of PWM pulse. |
| | [15:0] | R/W | 0 | BIT_PERI_PWM1_PERIOD | The period of PWM pulse. The time unit is configured from GTIMER specified by BIT_PWM1_GT_SEL field. |
| **REG_PERI_PWM2_CTRL_0** | | | | | |
| 0x374 | [31:9] | R/W | - | - | Reserved |
| | [4:1] | R/W | 2 | BIT_PERI_PWM2_GT_SEL | 0: GTIMER 0<br>1: GTIMER 1<br>…<br>7: GTIMER 7<br>Etc. |
| | [0] | R/W | 0 | BIT_PERI_PWM2_EN | Enable PWM2 |
| **REG_PERI_PWM2_CTRL_1** | | | | | |
| 0x378 | [31:16] | R/W | 0 | BIT_PERI_PWM2_DUTY | The on-duty duration of PWM pulse. |
| | [15:0] | R/W | 0 | BIT_PERI_PWM2_PERIOD | The period of PWM pulse. The time unit is configured from GTIMER specified by BIT_PWM1_GT_SEL field. |
| **REG_PERI_PWM3_CTRL_0** | | | | | |
| 0x37C | [31:9] | R/W | - | - | Reserved |
| | [4:1] | R/W | 2 | BIT_PERI_PWM3_GT_SEL | 0: GTIMER 0<br>1: GTIMER 1<br>…<br>7: GTIMER 7<br>Etc. |
| | [0] | R/W | 0 | BIT_PERI_PWM3_EN | Enable PWM3 |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_PERI_PWM3_CTRL_1** | | | | | |
| 0x380 | [31:16] | R/W | 0 | BIT_PERI_PWM3_DUTY | The on-duty duration of PWM pulse. |
| | [15:0] | R/W | 0 | BIT_PERI_PWM3_PERIOD | The period of PWM pulse. The time unit is configured from GTIMER specified by BIT_PWM1_GT_SEL field. |
| **REG_PERI_TIM_EVT_CTRL** | | | | | |
| 0x384 | [31] | R/W | 0 | BIT_PERI_GT_EVT3_EN | Enable timer event |
| | [30:28 | R/W | 0 | BIT_PERI_GT_EVT3_SRC_SEL | 0: Timer event from timer 0<br>7: Timer event from timer 7 |
| | [27:24] | R/W | 0 | BIT_PERI_GT_EVT3_PULSE_DUR | The event timer output pulse duration<br>0: 32µs<br>1: 64µs<br>2: 128µs<br>3: 256µs |
| | [23] | R/W | 0 | BIT_PERI_GT_EVT2_EN | Enable timer event |
| | [22:20 | R/W | 0 | BIT_PERI_GT_EVT2_SRC_SEL | 0: Timer event from timer 0<br>7: Timer event from timer 7 |
| | [19:16] | R/W | 0 | BIT_PERI_GT_EVT2_PULSE_DUR | The event timer output pulse duration<br>0: 32µs<br>1: 64µs<br>2: 128µs<br>3: 256µs |
| | [15] | R/W | 0 | BIT_PERI_GT_EVT1_EN | Enable timer event |
| | [14:12] | R/W | 0 | BIT_PERI_GT_EVT1_SRC_SEL | 0: Timer event from timer 0<br>7: Timer event from timer 7 |
| | [11:8] | R/W | 0 | BIT_PERI_GT_EVT1_PULSE_DUR | The event timer output pulse duration<br>0: 32µs<br>1: 64µs<br>2: 128µs<br>3: 256µs |
| | [7] | R/W | 0 | BIT_PERI_GT_EVT0_EN | Enable timer event |
| | [6:4] | R/W | 0 | BIT_PERI_GT_EVT0_SRC_SEL | 0: Timer event from timer 0<br>7: Timer event from timer 7 |
| | [3:0] | R/W | 0 | BIT_PERI_GT_EVT0_PULSE_DUR | The event timer output pulse duration<br>0: 32µs<br>1: 64µs<br>2: 128µs<br>3: 256µs |

## 10.4. GPIO Control

Table 12. GPIO Control (Base Address: 0x4000_1000)

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **gpio_swporta_dr: Port A Data Register** | | | | | |
| 0x00 | [31:0] | R/W | 0x0 | gpio_swporta_dr | Values written to this register are output on the I/O signals for Port A if the corresponding data direction bits for Port A are set to Output mode and the corresponding control bit for Port A is set to Software mode. |
| **gpio_swporta_ddr: Port A Data Direction Register** | | | | | |
| 0x04 | [31:0] | R/W | 0x0 | Port A Data Direction Register | Values written to this register independently control the direction of the corresponding data bit in Port A. The default direction can be configured as input or output after system reset through the GPIO_DFLT_DIR_A parameter.<br>0: Input (default)<br>1: Output |
| **gpio_swporta_ctl: Port A Data Source** | | | | | |
| 0x08 | [31:0] | R/W | 0x0 | Port A Data Source | The data and control source for a signal can come from either software or hardware; this bit selects between them.<br>0: Software mode (default)<br>1: Hardware mode |
| **gpio_inten: Interrupt Enable** | | | | | |
| 0x30 | [31:0] | R/W | 0x0 | Interrupt enable | Allows each bit of Port A to be configured for interrupts. By default the generation of interrupts is disabled. Whenever a 1 is written to a bit of this register, it configures the corresponding bit on Port A to become an interrupt; otherwise, Port A operates as a normal GPIO signal. Interrupts are disabled on the corresponding bits of Port A if the corresponding data direction register is set to Output or if Port A mode is set to Hardware.<br>0: Configure Port A bit as normal GPIO signal (default)<br>1: Configure Port A bit as interrupt |
| **gpio_intmask: Interrupt Mask** | | | | | |
| 0x34 | [31:0] | R/W | 0x0 | Interrupt mask | Controls whether an interrupt on Port A can create an interrupt for the interrupt controller by not masking it. By default, all interrupts bits are unmasked. Whenever a 1 is written to a bit in this register, it masks the interrupt generation capability for this signal; otherwise interrupts are allowed through. The unmasked status can be read as well as the resultant status after masking.<br>0: Interrupt bits are unmasked (default)<br>1: Mask interrupt |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **gpio_inttype_level: Interrupt Level** | | | | | |
| 0x38 | [31:0] | R/W | 0x0 | Interrupt level | Controls the type of interrupt that can occur on Port A.<br>Whenever a 0 is written to a bit of this register, it configures the interrupt type to be level-sensitive; otherwise, it is edge-sensitive.<br>0: Level-sensitive (default)<br>1: Edge-sensitive |
| **gpio_int_polarity: Interrupt Polarity** | | | | | |
| 0x3C | [31:0] | R/W | 0x0 | Interrupt polarity | Controls the polarity of edge or level sensitivity that can occur on input of Port A.<br>Whenever a 0 is written to a bit of this register, it configures the interrupt type to falling-edge or active-low sensitive; otherwise, it is rising-edge or active-high<br>sensitive.<br>0: Active-low (default)<br>1: Active-high |
| **gpio_intstatus: Interrupt Status** | | | | | |
| 0x40 | [31:0] | R | 0x0 | Interrupt status | Interrupt status of Port A |
| **gpio_porta_eoi: Clear Interrupt** | | | | | |
| 0x4C | [31:0] | R/W | 0x0 | Clear interrupt | Controls the clearing of edge type interrupts from Port A. When a 1 is written into a corresponding bit of this register, the interrupt is cleared. All interrupts are cleared when Port A is not configured for interrupts.<br>0: No interrupt clear (default)<br>1: Clear interrupt |
| **gpio_ext_porta: External Port A** | | | | | |
| 0x50 | [31:0] | R | 0x0 | External Port A | When Port A is configured as Input, then reading this location reads the values on the signal. When the data direction of Port A is set as Output, reading this location reads the data register for Port A. |

**Table 13. GPIO Control (Base Address: 0x4000_8000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_LOW_PRI_INT_STATUS** | | | | | |
| 0x04 | [31:17] | R/W | 0 | - | Reserved |
| | [16] | R/W | 0 | gpio_intr[31:6] | DW GPIO interrupt status [31:6] (OR'ed) |
| | [15:10] | R/W | 0 | - | Reserved |
| | [9] | R/W | 0 | timer_intr[7] | DW Timer interrupt status [7] |
| | [8] | R/W | 0 | timer_intr[6] | DW Timer interrupt status [6] |
| | [7] | R/W | 0 | timer_intr[5] | DW Timer interrupt status [5] |
| | [6] | R/W | 0 | timer_intr[4] | DW Timer interrupt status [4] |
| | [5] | R/W | 0 | timer_intr[3] | DW Timer interrupt status [3] |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [4] | R/W | 0 | gpio_intr[5] | DW GPIO interrupt status [5] |
| | [3] | R/W | 0 | gpio_intr[4] | DW GPIO interrupt status [4] |
| | [2] | R/W | 0 | gpio_intr[3] | DW GPIO interrupt status [3] |
| | [1] | R/W | 0 | gpio_intr[2] | DW GPIO interrupt status [2] |
| | [0] | R/W | 0 | spi_intr | SPI flash controller interrupt status |
| **REG_LOW_PRI_INT_MODE** | | | | | |
| 0x08 | [31:0] | R/W | 0 | BIT_LOW_PRI_INT_ MODE | Each bit control corresponding interrupt mode.<br>1: Rising edge trigger<br>0: High level active |
| **REG_LOW_PRI_INT_EN** | | | | | |
| 0x0c | [31:0] | R/W | 0 | BIT_LOW_PRI_INT_ MODE | Each bit control corresponding interrupt mode.<br>1: Rising edge trigger<br>0: High level active |

## *10.5. Hardware Timer (TIM)*

There are eight programmable 32-bit timers with free-running and user-defined count modes. The Timer clock source selection is shown below.

**Table 14. Hardware Timer (Base Address: 0x4000_0000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_PERI_GTIMER_CLK_SRC** | | | | | |
| 0x360 | [31:14] | R/W | 0 | - | Reserved |
| | [13] | R/W | 0 | - | Reserved |
| | [12:8] | R/W | 0 | - | Reserved |
| | [7] | R/W | 0 | BIT_PERI_GT7_CLK_SEL | GTIMER7 clock source<br>0: 32Khz, 1: 10Mhz |
| | [6] | R/W | 0 | BIT_PERI_GT6_CLK_SEL | GTIMER6 clock source<br>0: 32Khz, 1: 10Mhz |
| | [5] | R/W | 0 | BIT_PERI_GT5_CLK_SEL | GTIMER5 clock source<br>0: 32Khz, 1: 10Mhz |
| | [4] | R/W | 0 | BIT_PERI_GT4_CLK_SEL | GTIMER4 clock source<br>0: 32Khz, 1: 10Mhz |
| | [3] | R/W | 0 | BIT_PERI_GT3_CLK_SEL | GTIMER3 clock source<br>0: 32Khz, 1: 10Mhz |
| | [2] | R/W | 0 | BIT_PERI_GT2_CLK_SEL | GTIMER2 clock source<br>0: 32Khz, 1: 10Mhz |
| | [1] | R/W | 0 | BIT_PERI_GT1_CLK_SEL | GTIMER1 clock source<br>0: 32Khz, 1: 10Mhz |
| | [0] | R/W | 0 | BIT_PERI_GT0_CLK_SEL | GTIMER0 clock source<br>0: 32Khz, 1: 10Mhz |

**Table 15. Hardware Timer Range (Base Address: 0x4000_2000)**

| Address Range (Base+) | Function |
|---|---|
| 0x00 to 0x10 | Timer 1 register |
| 0x14 to 0x24 | Timer 2 register |
| 0x28 to 0x38 | Timer 3 register |
| 0x3C to 0x4C | Timer 4 register |
| 0x50 to 0x60 | Timer 5 register |
| 0x64 to 0x74 | Timer 6 register |
| 0x78 to 0x88 | Timer 7 register |
| 0x8C to 0x9C | Timer 8 register |

**Table 16. Hardware Timer (Base Address: 0x4000_2000)**

| Address Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **TimerNLoadCount N=1..7** | | | | | |
| 0x00 | [31:0] | R/W | 0xffffffff | TimerN Load Count Register | Value to be loaded into TimerN. This is the value from which counting commences. Any value written to this register is loaded into the associated timer. |
| **TimerNCurrentValue N=1..7** | | | | | |
| 0x04 | [31:0] | R | 0xffffffff | Timer N Current Value | Current Value of TimerN. This register is supported only when timer_N_clk is synchronous to pclk. Reading this register when using independent clocks results in an undefined value. |
| **TimerNControlReg N=1..7** | | | | | |
| 0x08 | [31:3] | - | - | - | Reserved |
| | [2] | R/W | - | Timer Interrupt Mask | Timer interrupt mask for TimerN.<br>0: Not masked<br>1: Masked |
| | [1] | R/W | - | Timer Mode | Timer mode for TimerN.<br>0: Free-running mode<br>1: User-defined count mode |
| | [0] | R/W | - | Timer Enable | Timer enable bit for TimerN.<br>0: Disable<br>1: Enable |
| **TimerNEOI N=1..7** | | | | | |
| 0x0C | [31:1] | - | - | - | Reserved |
| | [0] | R | - | TimerN End-of-Interrupt Register | Reading from this register returns all zeroes (0) and clears the interrupt from TimerN. |
| **TimerNIntStatus N=1..7** | | | | | |
| 0x10 | [31:1] | - | - | - | Reserved |
| | [0] | R | - | TimerN Interrupt Status Register | Contains the interrupt status for TimerN. |

| Address Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **TimersIntStatus** | | | | | |
| 0xA0 | [7:0] | R | - | Timers Interrupt Status Register | Contains the interrupt status of all timers in the component. If a bit of this register is 0, then the corresponding timer interrupt is not active – and the corresponding interrupt could be on either the timer_intr bus or the timer_intr_n bus, depending on the interrupt polarity you have chosen. Similarly, if a bit of this register is 1, then the corresponding interrupt bit has been set in the relevant interrupt bus. In both cases, the status reported is the status after the interrupt mask has been applied.<br>Reading from this register does not clear any active interrupts:<br>0: Either timer_intr or timer_intr_n is not active after masking<br>1: Either timer_intr or timer_intr_n is active after masking |
| **TimersEOI** | | | | | |
| 0xA4 | [7:0] | R | - | Timers End-of-Interrupt Register | Reading this register returns all zeroes (0) and clears all active interrupts. |

## *10.6. Quadrature Decoder*

A three axis (X, Y, and Z-axis) quadrature decoder is built into the RTL8762A. We implement a smart interrupt mechanism to reduce firmware loading. The RTL8762A embeds the input debounce circuitry with the programmable timer.

**Table 17. Quadrature Decoder (Base Address: 0x4000_4000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_00** | | | | | |
| 0x00 | [31] | R | 0 | int_x | Interrupt for X-axis<br>1: Interrupt<br>0: No interrupt |
| | [30] | R | 0 | ov_flag_x | Overflow flag for accumulation counter for X-axis<br>0: No overflow occurred<br>1: Overflow occurred; 0xFFFF--> 0x0000 |
| | [29] | R | 0 | step_2_flag_x | Illegal state flag for X-axis<br>0: No illegal state occurred<br>1: Illegal state occurred |
| | [28] | R | 0 | dir_x | Direction for X-axis<br>0: Down<br>1: Up |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [27] | R | 0 | ud_flag_x | Underflow flag for accumulation counter for X-axis<br>0: No underflow occurred<br>1: Underflow occurred, 0x0000--> 0xFFFF |
| | [23:16] | R | 0 | step_2_cntr_x | Step2 accumulation value for X-axis<br>0~255<br>Step is +1<br>This counter is for debug purpose.<br>When state change is 2-bit toggle, this is an illegal state and the counter is increased by one. |
| | [15:0] | R | 0 | acc_cntr_x | Accumulation value for X-axis<br>0x0000 to 0xFFFF<br>Step is +1 or -1<br>When direction is up, the counter is increased by one during each state change.<br>When direction is down, the counter is decreased by one during each state change.<br>The counter is a "wrap-around" type. |
| **REG_04** | | | | | |
| 0x04 | [31] | R | 0 | int_y | Interrupt for Y-axis<br>1: Interrupt<br>0: No interrupt |
| | [30] | R | 0 | ov_flag_y | Overflow flag for accumulation counter for Y-axis<br>0: No overflow occurred<br>1: Overflow occurred, 0xFFFF--> 0x0000 |
| | [29] | R | 0 | step_2_flag_y | Illegal state flag for Y-axis<br>0: No illegal state occurred<br>1: Illegal state occurred |
| | [28] | R | 0 | dir_y | Direction for Y-axis<br>0: Down<br>1: Up |
| | [27] | R | 0 | ud_flag_y | Underflow flag for accumulation counter for Y-axis<br>0: No underflow occurred<br>1: Underflow occurred, 0x0000--> 0xFFFF |
| | [23:16] | R | 0 | step_2_cntr_y | Step2 accumulation value for Y-axis<br>0~255<br>Step is +1<br>This counter is for debug purposes.<br>When state change is 2-bit toggle, this is an illegal state and the counter is increased by one. |
| | [15:0] | R | 0 | acc_cntr_y | Accumulation value for Y-axis<br>0x0000 to 0xFFFF<br>Step is +1 or -1<br>When direction is up, the counter is increased by one during each state change.<br>When direction is down, the counter is decrease by one during each state change.<br>The counter is a ‘wrap-around’ type. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_08** | | | | | |
| 0x08 | [31] | R | 0 | int_z | Interrupt for Z-axis<br>1: Interrupt<br>0: No interrupt |
| | [30] | R | 0 | ov_flag_z | Overflow flag for accumulation counter for Z-axis<br>0: No overflow occurred<br>1: Overflow occurred, 0xFFFF--> 0x0000 |
| | [29] | R | 0 | step_2_flag_z | Illegal state flag for Z-axis<br>0: No illegal state occurred<br>1: Illegal state occurred |
| | [28] | R | 0 | dir_z | Direction for Z-axis<br>0: Down<br>1: Up |
| | [27] | R | 0 | ud_flag_z | Underflow flag for accumulation counter for Z-axis<br>0: No underflow occurred<br>1: Underflow occurred, 0x0000--> 0xFFFF |
| | [23:16] | R | 0 | step_2_cntr_z | Step2 accumulation value for Z-axis<br>0~255<br>Step is +1<br>This counter is for debug purpose.<br>When state change is 2-bit toggle, this is an illegal state and the counter is increased by one. |
| | [15:0] | R | 0 | acc_cntr_z | Accumulation value for Z-axis<br>0x0000 to 0xFFFF<br>Step is +1 or -1<br>When direction is up, the counter is increased by one during each state change.<br>When direction is down, the counter is decrease by one during each state change.<br>The counter is a ‘wrap-around’ type. |
| **REG_0C** | | | | | |
| 0x0C | [27:16] | R/W | 0xF9 | reg_deb_time | SW must program debounce time before de-noise filter is enabled.<br>The stable period=(reg_deb_time+1) * 1/(quad decoder sampling clock)<br>Please refer 0x28 for the description of how to decide sampling clock of quad decoder |
| | [8] | R/W | 0 | reg_pause_en_z | 0: Resume the accumulation counter<br>1: Pause the accumulation counter |
| | [7] | R/W | 0 | reg_pause_en_y | 0: Resume the accumulation counter<br>1: Pause the accumulation counter |
| | [6] | R/W | 0 | reg_pause_en_x | 0: Resume the accumulation counter<br>1: Pause the accumulation counter |
| | [5] | R/W | 0 | reg_dec_en_z | 0: Disable quad decoder Z<br>1: Enable quad decoder Z |
| | [4] | R/W | 0 | reg_dec_en_y | 0: Disable quad decoder Y<br>1: Enable quad decoder Y |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3] | R/W | 0 | reg_dec_en_x | 0: Disable quad decoder X<br>1: Enable quad decoder X |
| | [2] | R/W | 0 | reg_en_nfilter_z | 1: Enable de-noise filter<br>0: Only double sync phase_a_z & phase_b_z |
| | [1] | R/W | 0 | reg_en_nfilter_y | 1: Enable de-noise filter<br>0: Only double sync phase_a_y & phase_b_y |
| | [0] | R/W | 0 | reg_en_nfilter_x | 1: Enable de-noise filter<br>0: Only double sync phase_a_x & phase_b_x |
| **REG_10** | | | | | |
| 0x10 | 22 | R | 0x0 | int_z | Interrupt for Z-axis<br>1: Interrupt<br>0: No interrupt |
| | 21 | R | 0x0 | int_y | Interrupt for Y-axis<br>1: Interrupt<br>0: No interrupt |
| | 20 | R | 0x0 | int_x | Interrupt for X-axis<br>1: Interrupt<br>0: No interrupt |
| | 18 | W | 0x0 | reg_int_z_clr | 1: Clear interrupt of Z-axis |
| | 17 | W | 0x0 | reg_int_y_clr | 1: Clear interrupt of Y-axis |
| | 16 | W | 0x0 | reg_int_x_clr | 1: Clear interrupt of X-axis |
| | 14 | R/W | 0x0 | reg_int_z_mask | Mask the interrupt of Z-axis<br>0: Disable<br>1: Mask interrupt |
| | 13 | R/W | 0x0 | reg_int_y_mask | Mask the interrupt of Y-axis<br>0: Disable<br>1: Mask interrupt |
| | 12 | R/W | 0x0 | reg_int_x_mask | Mask the interrupt of X-axis<br>0: Disable<br>1: Mask interrupt |
| | [11:10] | R/W | 0x0 | - | Reserved |
| | [9:8] | R/W | 0x1 | reg_int3_sel | Define the time period to issue an interrupt repeatedly<br>0x0: Every 100ms<br>0x1: Every 200ms<br>0x2: Every 400ms<br>0x3: Every 600ms |
| | [7:4] | R/W | 0x6 | reg_int2_sel | Define the accumulation count to issue an interrupt<br>0x0: Every 1 state change samples<br>0x1: Every 2 state change samples<br>0x2: Every 4 state change samples<br>0x3: Every 8 state change samples<br>0x4: Every 16 state change samples<br>…<br>0x14: Every 16384 state change samples<br>0x15: Every 32768 state change samples |
| | [3] | R/W | 0 | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [2] | R/W | 0 | reg_int3_en | 0: Disable interrupt 3<br>1: Enable interrupt 3<br>When the time-out counter reaches a specific number, quad decoder raises an interrupt to mcu.<br>The number is decided by reg_int3_sel. |
| | [1] | R/W | 0 | reg_int2_en | 0: Disable interrupt 2<br>1: Enable interrupt 2<br>When the accumulation counter reaches a specific number, quad decoder raises an interrupt to mcu.<br>The number is decided by reg_int2_sel. |
| | [0] | R/W | 0 | reg_int1_en | 0: Disable interrupt 1<br>1: Enable interrupt 1<br>When the quad decoder gets new data and the state changes (up or down), the quad decoder raises an interrupt to mcu.<br>Priority: INT1 > INT2 > INT3 > INT4<br>e.g.: when 0x10[3:0]=0xF, HW only active interrupt 1 |
| **REG_14** | | | | | |
| 0x14 | [17] | W | - | reg_clr_ud_flag_z | 1: Clear underflow flag in Z-axis |
| | [16] | W | - | reg_clr_ud_flag_y | 1: Clear underflow flag in Y-axis |
| | [15] | W | - | reg_clr_ud_flag_x | 1: Clear underflow flag in X-axis |
| | [14] | W | - | reg_clr_pre_state_z | 1: Clear pre-state register in Z-axis |
| | [13] | W | - | reg_clr_pre_state_y | 1: Clear pre-state register in Y-axis |
| | [12] | W | - | reg_clr_pre_state_x | 1: Clear pre-state register in X-axis<br>The definition for quad decoder state is {PHASE_X,PHASE_Y}.<br>The pre-state means the previous state in quad decoder.<br>The default is 0x0 |
| | [11] | W | - | reg_clr_step2_cntr_z | 1: Clear illegal counter in Z-axis |
| | [10] | W | - | reg_clr_step2_cntr_y | 1: Clear illegal counter in Y-axis |
| | [9] | W | - | reg_clr_step2_cntr_x | 1: Clear illegal counter in X-axis |
| | [8] | W | - | reg_clr_step2_flag_z | 1: Clear illegal flag in Z-axis |
| | [7] | W | - | reg_clr_step2_flag_y | 1: Clear illegal flag in Y-axis |
| | [6] | W | - | reg_clr_step2_flag_x | 1: Clear illegal flag in X-axis |
| | [5] | W | - | reg_clr_ov_flag_z | 1: Clear overflow flag in Z-axis |
| | [4] | W | - | reg_clr_ov_flag_y | 1: Clear overflow flag in Y-axis |
| | [3] | W | - | reg_clr_ov_flag_x | 1: Clear overflow flag in X-axis |
| | [2] | W | - | reg_clr_acc_cntr_z | 1: Clear accumulation counter in Z-axis |
| | [1] | W | - | reg_clr_acc_cntr_y | 1: Clear accumulation counter in Y-axis |
| | [0] | W | - | reg_clr_acc_cntr_x | 1: Clear accumulation counter in X-axis |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_18** | | | | | |
| 0x18 | [5:4] | R | - | dbg_current_state_z | Current state of Z-axis |
| | [3:2] | R | - | dbg_current_state_y | Current state of Y-axis |
| | [1:0] | R | - | dbg_current_state_x | Current state of X-axis |
| **REG_20** | | | | | |
| 0x20 | [31:0] | R | - | - | Reserved |
| **REG_24** | | | | | |
| 0x24 | [2] | R/W | - | reg_speedup_ mode | 1: Interrupt mode 3 speed up mode |
| | [1:0] | R/W | - | reg_dbg_sel | debug signal select |
| **REG_28** | | | | | |
| 0x28 | [15:0] | R/W | 0x1F3 | reg_div_num | Divider numeral.<br>e.g.: FPGA osc clock is 25MHz.<br>The quad decoder sampling clock is 25MHz/(reg_div_num+1). |

## *10.7. SPI 2-WIRE*

**Table 18. SPI 2-WIRE (Base Address: 0x4000_4000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_30** | | | | | |
| 0x30 | [23] | R/W | 0 | reg_spi_oe_dly_en | Extend 1T of SPI_OE<br>0x0: Disable<br>0x1: Enable |
| | [22] | R/W | 1 | reg_spi_cs_en | Enable SPI_CS<br>0x0: 2-wire mode, disable SPI_CS<br>0x1: 3-wire mode, enable SPI_CS |
| | [21] | R/W | 0 | reg_spi_end_ext_en | Using this control bit can extend the timing window for SPI output enable=0. The extend time is 1/(2*SPI_CLK)<br>1: Extend mode<br>0: Normal mode |
| | [20] | R/W | 0 | reg_burst_read_en | Burst read enable signal<br>1: Enable burst read<br>0: Disable |
| | [19:16] | R/W | 0x7 | reg_burst_read_num | The total number of read data bytes in burst mode<br>0x6: Read consecutive 7 bytes<br>0x7: Read consecutive 8 bytes<br>HW changes the reg_div_num to 0x0 automatically, when reg_burst_read_en=0 (single read mode) |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [15:8] | R/W | 0x13 | reg_div_num | The divider number to generate 2x SPI_CLK<br>For the target of SPI_CLK=1MHz, FW must be set divider number to generate two times of SPI clock for internal SPI engine to operatre correctly.<br>e.g.: SPI_CLK=1MHz, SYS_CLK=40MHz, reg_div_num=(SYS_CLK/(2*SPI_CLK)) - 1 |
| | [7:3] | R/W | 0x7 | reg_dly_cycle | The delay time from the end of address phase to the start of read data phase.<br>One cycle period=1/(2 * SPI_CLK)<br>Delay time=(reg_dly_cycle + 3) * one cycle period |
| | [0] | R/W | 0 | reg_spi_en | SPI enable signal<br>1: Enable 2-wire SPI<br>0: Disable 2-wire SPI |
| **REG_34** | | | | | |
| 0x34 | [15:0] | R/W | 0 | reg_spi_data | SPI data.<br>reg_spi_data[15]: r/w mode<br>1: Write mode<br>0: Read mode<br>reg_spi_data[14:8]: address<br>reg_spi_data[ 0:7]: write data ; in read mode, this data byte has no use |
| **REG_38** | | | | | |
| 38h | [4] | R/W | 0 | reg_spi_int_mask | Interrupt mask signal<br>0x0: No mask<br>0x1: Mask interrupt |
| | [3] | W | 0 | reg_spi_rd_data_clr | Clear all read data registers<br>1: Clear (write 1 to clear) |
| | [2] | W | 0 | reg_spi_rd_num_clr | Read number clear signal<br>1: Clear (write 1 to clear) |
| | [1] | W | 0 | reg_spi_int_clr | Interrupt clear signal<br>1: Clear (write 1 to clear) |
| | [0] | R/W | 0 | reg_spi_int_en | Interrupt enable signal<br>1: Enable<br>0: Disable |
| **REG_3C** | | | | | |
| 0x3C | [2] | R | 0 | spi_int | SPI interrupt indicator<br>1: Interrupt<br>0: No interrupt |
| | [4] | R | 0 | spi_busy | SPI busy indicator<br>1: SPI busy<br>0: SPI idle |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3:0] | R | 0 | rdata_num | The total number of data byte in each SPI read transaction<br>1: One byte<br>2: Two bytes<br>…<br>8: Eight bytes<br>---<br>14: 14 bytes<br>15: 15 bytes |
| **REG_40** | | | | | |
| 0x40 | [31:24] | R | 0 | rdata_3 | Read data 3<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the forth data byte. |
| | [23:16] | R | 0 | rdata_2 | Read data 2<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the third data byte. |
| | [15:8] | R | 0 | rdata_1 | Read data 1<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the second data byte. |
| | [7:0] | R | 0 | rdata_0 | Read data 0<br>For single read mode (reg_burst_read_en=0), FW only need to get this data byte.<br>For burst read mode (reg_burst_read_en=1), this data byte is the first data byte. |
| **REG_44** | | | | | |
| 0x44 | [31:24] | R | - | rdata_7 | Read data 7<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the eighth data byte. |
| | [23:16] | R | - | rdata_6 | Read data 6<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the seventh data byte. |
| | [15:8] | R | - | rdata_5 | Read data 5<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the sixth data byte. |
| | [7:0] | R | - | rdata_4 | Read data 4<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the fifth data byte. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_48** | | | | | |
| 0x48 | [31:24] | R | - | rdata_11 | Read data 11<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the eighth data byte. |
| | [23:16] | R | - | rdata_10 | Read data 10<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the seventh data byte. |
| | [15:8] | R | - | rdata_9 | Read data 9<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the sixth data byte. |
| | [7:0] | R | - | rdata_8 | Read data 8<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the fifth data byte. |
| **REG_4c** | | | | | |
| 0x4C | [31:24] | R | - | Reserved | - |
| | [23:16] | R | - | rdata_14 | Read data 14<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the seventh data byte. |
| | [15:8] | R | - | rdata_13 | Read data 13<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the sixth data byte. |
| | [7:0] | R | - | rdata_12 | Read data 12<br>Only for burst read mode (reg_burst_read_en=1), this data byte is the fifth data byte. |

## 10.8. Hardware KeyScan

The RTL8762A supports a programmable 8x20 key matrix with keyscan engine. All GPIOs can use pin multiplexing for easy configuration. The PAD structure is pull-high resistor.

**Table 19. Hardware KeyScan (Base Address: 0x4000_6000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| 0x00 | [31] | R/W | 0x0 | reg_key_scan_clk_en | Enable internal key scan clock.<br>Key scan clock enabling must be made after key scan configuration is done.<br>0x1: Enable key scan clock<br>0x0: Disable key scan clock |
| | [27:8] | R/W | 0x0 | reg_col_sel | The control register to define which column is used<br>0x0: No key column is select<br>0x1: key column 0 is select<br>0x2: key column 1 is select<br>0x3: key column 0 and column 1 are select<br>0x4: key column 2 is select<br>…<br>0xffffe: key column 1 to column 19 are select<br>0xfffff: key column 0 to column 19 are select |
| | [7:5] | R/W | 0x0 | reg_row_num | The control register to define how many key rows to use.<br>E.g.: reg_row_num=0x0, only use one key row<br>reg_row_num=0x1, use two key rows<br>reg_row_num=0x7, use row0 ~ row7 |
| | [4:0] | R/W | 0x0 | reg_col_num | The control register to define how many key columns to use.<br>E.g.: reg_col_num=0x0, only use one key column<br>reg_col_num=0x1, use two key columns<br>reg_col_num=0x13, use column 0 ~ column 19 |
| - | | | | | |
| 0x04 | [29:27] | R/W | 0x3 | reg_col_gt_post_sel | Post guard time for key column<br>Post guard time=1/128e3 * reg_col_gt_post_sel |
| | [26:24] | R/W | 0x03 | reg_col_gt_pre_sel | Pre guard time for key column<br>Pre guard time=1/128e3 * reg_col_gt_pre_sel |
| | [23:16] | R/W | 0x0 | reg_row_sel | The control register to define which row is used<br>0x0: No key row is selected<br>0x1: key row 0 is selected<br>0x2: key row 1 is selected<br>0x3: key row 0 and row 1 are selected<br>0x4: key row 2 is selected<br>…<br>0xff: key row 0 to row 7 are selected |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [10:8] | R/W | 0x2 | reg_totime_chk_sel | When in time-out state, HW detects key activity periodically<br>0x0: 5ms<br>0x1: 10ms<br>0x2: 20ms<br>0x3: 40ms<br>0x4: 80ms<br>0x5: 160ms<br>0x6: 320ms<br>0x7: 640ms |
| | [7] | R/W | 0x0 | reg_deb_en | Enable debounce function for key row<br>0x1: Enable debounce<br>0x0: Disable debounce |
| | [6:4] | R/W | - | - | Reserved |
| | [3:2] | R/W | 0x0 | reg_totime_sel | Time-out time when no active key is detected<br>0x0: 1s<br>0x1: 2s<br>0x2: 4s<br>0x3: 8s |
| | [1:0] | R/W | 0x0 | reg_wtime_sel | Wait time during each key scan<br>0x0: 12.5ms<br>0x1: 25ms<br>0x2: 50ms<br>0x3: 100ms |
| - | | | | | |
| 0x08 | [27] | R/W | 0x0 | reg_deb_time_en | 0x0: Disable to program reg_deb_time<br>0x1: Enable to program reg_deb_time |
| | [26:16] | R/W | 0x138 | reg_deb_time | Debounce time=reg_deb_time * 31.25us<br>Debounce time can only be configured when reg_deb_time_en=1<br>Do not set debounce time to 0x0 |
| | [15] | W1C | - | fifo_clr | Clear FIFO data<br>Write 1 to clear |
| | [14] | W1C | - | force_scan_fsm_idle | Force scan FSM to idle state<br>Write 1 to clear |
| | [13] | W1C | - | fifo_flag_clr2 | Clear overflow status flag<br>Write 1 to clear |
| | [12] | W1C | - | fifo_flag_clr1 | Clear scan data reject status flag<br>Write 1 to clear |
| | [7] | R/W | 0x0 | reg_key_scan_trig_sel | Select enter key press detection mode<br>0x0: Edge trigger<br>0x1: Level trigger |
| | [6] | R/W | 0x0 | reg_fifo_ov_ctrl | FIFO overflow control<br>0x0: Discard all the new scan data when FIFO is full<br>0x1: Discard the last scan data when FIFO is full and scan_key_limit does not reach limit. When FIFO is full and scan_key_limit reaches the limit, HW cannot write any new data in this situation. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [4:0] | R/W | 0x08 | reg_scan_key_limit | Limit the maximum allowable scan data<br>0x0: No limit<br>0x1: Only one scan data is allowable in each key scan<br>…<br>0x1a: Max 26 scan data is allowable in each key scan<br>0x1b ~ 0x1f: DO NOT USE. |
| - | | | | | |
| 0x0C | [16] | R/W | 0x0 | reg_int1_option | Option of interrupt mode 1.<br>0x0: When FIFO is not empty, HW issues interrupt at the end of current key scan 0x1: When FIFO offset is not equal in the consequent key scan, HW issues interrupt at the end of current key scan |
| | [15] | R/W | 0x0 | reg_int_mask | Interrupt mask signal<br>0x1: Mask interrupt |
| | [14] | R/W | 0x0 | reg_int3_clr | Interrupt clear signal for int3<br>0x1: Clear interrupt |
| | [13] | R/W | 0x0 | reg_int2_clr | Interrupt clear signal for int2<br>0x1: Clear interrupt |
| | [12] | R/W | 0x0 | reg_int1_clr | Interrupt clear signal for int1<br>0x1: Clear interrupt |
| | [11] | R/W | 0x0 | reg_int_man | 0x0: HW control interrupt mode<br>0x1: SW control interrupt mode |
| | [10] | R/W | 0x0 | reg_int_ctrl_man | When SW control interrupt mode is enabled:<br>0x1: Enable interrupt<br>0x0: Disable interrupt |
| | [9:8] | R/W | 0x0 | reg_int_to_sel | Interrupt time-out select<br>0x0: 100ms<br>0x1: 200ms<br>0x2: 400ms<br>0x3: 800ms |
| | [6:4] | R/W | 0x2 | reg_offset_th | Interrupt threshold<br>0x0: 4 scan data<br>0x1: 8 scan data<br>0x2: 12 scan data<br>0x3: 16 scan data<br>0x4: 20 scan data<br>0x5: 24 scan data<br>Others: 16 scan data |
| | [2] | R/W | 0x0 | reg_int3_en | 0x0: Disable time-out interrupt<br>0x1: Enable time-out interrupt |
| | [1] | R/W | 0x0 | reg_int2_en | 0x0: Disable over-threshold interrupt<br>0x1: Enable over-threshold interrupt |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [0] | R/W | 0x0 | reg_int1_en | 0x0: Disable int1 interrupt<br>0x1: Enable int1 interrupt<br>When key scan engine detects valid key press, HW issues an interrupt at the end of key scan. SW needs to clear the interrupt first, and then read FIFO data.<br>The priority: int1 > int2 > int3 |
| - | | | | | |
| 0x10 | [7:0] | R | - | FIFO_DATA | [4:0] Column index<br>[7:5] Row index |
| - | | | | | |
| 0x14 | [31] | R/W | 0x0 | reg_row_man | 0x1: Enable software row control manual mode<br>0x0: Row control signal is from key matrix |
| | [30] | R/W | 0x0 | reg_col_man | 0x1: Enable software column control manual mode<br>0x0: Column control signal is from key matrix |
| | [27:20] | R/W | 0xff | reg_row_ctrl_man | Software control key row manually<br>[27:20]: Row7~row0 |
| | [19:0] | R/W | 0xfffff | reg_col_ctrl_man | Software control key row manually<br>[19:0]: Column 19~column0 |
| - | | | | | |
| 0x18 | [23:16] | R | - | row data deb | Read the status of key row (after debounce) |
| | [7:0] | R | - | row data sync | Read the status of key row (after sync) |
| - | | | | | |
| 0x1C | [12:8] | R | - | ptr_offset | FIFO offset<br>0x0: No data in FIFO<br>0x1: One data in FIFO<br>…<br>0x1A: 26 data in FIFO |
| | [7] | R | - | fifo_we_reject_st | Flag to indicate any scan data is rejected<br>0x1: Reject occurred<br>0x0: No reject occurred |
| | [6] | R | - | fifo_we_ov_st | Flag to indicate FIFO is overflow<br>0x1: Overflow occurred<br>0x0: No overflow occurred |
| | [5] | R | - | fifo_full | Flag to indicate FIFO is full<br>0x1: FIFO is full<br>0x0: FIFO is not full |
| | [4] | R | - | fifo_empty | Flag to indicate FIFO is empty<br>0x1: FIFO is empty<br>0x0: FIFO is not empty |
| | [2:0] | R | - | current_state | Status of scan state<br>0x0: IDLE<br>0x1: INIT<br>0x2: SCAN<br>0x3: SCAN_EXT<br>0x4: SCAN END<br>0x5: WAIT<br>0x6: TIME-OUT |

# 10.9. AUXADC

There are two different operation modes of AUXADC in the RTL8762A. In AUX mode, AUXADC provides eight external channels and two internal channels with 12-bit sigma-delta ADC. In audio mode, the RTL8762A utilizes a built-in Programmable Gain Amplifier (PGA), digital DC removal, and microphone bias. The RTL8762A natively supports analog and digital microphones. Additionally, the RTL8762A embeds an MSBC encoder. The MIC input data path is shown below. In audio mode, the MIC signal is amplified by PGA and converted to a digital signal by sigma-delta ADC and the audio codec. The audio codec output is 16kHz, 16bits audio signal format. The PGA gain provides 8 configuration levels; 0dB/14dB/20dB/24dB/30dB/35dB/40dB/44dB/50dB.

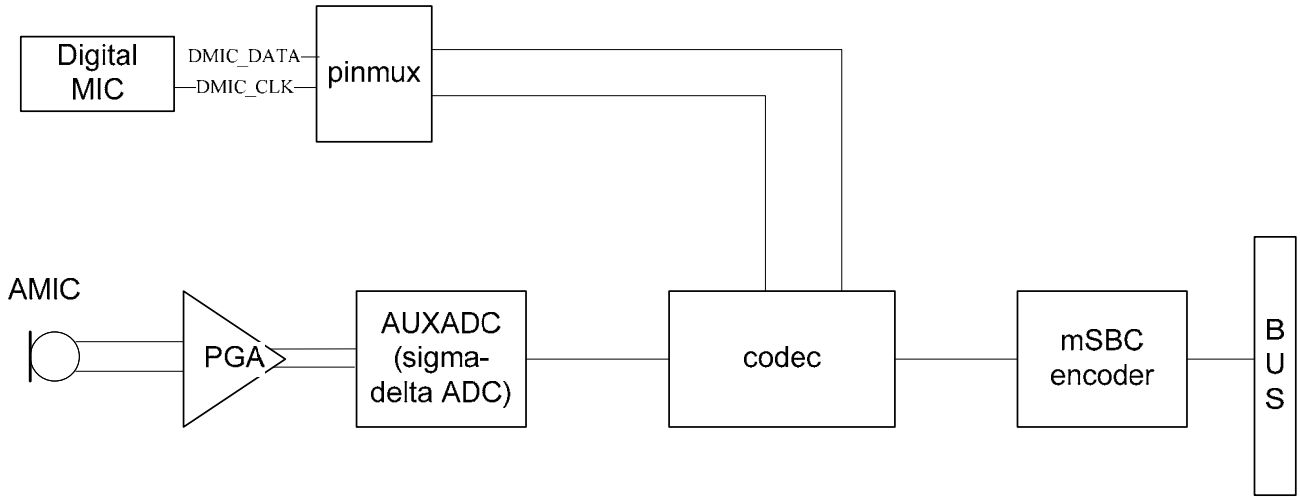


**Figure 14. MIC Input Data Path in Audio Mode**

**Table 20. AUXADC (Base Address: 0x4001_0000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **ADC FIFO Read Register** | | | | | |
| 0x00 | [31:0] | R | 32'b0 | - | ADC FIFO read only register |
| **ADC Control Register** | | | | | |
| 0x04 | [31:29] | NA | 5'b0 | - | Reserved |
| | [28] | W1C | 1'b0 | reg_adc_fifo_clr | 0x1: Clear FIFO data and FIFO pointer |
| | [27] | R/W | 1'b0 | reg_adc_temp_data_sel | 0x1: Temperature data from adc_data_out<br>0x0: Temperature data from adc_ct_out |
| | [26:24] | R/W | 3'b0 | reg_adc_dbg_sel | ADC debug signal select |
| | [23:22] | NA | 2'b0 | - | Reserved |
| | [21:16] | R/W | 6'b0 | reg_adc_threshold | ADC FIFO threshold.<br>Mainly used in one shot mode |
| | [15:14] | NA | 2'b0 | - | Reserved |
| | [13:8] | R/W | 6'd26 | reg_adc_burst_size | Burst size to trigger DMA request when FIFO is enabled |
| | [7:5] | NA | 4'b0 | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [4] | R/W | 1'b0 | reg_adc_endian | Endian selection<br>Do not use in audio mode |
| | [3] | R/W | 1'b0 | reg_adc_overwrite | Enable FIFO overwrite mode when FIFO is full |
| | [2] | R/W | 1'b0 | reg_adc_comp_only | Enable compare mode only without FIFO enable |
| | [1] | R/W | 1'b0 | reg_adc_oneshot | 0x1: Enable ADC one shot mode<br>0x0: Disable ADC one shot mode<br>FW needs to manually disable one shot mode after HW issues a one shot done interrupt |
| | [0] | R/W | 1'b0 | reg_adc_auto | 0x1: Enable ADC auto sensor mode<br>0x0: Disable ADC auto sensor ode<br>When FW enable reg_adc_auto, HW turns on all aux_adc power automatically. |
| **ADC Interrupt Enable Register** | | | | | |
| 0x08 | [31] | W1C | 1'b0 | clr_adc_int_st_15 | Interrupt clear of codec_en/msbc_en=1 interrupt |
| | [30] | W1C | 1'b0 | clr_adc_int_st_14 | Interrupt clear of FIFO threshold interrupt |
| | [29] | W1C | 1'b0 | clr_adc_int_st_13 | Interrupt clear of FIFO read error interrupt |
| | [28] | W1C | 1'b0 | clr_adc_int_st_12 | Interrupt clear of FIFO read request interrupt |
| | [27] | W1C | 1'b0 | clr_adc_int_st_11 | Interrupt clear of FIFO full interrupt |
| | [26] | W1C | 1'b0 | clr_adc_int_st_10 | Interrupt clear of one shot done interrupt |
| | [25] | W1C | 1'b0 | clr_adc_int_st_9 | Interrupt clear of temp. comparator interrupt |
| | [24] | W1C | 1'b0 | clr_adc_int_st_8 | Interrupt clear of vbat comparator interrupt |
| | [23] | W1C | 1'b0 | clr_adc_int_st_7 | Interrupt clear of ch7 comparator interrupt |
| | [22] | W1C | 1'b0 | clr_adc_int_st_6 | Interrupt clear of ch6 comparator interrupt |
| | [21] | W1C | 1'b0 | clr_adc_int_st_5 | Interrupt clear of ch5 comparator interrupt |
| | [20] | W1C | 1'b0 | clr_adc_int_st_4 | Interrupt clear of ch4 comparator interrupt |
| | [19] | W1C | 1'b0 | clr_adc_int_st_3 | Interrupt clear of ch3 comparator interrupt |
| | [18] | W1C | 1'b0 | clr_adc_int_st_2 | Interrupt clear of ch2 comparator interrupt |
| | [17] | W1C | 1'b0 | clr_adc_int_st_1 | Interrupt clear of ch1 comparator interrupt |
| | [16] | W1C | 1'b0 | clr_adc_int_st_0 | Interrupt clear of ch0 comparator interrupt |
| | [15] | R/W | 1'b0 | reg_adc_codec_hw_en | ADC codec_en/msbc_en=1 interrupt enable<br>0x1: Enable |
| | [14] | R/W | 1'b0 | reg_adc_fifo_th_en | ADC FIFO threshold interrupt enable<br>0x1: Enable |
| | [13] | R/W | 1'b0 | reg_adc_fifo_rd_error_en | ADC FIFO read error interrupt enable<br>0x1: Enable |
| | [12] | R/W | 1'b0 | reg_adc_fifo_rd_req_en | ADC FIFO read request interrupt enable<br>0x1: Enable |
| | [11] | R/W | 1'b0 | reg_adc_fifo_full_en | ADC FIFO full interrupt enable<br>0x1: Enable |
| | [10] | R/W | 1'b0 | reg_adc_one_shot_done_en | ADC one shot done interrupt enable<br>0x1: Enable |
| | [9] | R/W | 1'b0 | reg_adc_comp_temp_en | ADC temperature comparator interrupt enable<br>0x1: Enable |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [8] | R/W | 1'b0 | reg_adc_comp_vbat_en | ADC vbat comparator interrupt enable<br>0x1: Enable |
| | [7] | R/W | 1'b0 | reg_adc_comp_7_en | ADC ch7 comparator interrupt enable<br>0x1: Enable |
| | [6] | R/W | 1'b0 | reg_adc_comp_6_en | ADC ch6 comparator interrupt enable<br>0x1: Enable |
| | [5] | R/W | 1'b0 | reg_adc_comp_5_en | ADC ch5 comparator interrupt enable<br>0x1: Enable |
| | [4] | R/W | 1'b0 | reg_adc_comp_4_en | ADC ch4 comparator interrupt enable<br>0x1: Enable |
| | [3] | R/W | 1'b0 | reg_adc_comp_3_en | ADC ch3 comparator interrupt enable<br>0x1: Enable |
| | [2] | R/W | 1'b0 | reg_adc_comp_2_en | ADC ch2 comparator interrupt enable<br>0x1: Enable |
| | [1] | R/W | 1'b0 | reg_adc_comp_1_en | ADC ch1 comparator interrupt enable<br>0x1: Enable |
| | [0] | R/W | 1'b0 | reg_adc_comp_0_en | ADC ch0 comparator interrupt enable<br>0x1: Enable |
| **ADC Interrupt Status Register** | | | | | |
| 0x0C | [31:16] | R | 1'b0 | - | Reserved |
| | [15] | R | 1'b0 | reg_adc_codec_hw_en_st | codec_en/msbc_en=1 interrupt status |
| | [14] | R | 1'b0 | reg_adc_fifo_threshold_st | FIFO over threshold interrupt status |
| | [13] | R | 1'b0 | reg_adc_fifo_rd_error_st | ADC FIFO read error interrupt status |
| | [12] | R | 1'b0 | reg_adc_fifo_rd_req_st | ADC FIFO read request interrupt status |
| | [11] | R | 1'b0 | reg_adc_fifo_full_st | ADC FIFO full interrupt status |
| | [10] | R | 1'b0 | reg_adc_one_shot_done_st | One shot done status |
| | [9] | R | 1'b0 | reg_adc_comp_temp_st | Temp. compare status |
| | [8] | R | 1'b0 | reg_adc_comp_vbat_st | vbat compare status |
| | [7] | R | 1'b0 | reg_adc_comp_7_st | Channel 7 compare status |
| | [6] | R | 1'b0 | reg_adc_comp_6_st | Channel 6 compare status |
| | [5] | R | 1'b0 | reg_adc_comp_5_st | Channel 5 compare status |
| | [4] | R | 1'b0 | reg_adc_comp_4_st | Channel 4 compare status |
| | [3] | R | 1'b0 | reg_adc_comp_3_st | Channel 3 compare status |
| | [2] | R | 1'b0 | reg_adc_comp_2_st | Channel 2 compare status |
| | [1] | R | 1'b0 | reg_adc_comp_1_st | Channel 1 compare status |
| | [0] | R | 1'b0 | reg_adc_comp_0_st | Channel 0 compare status |
| **ADC Compare Value Register** | | | | | |
| 0x10 | [31:16] | R/W | 16'b0 | reg_adc_comp_th_1 | ADC compare threshold of channel 1 |
| | [15:0] | R/W | 16'b0 | reg_adc_comp_th_0 | ADC compare threshold of channel 0 |
| **ADC Compare Value Register** | | | | | |
| 0x14 | [31:16] | R/W | 16'b0 | reg_adc_comp_th_3 | ADC compare threshold of channel 3 |
| | [15:0] | R/W | 16'b0 | reg_adc_comp_th_2 | ADC compare threshold of channel 2 |
| **ADC Compare Value Register** | | | | | |
| 0x18 | [31:10] | R | 22'b0 | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [9] | R/W | 1'b0 | reg_adc_greater_than_9 | 0x1: Temp. ADC data > reg_adc_comp_th_temp<br>0x0: Temp. ADC data <= reg_adc_comp_th_temp |
| | [8] | R/W | 1'b0 | reg_adc_greater_than_8 | 0x1: vbat ADC data > reg_adc_comp_th_vbat<br>0x0: vbat ADC data <= reg_adc_comp_th_vbat |
| | [7] | R/W | 1'b0 | reg_adc_greater_than_7 | 0x1: CH7 ADC data > reg_adc_comp_th_7<br>0x0: CH7 ADC data <= reg_adc_comp_th_7 |
| | [6] | R/W | 1'b0 | reg_adc_greater_than_6 | 0x1: CH6 ADC data > reg_adc_comp_th_6<br>0x0: CH6 ADC data <= reg_adc_comp_th_6 |
| | [5] | R/W | 1'b0 | reg_adc_greater_than_5 | 0x1: CH5 ADC data > reg_adc_comp_th_5<br>0x0: CH5 ADC data <= reg_adc_comp_th_5 |
| | [4] | R/W | 1'b0 | reg_adc_greater_than_4 | 0x1: CH4 ADC data > reg_adc_comp_th_4<br>0x0: CH4 ADC data <= reg_adc_comp_th_4 |
| | [3] | R/W | 1'b0 | reg_adc_greater_than_3 | 0x1: CH3 ADC data > reg_adc_comp_th_3<br>0x0: CH3 ADC data <= reg_adc_comp_th_3 |
| | [2] | R/W | 1'b0 | reg_adc_greater_than_2 | 0x1: CH2 ADC data > reg_adc_comp_th_2<br>0x0: CH2 ADC data <= reg_adc_comp_th_2 |
| | [1] | R/W | 1'b0 | reg_adc_greater_than_1 | 0x1: CH1 ADC data > reg_adc_comp_th_1<br>0x0: CH1 ADC data <= reg_adc_comp_th_1 |
| | [0] | R/W | 1'b0 | reg_adc_greater_than_0 | 0x1: CH0 ADC data > reg_adc_comp_th_0<br>0x0: CH0 ADC data <= reg_adc_comp_th_0 |
| **ADC Power** | | | | | |
| 0x1C | [31] | R | 1'b0 | - | Reserved |
| | [30] | R | 1'b0 | adc_cic_rst_n | the status of ADC CIC filter rst_n (low active) |
| | [29] | R | 1'b0 | adc_pwr18_on | the status of ADC 1.8V ON |
| | [28] | R/W | 1'b0 | reg_adc_amic_mode_en | 0x1: Enable ADC analog MIC mode<br>0x0: Disable ADC analog MIC mode |
| | [27] | R/W | 1'b0 | reg_adc_oneshot_pwr_off | 0x1: Force ADC power off in one shot mode when reg_adc_oneshot_force_on=1. FW need to write reg_adc_oneshot_pwr_off=0 manually. |
| | [26] | R/W | 1'b0 | reg_adc_oneshot_mode | 0x1: ADC one shot mode<br>0x0: ADC auto mode |
| | [25] | R/W | 1'b0 | reg_adc_oneshot_force_on | 0x1: Force ADC power always on in one shot mode. Use reg_adc_oneshot_pwr_off=1 to force ADC power off.<br>0x0: ADC power is on only after FW trigger reg_adc_oneshot=1. When one shot is done, HW turn ADC off automatically |
| | [24:22] | R/W | 3'b0 | reg_adc_pwr_cut_cntr3 | Power cut counter3 based on 1µs pulse (Not used; no function) |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [21:19] | R/W | 3'b0 | reg_adc_pwr_cut_cntr2 | Power cut counter2 based on 1µs pulse<br>0x0: 11µs<br>0x1: 12µs<br>0x2: 13µs<br>0x3: 14µs<br>0x4: 18µs<br>0x5: 34µs<br>0x6: 66µs<br>0x7: 130µs |
| | [18:16] | R/W | 3'b0 | reg_adc_pwr_cut_cntr1 | Power cut counter1 based on 1µs pulse<br>0x0: 1.x µs<br>0x1: 2.x µs<br>0x2: 3.x µs<br>0x3: 7.x µs<br>0x4: 15.x µs<br>0x5: 31.x µs<br>0x6: 63.x µs<br>0x7: 127.x µs |
| | [15:12] | R | 4'b0 | adc_pwr_state | ADC power status |
| | [11] | R | 1'b0 | adc_fifo_en | Status of ADC FIFO enable |
| | [10] | R | 1'b0 | adc_en_hw | Status of ADC hw enable |
| | [9] | R | 1'b0 | adc_ldo_24_on | Status of ANAPAR_LDO[24] ON |
| | [8] | R | 1'b0 | adc_ldo_20_on | Status of ANAPAR_LDO[20] ON |
| | [7] | R | 1'b0 | adc_ldo_16_on | Status of ANAPAR_LDO[16] ON |
| | [6] | R | 1'b0 | adc_pwr33_on | Status of ADC 3.3V ON |
| | [5] | R | 1'b0 | adc_pwr12_on | Status of ADC 1.2V ON |
| | [4] | R/W | 1'b1 | reg_adc_iso_manual | 0x1: Enable ISO cell manually (Not used; no function) |
| | [3] | R/W | 1'b0 | reg_adc_pwr18_manual | 0x1: Enable 1.8V power manually. |
| | [2] | R/W | 1'b0 | reg_adc_pwr33_manual | 0x1: Enable 3.3V power manually. |
| | [1] | R/W | 1'b0 | reg_adc_pwr12_manual | 0x1: Enable 1.2V power manually. |
| | [0] | R/W | 1'b0 | reg_adc_pwr_man | 0x1: Enable ADC power manually<br>0x0: Enable ADC power automatically by reg_adc_auto or reg_adc_oneshot |
| **ADC anapar_ad0** | | | | | |
| 0x20 | [31:2] | R/W | - | - | Reserved |
| | [3] | R/W | - | reg_adc_audio_en | Enable audio mode. |
| | [1:2] | R/W | - | - | Reserved |
| | [0] | R/W | - | reg_adc_en_manual | Enable ADC manually.<br>Set after reg_adc_iso_manual.<br>Ignored when reg_adc_pwr_man set. |
| **ADC Calibration Data** | | | | | |
| 0x40 | [15:0] | R/W | 16'b0 | reg_caliV | ADC DC calibration V data |
| | [31:16] | R/W | 16'b0 | reg_caliG | ADC DC calibration G data |
| **ADC Compare Value Register** | | | | | |
| 0x44 | [15:0] | R/W | 16'b0 | reg_adc_comp_th_4 | ADC compare threshold of channel 4 |
| | [31:16] | R/W | 16'b0 | reg_adc_comp_th_5 | ADC compare threshold of channel 5 |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **ADC Compare Value Register** | | | | | |
| 0x48 | [15:0] | R/W | 16'b0 | reg_adc_comp_th_6 | ADC compare threshold of channel 6 |
| | [31:16] | R/W | 16'b0 | reg_adc_comp_th_7 | ADC compare threshold of channel 7 |
| **ADC** | | | | | |
| 0x4C | [31:0] | R/W | 32'b0 | reg_anapar_rsvd | ADC reserved register |
| **REG_50** | | | | | |
| 0x50 | [0] | R/W | 1 | reg_audio_ip_rst_n | Audio CODEC IP reset signal<br>1: Disable reset<br>0: Enable reset; need to keep 6T at least |
| **REG_54** | | | | | |
| 0x54 | [11] | R/W | 1 | reg_audio_clk_man_mode | reg_ad_clk_en and reg_ad_ana_clk_en manual mode control<br>0x1: Manual mode control<br>0x0: HW control |
| | [10:9] | R/W | 1 | reg_dmic_clk_sel | DMIC clock<br>0x3: 625KHz<br>0x2: 1.25MHz<br>0x1: 2.5MHz<br>0x0: 5MHz |
| | [8] | R/W | 0 | reg_dmic_clk_en | 1: Enable dmic clk<br>0: Disable dmic clk |
| | [7:4] | R/W | 5 | reg_adc_sample_rate | AMIC sampling rate (Not used; no function)<br>0x8: 44.1KHz<br>0x7: 8KHz<br>0x5: 16KHz<br>0x3: 32KHz<br>0x0: 48KHz<br>Others: Reserved; do not use. |
| | [1] | R/W | 0 | reg_ad_clk_en | 1: Enable ad_clk<br>0: Disable ad_clk |
| | [0] | R/W | 0 | reg_ad_ana_clk_en | 1: Enable ana_clk<br>0: Disable ana_clk |
| **REG_58** | | | | | |
| 0x58 | [11:10] | R/W | 0 | reg_ad_zdet_tout | ADC zero detection time out select<br>0x3: 64 samples<br>0x2: 1024*64 samples<br>0x1: 1024*32 samples<br>0x0: 1024*16 samples |
| | [9:8] | R/W | 2 | reg_ad_zdet_func | ADC zero detection function select<br>0x3: No change<br>0x2: Zero detection & increase/decrease change<br>0x1: Zero detection & immediate change<br>0x0: Immediate change |
| | [6:5] | R/W | 0 | reg_ad_comp_gain | ADC compensate gain<br>0x3: 3dB<br>0x2: 2dB<br>0x1: 1dB<br>0x0: 0dB |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [4] | R/W | 1 | reg_ad_dchpf_en | High pass filter enable control (filter DC)<br>1: Enable<br>0: Disable |
| | [3] | R/W | 1 | reg_ad_mix_mute | Analog ADC input path mute control Left Channel<br>1: Mute<br>0: Un-Mute |
| | [2:1] | R/W | 0 | reg_ad_lpf1st_fc_sel | DMIC SRC LPF fc<br>0x0: 31.04KHz<br>0x1: 46.92KHz<br>0x2: 63.06KHz,<br>0x3: 79.45KHz |
| | [0] | R/W | 0 | reg_ad_test_en | Test mode for AMIC input path select<br>1: From DMIC data<br>0: From AMIC data |
| **REG_5C** | | | | | |
| 0x5C | [13:8] | R/W | 0 | reg_adj_hpf_coef_num | Coefficient fine select (0~63) |
| | [7:5] | R/W | 0 | reg_adj_hpf_coef_sel | Coefficient coarse select, fc range (num==0 ~ num==63)<br>0x2: fs=48k or 44.1k corresponding fc=(30~2168Hz) or (28~1992Hz)<br>0x1: fs=32k corresponding fc=(40~3278Hz)<br>0x0: fs=8k, or 16k corresponding fc=(20~2000Hz), or (40~4000Hz) |
| | [4] | R/W | 1 | reg_adj_hpf_2nd_en | Adaptive 2nd High pass filter enable control<br>0: Disable 1: Enable |
| | [3] | R/W | 1 | reg_dmic_mix_mute | DMIC input path mute control Left Channel<br>0'b: Un-Mute 1'b: Mute |
| | [2:1] | R/W | 0 | reg_dmic_lpf1st_fc_sel | DMIC SRC LPF fc<br>00b: 31.04KHz, 01b: 46.92KHz,<br>10b: 63.06KHz, 11b: 79.45KHz |
| | [0] | R/W | 0 | reg_dmic_ri_fa_sel | DMIC Data Latching Control<br>0: Rising latch 1: Falling latch |
| **REG_60** | | | | | |
| 0x60 | [9:8] | R/W | 0 | reg_dmic_boost_gain | DMIC boost gain control<br>0x3: 36dB<br>0x2: 24dB<br>0x1: 12dB<br>0x0: 0dB |
| | [7] | R/W | 1 | reg_ad_mute | Digital Mute From ADC Left Channel Digital Mixer<br>1: Mute<br>0: Un-Mute |
| | [6:0] | R/W | 0x2f | reg_ad_gain | ADC digital volume (-17.625dB ~ +30dB in 0.375 dB step)<br>7'h00: -17.625dB<br>…<br>7'h2f: 0dB<br>7'h30: 0.375dB<br>…<br>7'h7f: 30dB |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **REG_64** | | | | | |
| 0x64 | [7:6] | R/W | 0 | reg_mic_stable_time_sel | MIC bias stable time select<br>0x0: 2μs<br>0x1: 4μs<br>0x2: 8μs<br>0x3: 12μs |
| | [5] | R/W | 0 | reg_msbc_en_man_mode | 0x1: msbc encode manually enable mode |
| | [4] | R/W | 0 | reg_dmic_fifo_en | 0x1: Force ADC FIFO enable |
| | [3] | R/W | 0 | reg_byp_msbc_enc_mode | 0x1: Bypass msbc encode |
| | [2:1] | R/W | 0 | reg_option_sb_data_trnc | msbc encoder output scale control<br>0x0: x 1<br>0x1: x 1/2<br>0x2: x 4<br>0x3: x 2 |
| | [0] | R/W | 0 | reg_msbc_enc_en | msbc encoder enable<br>1: Enable<br>0: Disable |
| **REG_6C** | | | | | |
| 0x6C | [31] | R | 0 | mic_ovcd_flag | 0x1: MIC bias is over current |
| | [8] | R/W | 0 | reg_byp_reg6c | 0x1: Bypass 0x6c[7:0] setting<br>0x0: When using HW mode to enable CODEC and msbc_enc, HW writes 0x6C[7:0] to CODEC automatically |
| | [7] | R/W | 1 | reg_ad_mute | Digital Mute From ADC Left Channel Digital Mixer<br>1: Mute<br>0: Un-Mute |
| | [6:0] | R/W | 0x2f | reg_ad_gain | ADC digital volume (-17.625dB ~ +30dB in 0.375 dB step)<br>7'h00: -17.625dB<br>…<br>7'h2f: 0dB<br>7'h30: 0.375dB<br>…<br>7'h7f: 30dB |
| **REG_70** | | | | | |
| 0x70 | [1] | R/W | 0 | reg_fpga_codec_pat_start | [FPGA only] enable apb_adc internal pattern.<br>1: Starts internal pattern generator and counts from 0~65535 and wraps around<br>0: Stops internal pattern generator |
| | [0] | R/W | 0 | reg_fpga_codec_test_en | [FPGA only] select adc FIFO control path<br>1: Data from apb_adc internal pattern generator<br>0: Normal mode; data from codec |
| **ADC Compare Value Register** | | | | | |
| 0x68 | [15:0] | R/W | 16'b0 | reg_adc_comp_th_vbat | ADC compare threshold of vbat |
| | [31:16] | R/W | 16'b0 | reg_adc_comp_th_temp | ADC compare threshold of temp. |
| **ADC temp. Data** | | | | | |
| 0x78 | [15:0] | R/W | 16'b0 | adc_temp_data | ADC temp. data |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **ADC vbat Data** | | | | | |
| 0x7C | [15:0] | R/W | 16'b0 | adc_vbat_data | ADC vbat data |
| **ADC ch1 Data** | | | | | |
| 0x80 | [15:0] | R/W | 16'b0 | adc_ch1_data | ADC ch1 data in one shot mode |
| **ADC ch1 Data** | | | | | |
| 0x84 | [15:0] | R/W | 16'b0 | adc_ch2_data | ADC ch2 data in one shot mode |
| **ADC ch1 Data** | | | | | |
| 0x88 | [15:0] | R/W | 16'b0 | adc_ch3_data | ADC ch3 data in one shot mode |
| **ADC ch1 Data** | | | | | |
| 0x8C | [15:0] | R/W | 16'b0 | adc_ch4_data | ADC ch4 data in one shot mode |
| **ADC ch1 Data** | | | | | |
| 0x90 | [15:0] | R/W | 16'b0 | adc_ch5_data | ADC ch5 data in one shot mode |
| **ADC ch1 Data** | | | | | |
| 0x94 | [15:0] | R/W | 16'b0 | adc_ch6_data | ADC ch6 data in one shot mode |
| **ADC ch1 Data** | | | | | |
| 0x98 | [15:0] | R/W | 16'b0 | adc_ch7_data | ADC ch7 data in one shot mode |
| **ADC ch1 Data** | | | | | |
| 0x9C | [15:0] | R/W | 16'b0 | adc_ch8_data | ADC ch8 data in one shot mode |

# 10.10. Data UART

Table 21. Data UART (Base Address: 0x4004_0000)

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **Divisor Latch (LS) Register** | | | | | |
| 0x00 | [31:8] | R/W | - | - | Reserved |
| | [7:0] | R/W | 0 | DLL | Divisor [7:0]; accessible when DLAB=1 |
| **Divisor Latch (MS) Register** | | | | | |
| 0x01 | [31:8] | R/W | - | - | Reserved |
| | [7:0] | R/W | 0 | DLM | Divisor [15:8]; accessible when DLAB=1 |
| **Interrupt Enable Register** | | | | | |
| 0x04 | [31:8] | R/W | - | - | Reserved |
| | [7:4] | R/W | 0 | - | Reserved<br>Should be logic 1'b0 |
| | [3] | R/W | - | edssi | Enable Modem Status Interrupt (EDSSI) (modem status transition)<br>0: Disabled<br>1: Enabled |
| | [2] | R/W | 0 | elsi | Enable Receiver Line Status Interrupt (ELSI) (receiver line status)<br>0: Disabled<br>1: Enabled |
| | [1] | R/W | - | etbei | Enable Transmitter FIFO Empty interrupt (ETBEI) (tx FIFO empty)<br>0: Disabled<br>1: Enabled |
| | [0] | R/W | 0 | erbi | Enable Received Data Available Interrupt (ERBFI) (rx trigger or timeout)<br>0: Disabled<br>1: Enabled |
| **Interrupt Identification Register** | | | | | |
| 0x08 | [31:8] | R | - | - | Reserved |
| | [7] | R | 0 | fifo_en | fifo_en field of FCR bit[0] |
| | [6] | R | - | fifo_en | fifo_en field of FCR bit[0] |
| | [5:4] | R | 0 | RSVD | Enable Receiver Line Status Interrupt (ELSI) (receiver line status)<br>0: Disabled<br>1: Enabled |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3:1] | R | - | int_id[2:0] | Bit1 and Bit2, the two bits of the IIR, are used to identify the highest priority interrupt pending as indicated in the following table.<br>Bit3: In FIFO mode this bit is set along with bit 2 when a timeout interrupt is pending<br>Bit3~Bit1 displays the list of possible interrupts along with the bits they enable, priority, and their source and reset control.<br>3'b011: Interrupt Priority: 1st priority (int_3)<br>Interrupt Type: Receiver Line Status (read lsr)<br>Interrupt Source: Parity, overrun, Framing errors, or break interrupt<br>Interrupt Reset Control: Reading the Line Status Register<br>3'b010: Interrupt Priority: 2nd priority<br>Interrupt Type: Receiver Data Available or trigger level reached.(int_2)<br>Interrupt source: FIFO Trigger level reached or rxfifo full Interrupt Reset Control: FIFO drops below trigger level (depends on FCR[7:6]: rcvr_trig 00: 1 01: 4 10: 8 11: 14) 3'b110: Interrupt Priority: 2nd priority<br>Interrupt Type: Timeout Indication<br>Interrupt source: There is at least 1 character in the FIFO but no character has been input to the FIFO or read from it for the last 4 character times.<br>Interrupt Reset Control: Reading Receiver Buffer Register (RBR)<br>'3'b001: Interrupt Priority: 3rd priority<br>Interrupt Type: Transmitter holding register TXFIFO empty<br>Interrupt source: Transmitter holding register TXFIFO empty<br>Interrupt Reset Control: Writing to the TXFIFO (THR)Transmitter Holding Register or reading IIR (if source of interrupt)<br>3'b000: Interrupt Priority: 4th priority<br>Interrupt Type: Modem Status<br>Interrupt source: CTS, DSR, RI, or DCD (input relative signal)<br>Interrupt Reset Control: Reading the Modem status register |
| | [0] | R | 0 | int_pend | Indicates that an interrupt is pending when it is logic '0'. When it is '1', no interrupt is pending<br>0: An interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine<br>1: No interrupt is pending |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **FIFO Control Register** | | | | | |
| 0x08 | [7:6] | R/W | - | rxfifo_trigger_level | Defines the 16-entry Receiver FIFO Interrupt trigger level 0~15 bytes<br>00: 1 byte<br>01: 4 bytes<br>10: 8 bytes<br>11: 14 bytes |
| | [5:4] | R/W | - | - | Reserved |
| | [3] | W | - | dma_mode | Supports DMA mode. |
| | [2] | W | - | clear_txfifo | Writing logic '1' to Bit 2 clears the Transmitter FIFO and resets its logic.<br>The shift register is not cleared, i.e., transmitting of the current character continues.<br>The 1 that is written to this bit position is self-clearing |
| | [1] | W | - | clear_rxfifo | Writing logic '1' to Bit 1 clears the Receiver FIFO and resets its logic, however it does not clear the shift register, i.e. receiving of the current character continues.<br>The 1 that is written to this bit position is self-clearing |
| | [0] | W | - | fifo_en | FIFO enable<br>Set to 1 for only enabling the report of Error in RCVR FIFO field in LSR bit [7],<br>(in our application this is FIFO mode only) |
| **Line Control Register** | | | | | |
| 0x0C | [31:8] | R/W | - | - | Reserved |
| | [7] | R/W | 0 | dlab | Divisor Latch Access bit<br>0: The divisor latches cannot be accessed<br>1: The divisor latches can be accessed<br>*Note: DLL/DLM can only be accessed when dlab bit=1*<br>*IER can only be accessed when dlab bit=0*<br>*THR/RBR does not care about dlab bit value.* |
| | [6] | R/W | 0 | break_ctrl | Break Control bit<br>0: Break is disabled<br>1: The serial out is forced into logic '0' (break state). Break control bit causes a break condition to be transmitted to the receiving UART.<br>When it is set to a logic 1, the serial output (Sout) is forced to the Spacing (logic 0) state. The break is disabled by setting bit 6 to logic 0. The Break Control bit acts only on SOUT and has no effect on the transmitter logic. |
| | [5] | R/W | 0 | stick_parity | Stick Parity bit.<br>1: Stick Parity bit at 1'b1<br>0: Stick Parity bit at 1'b0 |
| | [4] | R/W | - | even_parity_sel | Even Parity select.<br>0: Odd number of logic '1' is transmitted and checked in each word (data and parity combined). In other words, if the data has an even number of '1's in it, then the parity bit is '1'<br>1: Even number of '1's is transmitted in each word. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3] | R/W | 0 | parity_en | Parity Enable<br>0: No parity<br>1: Parity bit is generated on each outgoing character and is checked on each incoming one. |
| | [2] | R/W | 0 | stb | This bit specifies the number of Stop bits transmitted and received in each serial character<br>0: No stop bits<br>1: 1 stop bits.<br>Note that the receiver always checks the first stop bit only. |
| | [1] | R/W | - | Reserved | Even Parity select<br>0: Odd number of Logic '1' is transmitted and checked in each word (data and parity combined). In other words, if the data has an even number of '1' in it, then the parity bit is '1'<br>1: Even number of '1's is transmitted in each word. |
| | [0] | R/W | 1 | 0 wls0 | Word length selection 0, 0: data is 7-bit word length.<br>1: Data is 8-bit word length. |
| **Modem Control Register** | | | | | |
| 0x10 | [31:8] | - | - | - | Reserved |
| | [7:6] | R/W | 0 | - | Reserved |
| | [5] | R/W | 0 | autoflow_en | AutoFlow Enable (AFE)<br>This Bit (AFE) is the auto flow control enable. When set, the autoflow control as described in the detailed description is enabled.<br>In diagnostic mode, data that is transmitted is immediately received. This allows the CPU to verify the transmit and receive data paths. The receiver and transmitter interrupts are fully operational. The modem control interrupts are also operational, but the modem control interrupt's sources are now the lower four bits of the MCR instead of four modem control inputs. All interrupts are still controlled by the IER<br>{AFE, RTS}:<br>2'b11: Auto-RTS_ and auto-CTS_ enabled (autoflow control enabled)<br>2'b10: Auto-CTS_ only enabled.<br>2'b0X: Auto-RTS_ and auto-CTS_ disabled<br>*Note: Autoflow control is comprised of auto-CTS and auto-RTS . With auto-CTS , the CTS input must be active before the transmitter FIFO can emit data. With auto-RTS_, RTS_ becomes active when the receiver needs more data and notifies the sending serial device. When RTS_ is connected to CTS_, data transmission does not occur unless the receiver FIFO has space for the data; thus, overrun errors are eliminated with the autoflow control enabled. If not, overrun errors occurs when the transmit data rate exceeds the receiver FIFO read latency.* |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | | | | | *Note: Auto-RTS_*<br>*Auto-RTS_ data flow control originates in the receiver timing and control block and is linked to the programmed receiver FIFO trigger level. When the receiver FIFO level reaches a trigger level, RTS_ is de-asserted. The sending communication element may send an additional byte after the trigger level is reached (assuming the sending communication element has another byte to send) because it may not recognize the de-assertion of RTS_ until after it has begun sending the additional byte. RTS_ is automatically reasserted once the Receive FIFO is emptied.*<br>*Note: Auto-CTS_: The Transmitter circuitry checks CTS_ before sending the next data byte. When CTS_ is active, it sends the next byte. Top stops the transmitter from sending the following byte, CTS must be released before the middle of the last stop bit that is currently being sent. The auto-CTS function reduces interrupts to the host system. When flow control is enabled, CTS level changes do not trigger host interrupts because of the device automatically controls its own transmitter. Without auto-CTS , the transmitter sends any data present in the transmitter FIFO and a receiver overrun error may result.* |
| | [4] | R/W | 0 | loopback_en | LoopBack mode<br>0: Normal operation<br>1: Loopback mode<br>This bit provides a local loopback feature for diagnostic testing of the UART. When bit 4 is set to logic 1, the following occurs: the transmitter Serial Output (SOUT) is set to the Marking (logic 1) state; the receiver Serial Input (SIN) is disconnected; the output of the Transmitter Shift Register is 'looped back' into the Receiver Shift Register input; the four MODEM control inputs (DSR_, CTS_, RI_ and DCD_) are disconnected; the four MODEM Control output (DTR_, RTS_, OUT1_, and OUT2_) are internally connected to the four MODEM Control inputs, and the MODEM Control output pins are forced to their inactive state (high).<br>In loopback mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit-and-received data paths of the UART.<br>In loopback mode, the receiver and transmitter interrupts are fully operational. Their sources are external to the part. The MODEM Control interrupts are also operational, but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the four MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register. |
| | - | - | - | - | - |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3] | R/W | 0 | out2 | Out2. This bit controls the output 2 (OUT2_) signal, which is an auxiliary user-designated output. Bit3 affects the OUT2_ output in a manner identical to that described above for bit 0.<br>In loopback mode, connected to Data Carrier Detect (DCD) |
| | [2] | R/W | 0 | out1 | Out1<br>This bit controls the Output 1 (OUT1_) signal, which is an auxiliary user-designated output. Bit 2 affects the OUT1_ in a manner identical to that described above for bit 0.<br>In loopback mode, connected Ring Indicator (RI) signal input |
| | [1] | R/W | 0 | rts | Request to Send (RTS) signal control<br>0: RTS is logic 1<br>1: RTS is logic 0<br>This bit controls the Request to Send (RTS_) output. Bit 1 affects the RTS_ output in a manner identical to that described above for bit 0. |
| | [0] | R/W | 0 | dtr | Data Terminal Ready (DTR) signal control<br>0: DTR is logic 1<br>1: DTR is logic 0<br>This bit controls the Data Terminal Ready (DTR_) output. When bit 0 is set to a logic 1, the DTR_ output is forced to a logic 0. When bit0 is set to a logic 0, the DTR_ output is forced to a logic 1. |
| **Line Status Register** | | | | | |
| 0x14 | [31:8] | - | - | - | Reserved |
| | [7] | R | 0 | rxfifo_err | Uart_rx_error<br>1: In FIFO mode, this bit (LSR bit7) is set when there is at least a parity error, framing error, or break indication in the FIFO. LSR7 is clear when the CPU reads the LSR, if there are no subsequent errors in the FIFO. |
| | [6] | R | - | txfifo_empty | Transmitter Empty (TEMT) indicator (TXFIFO_empty)<br>0: Otherwise<br>1: This bit is set to logic 1 whenever the Transmitter Holding Register (THR) and the Transmitter Shift Register (TSR) are both empty. It is reset to logic 0 whenever either the THR or TSR contains a data character. In FIFO mode this bit is set to one whenever the transmitter FIFO and shift register are both empty. |
| | [5] | R | 0 | txfifo_empty | TX FIFO empty indicator.<br>0: Otherwise<br>1: Indicates that the UART is ready to accept a new character for transmission. In addition, this bit causes the UART to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register |
| | - | - | - | - | - |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [5] | R | 0 | txfifo_empty | The bit is reset to logic 0 concurrently with the loading of the Transmitter Holding Register by the CPU.<br>In FIFO mode, this bit is set when the Transmitter FIFO is empty; it is cleared when at least 1 byte is written to the Transmitter FIFO. |
| | [4] | R | - | break_err_int | Break Interrupt (BI) indicator<br>0: No break condition in the current character<br>1: Set to logic 1 whenever the received data input is held in the Spacing (logic 0) state for a longer than a full word transmission time (that is, the total time of Start bit + data bits + Parity + Stop bits). The BI indicator is reset whenever the CPU reads the contents of the Line Status Register.<br>This error is revealed to the CPU when its associated character is at the top of the FIFO. When a break occurs only one zero character is loaded into the FIFO. The next character transfer is enabled after Sin goes to the marking state (Logic 1) and receives the next valid start bit. |
| | [3] | R | 0 | framing_err | Framing Error (FE) indicator<br>0: No framing error in the current character<br>1: The received character at the top of the FIFO did not have a valid stop bit. Generally it indicates that all the following data is corrupt.<br>Bit 3 is set to a logic 1 whenever the Stop bit following the last data bit or parity bit is detected as a logic 0 bit (Spacing level). The FE indicator is reset whenever the CPU reads the contents of the Line Status Register.<br>This error is revealed to the CPU when its associated character is at the top of the FIFO. The UART will try to resynchronize after a framing error. To do this it assumes that the framing error was due to the next start bit, so it samples this 'start' bit twice and then takes in the 'data' |
| | [2] | R | 0 | parity_err | Parity Error (PE) indicator<br>0: No parity error in current character<br>1: Indicates that the received data character does not have the correct even or odd parity, as selected by the even-parity-select bit. The PE bit is set to a logic '1' upon detection of a parity error and is reset to logic 0 whenever the CPU reads the contents of the Line Status Register.<br>In FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO (next character to be read). |
| | [1] | R | 0 | overrun_err | Overrun Error (OE) indicator<br>0: No Overrun state<br>1: Indicates that data in the RX FIFO was not read by the CPU before the next character was transferred into the RX FIFO, thereby destroying the previous character. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [1] | R | 0 | overrun_err | The OE indicator is set to a logic 1 upon detection of an overrun condition and reset whenever the CPU reads the contents of the Line Status Register.<br>If the FIFO mode data continues to fill the FIFO beyond the trigger level, an overrun error will occur only after the FIFO is full and the next character has been completely received in the shift register. OE is indicated to the CPU as soon as it occurs. The character in the shift register is overwritten, but it is not transferred to the FIFO. |
| | [0] | R | 0 | rxfifo_datardy | Data Ready (DR) indicator<br>0: No characters in the Receiver FIFO<br>1: At least one character has been received and transferred into the FIFO<br>Bit0 is reset to logic '0' by reading all of the data in the Receiver Buffer Register or the RX FIFO. |
| **Modem Status Register** | | | | | |
| 0x18 | [7] | R | 0 | r_dcd | Complement of the DCD input or equals to Out2 in loopback mode. |
| | [6] | R | - | r_ri | Complement of the RI input or equals to Out1 in loopback mode. |
| | [5] | R | 0 | r_dsr | Complement of the DSR input or equals to DTR in loopback mode. |
| | [4] | R | - | r_cts | Complement of the CTS input or equals to RTS in loopback mode. |
| | [3] | R | - | d_dcd | Delta Data Carrier Detect (DDCD) indicator<br>1: The DCD line has changed its state<br>0: Otherwise. |
| | [2] | R | 0 | teri | Trailing Edge of Ring Indicator (TERI) detector. The RI line has changed its state from low to high state |
| | [1] | R | - | d_dsr | Delta Data Set Ready (DDSR) indicator<br>1: The DSR line has changed its state<br>0: Otherwise |
| | [0] | R | 0 | d_cts | Delta Clear to Send (DCTS) indicator<br>1: The CTS line has changed its state<br>0: Otherwise |
| **Scratch Pad Register** | | | | | |
| 0x1C | [31:27] | R/W | 0 | - | Reserved |
| | [26:16] | R/W | - | xfactor_adj[10:0] | One factor of Baud rate calculation, i.e., the ovsr_adj[10:0] of the following formula. |
| | [15:12] | R/W | 0 | - | Reserved |
| | [11:8] | R/W | - | Dbg_sel[3:0] | Debug port selection |
| | [7] | R/W | - | rx break signal interrupt status | Write one to clear |
| | [6] | R/W | 0 | rx break signal interrupt enable | - |
| | [5] | R/W | - | fl_set_bi_err | Delta Data Set Ready (DDSR) indicator<br>1: The DSR line has changed its state<br>0: Otherwise |
| | - | - | - | - | - |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [4] | R/W | 0 | fl_frame_err | Delta Clear to Send (DCTS) indicator<br>1: The CTS line has changed its state<br>0: Otherwise |
| | [3] | R/W | - | Pin_lb_test | For UART IP txd/rxd/rts/cts pin loopback test |
| | [2:0] | R/W | 0 | Reserved | Delta Clear to Send (DCTS) indicator.<br>1: The CTS line has changed its state<br>0: Otherwise |
| **STS Register** | | | | | |
| 0x20 | [31:8] | R/W | 0 | - | Reserved |
| | [7:4] | R/W | - | xfactor | Factor of Baud rate calculation, i.e., the ovsr[3:0] of following formula. |
| | [3] | R/W | 0 | rx break signal interrupt enable | Reset UART Receiver |
| | [2:0] | R/W | - | - | Reserved |
| **Receiver Buffer Register** | | | | | |
| 0x24 | [31:8] | R | 0 | - | Reserved |
| | [7] | R | 0 | rxdatabit7 | RX data bit 7 |
| | [6] | R | 0 | rxdatabit6 | RX data bit 6 |
| | [5] | R | 0 | rxdatabit5 | RX data bit 5 |
| | [4] | R | 0 | rxdatabit4 | RX data bit 4 |
| | [3] | R | 0 | rxdatabit3 | RX data bit 3 |
| | [2] | R | 0 | rxdatabit2 | RX data bit 2 |
| | [1] | R | 0 | rxdatabit1 | RX data bit 1 |
| | [0] | R | 0 | rxdatabit0 | RX data bit 0<br>*Note: Bit 0 is the least significant bit. It is the first bit serially received.* |
| **Transmitter Holding Register** | | | | | |
| 0x24 | [31:8] | - | 0 | - | Reserved |
| | [7] | W | 0 | txdatabit7 | TX data bit 7 |
| | [6] | W | 0 | txdatabit6 | TX data bit 6 |
| | [5] | W | 0 | txdatabit5 | TX data bit 5 |
| | [4] | W | 0 | txdatabit4 | TX data bit 4 |
| | [3] | W | 0 | txdatabit3 | TX data bit 3 |
| | [2] | W | 0 | txdatabit2 | TX data bit 2 |
| | [1] | W | 0 | txdatabit1 | TX data bit 1 |
| | [0] | W | 0 | txdatabit0 | TX data bit 0<br>*Note: Bit 0 is the least significant bit. It is the first bit serially transmitted.* |

# 10.11. IR RC

Table 22. IR RC (Base Address: 0x4004_1000)

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| REG_00 | | | | | |
| 0x00 | [31:30] | - | - | - | Reserved |
| | [29:24] | R | - | RX_FIFO_OFFSET | RX FIFO offset<br>0x0: rx FIFO is empty<br>0x1: 1 rx data<br>0x2: 2 rx data<br>…<br>0x1f: 31 rx data<br>0x20: rx FIFO is full |
| | [23:22] | - | - | - | Reserved |
| | [21:16] | R | 0 | TX_FIFO_OFFSET | TX FIFO offset<br>0x0: tx FIFO is empty<br>0x1: 1 tx data<br>0x2: 2 tx data<br>…<br>0x1f: 31 tx data<br>0x20: tx FIFO is full |
| | [15:10] | - | - | - | Reserved |
| | [9] | R | 0 | RX_INT_ALL_FLAG | RX interrupt all flag<br>RX_INT_ALL_FLAG=<br>RX_INT_TH_FLAG \| RX_FIFO_<br>FULL_FLAG \| RX_START_FLAG \|<br>RX_END_FLAG<br>0x1: RX interrupt occurred<br>0x0: No RX interrupt occurred |
| | [8] | R | 0 | TX_INT_ALL_FLAG | TX Interrupt all flag<br>TX_INT_ALL_FLAG=<br>TX_INT_TH_FLAG \| TX_FIFO_<br>EMPTY_FLAG<br>0x1: TX interrupt occurred<br>0x0: No TX interrupt occurred |
| | [7] | R | 0 | RX_END_FLAG | Detect valid RX end<br>0x1: Valid RX end is detected<br>0x0: Valid RX end is not detected |
| | [6] | R | 0 | RX_START_FLAG | Detect valid RX start<br>0x1: Valid RX start is detected<br>0x0: Valid RX start is not detected |
| | [5] | R | 0 | RX_FIFO_FULL_FLAG | RX FIFO FULL flag<br>0x1: FIFO is full<br>0x0: FIFO is not full |
| | [4] | R | 0 | RX_FIFO_ EMPTY_<br>FLAG | RX FIFO EMPTY flag<br>0x1: FIFO is empty<br>0x0: FIFO is not empty |
| | [3] | R | 0 | RX_INT_TH_FLAG | RX Interrupt flag<br>When fifo_offset=rx_int_th, HW issues an interrupt to notify FW.<br>0x1: Interrupt occurred<br>0x0: No interrupt |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [2] | R | 0 | TX_FIFO_FULL_FLAG | TX FIFO Full Flag<br>0x1: FIFO is full<br>0x0: FIFO is not full |
| | [1] | R | 0 | TX_FIFO_ EMPTY_ FLAG | TX FIFO Empty Flag<br>0x1: FIFO is empty<br>0x0: FIFO is not empty |
| | [0] | R | 0 | TX_INT_TH_FLAG | TX Interrupt flag<br>When fifo_offset=tx_int_th, HW issues an interrupt to notify FW.<br>0x1: Interrupt occurred<br>0x0: No interrupt |
| **REG_04** | | | | | |
| 0x04 | [31:28] | - | - | - | Reserved |
| | [27:16] | R/W | 12'd370 | IR_DUTY_NUM | Duty cycle setting for modulation frequency<br>E.G.: for 1/3 duty cycle, IR_DUTY_NUM=(IR_DIV_ NUM+1)/3 -1 |
| | [15:12] | - | - | - | Reserved |
| | [11:0] | R/W | 12'd1111 | IR_DIV_NUM | Divider number to generate Irda modulation frequency<br>E.G.: sys_clk=40MHz<br>modulation_freq=38KHz<br>IR_DIV_NUM=(sys_clk / modulation_freq) - 1 |
| **REG_08** | | | | | |
| 0x08 | [31] | - | - | - | Reserved |
| | [30] | W | 0 | IR_FIFO_CLR | Clear FIFO data<br>Write 1 to clear |
| | [29] | W | 0 | IR_RX_INT_END_CLR | Clear RX valid end interrupt<br>Write 1 to clear |
| | [28] | W | 0 | IR_RX_INT_START_CLR | Clear RX valid start interrupt<br>Write 1 to clear |
| | [27] | W | 0 | IR_RX_INT_FULL_CLR | Clear RX FIFO full interrupt<br>Write 1 to clear |
| | [26] | W | 0 | IR_RX_INT_TH_CLR | Clear RX FIFO threshold interrupt<br>Write 1 to clear |
| | [25] | W | 0 | R_TX_INT_EMPTY_CLR | Clear TX FIFO empty interrupt<br>Write 1 to clear |
| | [24] | - | - | - | Reserved |
| | [23] | W | - | IR_TX_INT_TH_CLR | Clear TX FIFO threshold interrupt<br>Write 1 to clear |
| | [22] | - | - | - | Reserved |
| | [21] | R/W | 1 | IR_RX_FIFO_FULL_EN | IRDA RX FIFO full Interrupt enable<br>0x1: Enable<br>0x0: Disable |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [20] | R/W | 1 | IR_RX_INT_EN | IRDA RX Interrupt enable<br>0x1: Enable rx start/end/FIFO threshold interrupt<br>0x0: Disable |
| | [19:18] | - | - | - | Reserved |
| | [17] | R/W | 1 | IR_TX_FIFO_EMPTY_EN | IRDA TX FIFO empty Interrupt enable<br>0x1: Enable<br>0x0: Disable |
| | [16] | R/W | 1 | IR_TX_INT_EN | IRDA TX Interrupt enable<br>When tx_fifo_offset=tx_int_th, HW raise interrupt to notify FW<br>0x1: Enable<br>0x0: Disable |
| | [15:9] | - | - | - | Reserved |
| | [8] | R/W | 0 | IR_RX_START_MAN | RX start control manual mode<br>0x1: RX start/end is fully controlled by FW<br>0x0: RX start/end id controlled by HW |
| | [7:2] | R/W | 0 | IR_DIV_1ms_NUM | The 1ms divider number<br>E.G.: for modulation clock=38KHz, IR_DIV_1ms_NUM=38 -1 |
| | [1] | R/W | 0 | IR_RX_EN | Enable IRDA RX Function |
| | [0] | R/W | 0 | IR_TX_EN | Enable IRDA TX Function |
| **REG_0C** | | | | | |
| 0x0C | [31] | - | - | - | Reserved |
| | [30:25] | R | 0 | IR_TX_TOTAL_NUM | Total TX Count<br>The overall TX data number during TX start=1.<br>The max number is 0x3f |
| | [24:17] | R | 0 | IR_TX_TIME | TX Total Time<br>Unit=1m Sec<br>The max number is 0xff |
| | [16] | R | 0 | IR_TX_STATUS | IR TX Status<br>0: Idle<br>1: Busy |
| | [15] | R | 0 | IR_FIFO_TX_FULL | 1: FIFO Full |
| | [14] | R | 0 | IR_FIFO_TX_EMPTY | 1: FIFO Empty |
| | [13:8] | R | 0 | IR_FIFO_TX_OFFSET | TX FIFO offset<br>0: FIFO is empty<br>1: One data in FIFO<br>2: Two data in FIFO<br>…<br>31: 31 data in FIFO<br>32: FIFO is full |
| | [7] | R | 0 | IR_TX_INT_FLAG | IF (IR_FIFO_TX_OFFSET == IR_TX_INT_TH_NUM)<br>Then Interrupt<br>0x1: TX offset=TX threshold |
| | [6:2] | R/W | 0x8 | IR_TX_INT_TH_NUM | TX Interrupt threshold number=0~31 |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [1] | R/W | 0 | IR_TX_INVERSE | TX Inverse<br>0x0: Normal<br>0x1: inverse |
| | [0] | R/W | 0 | IR_TX_START | TX Begin<br>0x1: TX start<br>0x0: TX idle |
| **REG_10** | | | | | |
| 0x10 | [15] | W | 0x0 | IR_TX_DATA_TYPE | Waveform Time Type<br>0: Low<br>1: High |
| | [14] | W | 0x0 | IR_TX_DATA_END_FLAG | Waveform ended<br>0: Waveform not ended<br>1: Waveform ended |
| | [13:0] | W | 0x0 | IR_TX_DATA_TIME | Waveform Time<br>Unit: (1/modulation_freq)<br>Range: 0~16383 |
| **REG_14** | | | | | |
| | [31] | R | - | IR_RX_END_D_INT | Detect valid end flag<br>1: HW detect RX valid end<br>0: No RX valid end |
| | [30] | R | - | IR_RX_START_D_INT | Detect valid start flag<br>1: HW detect RX valid start<br>0: No RX valid start |
| | [29:23] | R | - | IR_RX_TOTAL_NUM | Total RX Count<br>The overall RX data number during RX start=1.<br>The max number is 0x7f |
| | [22:15] | R | - | IR_RX_TIME | RX Total Time<br>Unit=1m Sec<br>The max number is 0xff |
| | [14] | R | - | IR_RX_STATUS | IR RX Status<br>0: Idle<br>1: Busy |
| | [13] | R | - | IR_FIFO_RX_FULL | 1:FIFO Full |
| | [12] | R | - | IR_FIFO_RX_EMPTY | 1: FIFO Empty |
| | [11:6] | R | - | IR_FIFO_RX_OFFSET | RX FIFO offset<br>0: FIFO is empty<br>1: One data in FIFO<br>2: Two data in FIFO<br>…<br>31: 31 data in FIFO<br>32: FIFO is full |
| | [5] | R | - | IR_RX_INT_FLAG | IF (IR_FIFO_RX_OFFSET = IR_RX_INT_TH_NUM)<br>Then Interrupt |
| | [4:2] | - | - | - | Reserved |
| | [1] | R/W | 0 | IR_RX_INVERSE | RX Inverse<br>0x0: Normal<br>0x1: Invert RX data |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [0] | R/W | 0 | IR_RX_START | RX Begin<br>0x1: Start<br>0x0: Idle |
| **REG_18** | | | | | |
| 0x18 | [31:27] | R/W | 5'd24 | IR_RX_INT_TH_NUM | RX Interrupt threshold number= 0~31 |
| | [26] | - | - | - | Reserved |
| | [25:16] | R/W | 10'd380 | IR_ DETECT _END_TIME | Detect Valid End time =<br>(IR_ DETECT _END_TIME-1)<br>* (1/modulation_freq) |
| | [15:10] | - | - | - | Reserved |
| | [9:0] | R/W | 10'd570 | IR_ DETECT _START_TIME | Detect Valid Start time =<br>(IR_ DETECT _START_TIME-1)<br>* (1/modulation_freq) |
| **REG_1C** | | | | | |
| 0x1C | [15] | R | - | RX_DATA_TYPE | Waveform Time Type<br>0: Low<br>1: High |
| | [14] | R | - | IR_RX_DATA_END _FLAG | Waveform ended<br>0: Waveform not ended<br>1: Waveform ended |
| | [13:0] | R | - | IR__RX_DATA_TIME | Waveform Time<br>Unit: (1/modulation_freq)<br>Range: 0~16383 |
| **REG_20** | | | | | |
| 0x20 | [31:0] | R | - | - | Reserved |

## *10.12. SPI0/SPI1*

There are two individual SPI interfaces in the RTL8762A. SPI0 supports master and slave mode. SPI1 supports master mode only. The RTL8762A supports uni-directional 4 pin SPI (SPI_CLK, SPI_MISO, SsPI_MOSI, SPI_CS_N).

**Table 23. SPI0/SPI1 (Base Address: 0x4000_1000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| 0x308 | [31:9] | - | - | - | Reserved |
| | [8] | R/W | 0 | SPI0_MASTER _MODE | SPI0 master mode select<br>0: Slave mode<br>1: Master mode |
| | [7:0] | - | - | - | Reserved |

**Table 24. SPI0/SPI1 (Base Address: 0x4004_2000)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **CTRLR0: Control Register 0** | | | | | |
| 0x000 (SPI0) 0x400 (SPI1) | [15:11] | - | - | - | Reserved |
| | [10] | R/W | 0x0 | SLV_OE | Slave Output Enable. Relevant only when the DW_apb_ssi has been configured as a serial-slave device. When configured as a serial master, this bit field has no functionality. This bit enables or disables the setting of the ssi_oe_n output from the DW_apb_ssi serial slave.<br>When SLV_OE =1, the ssi_oe_n output can never be active. When the ssi_oe_n output controls the tri-state buffer on the txd output from the slave, a high impedance state is always present on the slave txd output when SLV_OE=1. This is useful when the master transmits in broadcast mode (master transmits data to all slave devices). Only one slave may respond with data on the master rxd line.<br>This bit is enabled after reset and must be disabled by software (when broadcast mode is used), if you do not want this device to respond with data.<br>0: Slave txd is enabled<br>1: Slave txd is disabled |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [9:8] | R/W | 0x0 | TMOD | Transfer Mode. Selects the mode of transfer for serial communication. This field does not affect the transfer duplicity. Only indicates whether receive or transmit data are valid.<br>In transmit-only mode, data received from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer.<br>In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer.<br>In transmit-and-receive mode, both transmit and receive data are valid. The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be accessed by the host processor.<br>In EEPROM-read mode, receive data is not valid while control data is being transmitted.<br>When all control data is sent to the EEPROM, receive data becomes valid and transmit data becomes invalid. All data in the transmit FIFO is considered control data in this mode. This transfer mode is only valid when the DW_apb_ssi is configured as a master<br>device.<br>00: Transmit & Receive<br>01: Transmit Only<br>10: Receive Only<br>11: EEPROM Read |
| | [7] | R/W | 0x0 | SCPOL | Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the DW_apb_ssi master is not actively transferring data on the serial bus.<br>0: Inactive state of serial clock is low<br>1: Inactive state of serial clock is high |
| | [6] | R/W | 0x0 | SCPH | Serial Clock Phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH=0, data are captured on the first edge of the serial clock. When SCPH=1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock.<br>0: Serial clock toggles in middle of first data bit<br>1: Serial clock toggles at start of first data bit |
| | [5:4] | R/W | 0x0 | FRF | Frame Format. Selects which serial protocol transfers the data.<br>00: Motorola SPI<br>01: Reserved<br>10: Reserved<br>11: Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3:0] | R/W | 0x7 | DFS | Data Frame Size. Selects the data frame length. When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded.<br>0000: Reserved – undefined operation<br>0001: Reserved – undefined operation<br>0010: Reserved – undefined operation<br>0011: 4-bit serial data transfer<br>0100: 5-bit serial data transfer<br>0101: 6-bit serial data transfer<br>0110: 7-bit serial data transfer<br>0111: 8-bit serial data transfer<br>1000: 9-bit serial data transfer<br>1001: 10-bit serial data transfer<br>1010: 11-bit serial data transfer<br>1011: 12-bit serial data transfer<br>1100: 13-bit serial data transfer<br>1101: 14-bit serial data transfer<br>1110: 15-bit serial data transfer<br>1111: 16-bit serial data transfer |
| **CTRLR1: Control Register 1** | | | | | |
| 0x004 (SPI0)<br>0x404 (SPI1) | [15:0] | R/W | 0x0 | NDF | Number of Data Frames. When TMOD=10 or TMOD=11, this register field sets the number of data frames to be continuously received by the DW_apb_ssi. The DW_apb_ssi continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64KB of data in a continuous transfer.<br>When the DW_apb_ssi is configured as a serial slave, the transfer continues for as long as the slave is selected. Therefore, this register serves no purpose and is not present when the DW_apb_ssi is configured as a serial slave. |
| **SSIENR: SSI Enable Register** | | | | | |
| 0x008 (SPI0)<br>0x408 (SPI1) | [0] | R/W | 0x00 | SSI_EN | SSI Enable. Enables and disables all DW_apb_ssi operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the DW_apb_ssi control registers when enabled.<br>When disabled, the ssi_sleep output is set (after delay) to inform the system that it is safe to remove the ssi_clk, thus saving power consumption in the system.<br>Reset Value: 0x0 |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **SER: Slave Enable Register** | | | | | |
| 0x010 (SPI0) | [31:8] | - | - | - | Reserved |
| | [7:0] | R/W | 0x0 | SER | Slave Select Enable Flag. Each bit in this register corresponds to a slave select line (ss_x_n]) from the DW_apb_ssi master. When a bit in this register is set (1), the corresponding slave select line from the master is activated when a serial transfer begins. Setting or clearing bits in this register has no effect on the corresponding slave select outputs until a transfer is started.<br>Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate.<br>When not operating in broadcast mode, only one bit in this field should be set.<br>1: Selected<br>0: Not Selected |
| 0x410 (SPI1) | [31:1] | - | - | - | Reserved |
| | [0] | - | - | SER | - |
| **BAUDR: Baud Rate Select** | | | | | |
| 0x014 (SPI0) 0x414 (SPI1) | [15:0] | R/W | 0x00 | SCKDV | SSI Clock Divider. The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the following equation:<br>F_sclk_out=F_ssi_clk<br>SCKDV<br>where SCKDV is any even value between 2 and 65534.<br>For example:<br>for Fssi_clk=3.6864MHz and SCKDV=2<br>Fsclk_out=3.6864/2=1.8432MHz |
| **TXFTLR: Transmit FIFO Threshold Level** | | | | | |
| 0x018 (SPI0) 0x418 (SPI1) | [31:5] | - | - | - | Reserved |
| | [4:0] | R/W | 0x0 | TFT | Transmit FIFO Threshold. Controls the level of entries at which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2~256; this register is sized to the number of address bits needed to access the FIFO.<br>If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value.<br>When the number of transmit FIFO entries is greater than or equal to this value + 1, the transmit FIFO full interrupt is triggered. |
| **RXFTLR: Receive FIFO Threshold Level** | | | | | |
| 0x01C | [31:4] | - | - | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| (SPI0) 0x41C (SPI1) | [3:0] | R/W | 0x0 | RFT | Receive FIFO Threshold. Controls the level of entries at which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2~256. This register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value.<br>When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered. |
| **TXFLR: Transmit FIFO Level Register** | | | | | |
| 0x020 (SPI0) 0x4020 (SPI1) | [31:6] | - | - | - | Reserved |
| | [5:0] | R | 0x0 | TXTFL | Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO. |
| **RXFLR: Receive FIFO Level Register** | | | | | |
| 0x024 (SPI0) 0x424 (SPI1) | [31:5] | - | - | - | Reserved |
| | [4:0] | R | 0x0 | RXTFL | Receive FIFO Level. Contains the number of valid data entries in the receive FIFO. |
| **SR: Status Register** | | | | | |
| 0x028 (SPI0) 0x428 (SPI1) | [6] | R | 0x0 | DCOL | Data Collision Error. Relevant only when the DW_apb_ssi is configured as a master device.<br>This bit is set if the DW_apb_ssi master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read.<br>0: No error<br>1: Transmit data collision error |
| | [5] | R | 0x0 | TXE | Transmission Error. Set if the transmit FIFO is empty when a transfer is started. This bit can be set only when the DW_apb_ssi is configured as a slave device. Data from the previous transmission is resent on the txd line. This bit is cleared when read.<br>0: No error<br>1: Transmission error |
| | [4] | R | 0x0 | RFF | Receive FIFO Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty locations, this bit is cleared.<br>0: Receive FIFO is not full<br>1: Receive FIFO is full |
| | [3] | R | 0x0 | RFNE | Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO.<br>0: Receive FIFO is empty<br>1: Receive FIFO is not empty |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [2] | R | 0x0 | TFE | Transmit FIFO Empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt.<br>0: Transmit FIFO is not empty<br>1: Transmit FIFO is empty |
| | [1] | R | 0x1 | TFNF | Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full.<br>0: Transmit FIFO is full<br>1: Transmit FIFO is not full |
| | [0] | R | 0x0 | BUSY | SSI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the DW_apb_ssi is idle or disabled.<br>0: DW_apb_ssi is idle or disabled<br>1: DW_apb_ssi is actively transferring data |
| **IMR: Interrupt Mask Register** | | | | | |
| 0x02C (SPI0) 0x42C (SPI1) | [31:6] | - | - | - | Reserved |
| | [5] | R/W | 0x1 | MSTIM | Multi-Master Contention Interrupt Mask. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device.<br>0: ssi_mst_intr interrupt is masked<br>1: ssi_mst_intr interrupt is not masked |
| | [4] | R/W | 0x1 | RXFIM | Receive FIFO Full Interrupt Mask<br>0: ssi_rxf_intr interrupt is masked<br>1: ssi_rxf_intr interrupt is not masked |
| | [3] | R/W | 0x1 | RXOIM | Receive FIFO Overflow Interrupt Mask<br>0: ssi_rxo_intr interrupt is masked<br>1: ssi_rxo_intr interrupt is not masked |
| | [2] | R/W | 0x1 | RXUIM | Receive FIFO Underflow Interrupt Mask<br>0: ssi_rxu_intr interrupt is masked<br>1: ssi_rxu_intr interrupt is not masked |
| | [1] | R/W | 0x1 | TXOIM | Transmit FIFO Overflow Interrupt Mask<br>0: ssi_txo_intr interrupt is masked<br>1: ssi_txo_intr interrupt is not masked |
| | [0] | R/W | 0x1 | TXEIM | Transmit FIFO Empty Interrupt Mask<br>0: ssi_txe_intr interrupt is masked<br>1: ssi_txe_intr interrupt is not masked |
| **ISR: Interrupt Status Register** | | | | | |
| 0x030 (SPI0) 0x430 (SPI1) | [31:6] | - | - | - | Reserved |
| | [5] | R | 0x0 | MSTIS | Multi-Master Contention Interrupt Status. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device.<br>0: ssi_mst_intr interrupt not active after masking<br>1: ssi_mst_intr interrupt is active after masking |
| | [4] | R | 0x0 | RXFIS | Receive FIFO Full Interrupt Status<br>0: ssi_rxf_intr interrupt is not active after masking<br>1: ssi_rxf_intr interrupt is full after masking |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3] | R | 0x0 | RXOIS | Receive FIFO Overflow Interrupt Status<br>0: ssi_rxo_intr interrupt is not active after masking<br>1: ssi_rxo_intr interrupt is active after masking |
| | [2] | R | 0x0 | RXUIS | Receive FIFO Underflow Interrupt Status<br>0: ssi_rxu_intr interrupt is not active after masking<br>1: ssi_rxu_intr interrupt is active after masking |
| | [1] | R | 0x0 | TXOIS | Transmit FIFO Overflow Interrupt Status<br>0: ssi_txo_intr interrupt is not active after masking<br>1: ssi_txo_intr interrupt is active after masking |
| | [0] | R | 0x0 | TXEIS | Transmit FIFO Empty Interrupt Status<br>0: ssi_txe_intr interrupt is not active after masking<br>1: ssi_txe_intr interrupt is active after masking |
| **RISR: Raw Interrupt Status Register** | | | | | |
| 0x034 (SPI0)<br>0x434 (SPI1) | [31:6] | - | - | - | Reserved |
| | [5] | R | 0x0 | MSTIR | Multi-Master Contention Raw Interrupt Status. This bit field is not present if the<br>DW_apb_ssi is configured as a serial-slave device.<br>0: ssi_mst_intr interrupt is not active prior to masking<br>1: ssi_mst_intr interrupt is active prior masking |
| | [4] | R | 0x0 | RXFIR | Receive FIFO Full Raw Interrupt Status<br>0: ssi_rxf_intr interrupt is not active prior to masking<br>1: ssi_rxf_intr interrupt is active prior to masking |
| | [3] | R | 0x0 | RXOIR | Receive FIFO Overflow Raw Interrupt Status<br>0: ssi_rxo_intr interrupt is not active prior to masking<br>1: ssi_rxo_intr interrupt is active prior masking |
| | [2] | R | 0x0 | RXUIR | Receive FIFO Underflow Raw Interrupt Status<br>0: ssi_rxu_intr interrupt is not active prior to masking<br>1: ssi_rxu_intr interrupt is active prior to masking |
| | [1] | R | 0x0 | TXOIR | Transmit FIFO Overflow Raw Interrupt Status<br>0: ssi_txo_intr interrupt is not active prior to masking<br>1: ssi_txo_intr interrupt is active prior masking |
| | [0] | R | 0x0 | TXEIR | Transmit FIFO Empty Raw Interrupt Status<br>0: ssi_txe_intr interrupt is not active prior to masking<br>1: ssi_txe_intr interrupt is active prior masking |
| **TXOICR: Transmit FIFO Overflow Interrupt Clear Register** | | | | | |
| 0x038<br>0x438 | [0] | R | 0x0 | TXOICR | Clear Transmit FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_txo_intr interrupt; writing has no effect. |
| **RXOICR: Receive FIFO Overflow Interrupt Clear Register** | | | | | |
| 0x03C<br>0x43C | [0] | R | 0x0 | RXOICR | Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxo_intr interrupt; writing has no effect. |
| **RXUICR: Receive FIFO Underflow Interrupt Clear Register** | | | | | |
| 0x040 (SPI0)<br>0x400 (SPI1) | [0] | R | 0x0 | RXUICR | Clear Receive FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxu_intr interrupt; writing has no effect. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **MSTICR: Multi-Master Interrupt Clear Register** | | | | | |
| 0x044<br>0x444 | [0] | R | 0x0 | MSTICR | Clear Multi-Master Contention Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_mst_intr interrupt; writing has no effect. |
| **ICR: Interrupt Clear Register** | | | | | |
| 0x048<br>(SPI0)<br>0x448<br>(SPI1) | [0] | R | 0x0 | ICR | Clear Interrupts. This register is set if any of the interrupts below are active. A read clears the ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr, and the ssi_mst_intr interrupts. Writing to this register has no effect. |
| **DMACR: DMA Control Register** | | | | | |
| 0x04C<br>(SPI0)<br>0x44C<br>(SPI1) | [1] | R/W | 0x0 | TDMAE | Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel.<br>0: Transmit DMA disabled<br>1: Transmit DMA enabled |
| | [0] | R/W | 0x0 | RDMAE | Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel<br>0: Receive DMA disabled<br>1: Receive DMA enabled |
| **DMATDLR: DMA Transmit Data Level** | | | | | |
| 0x050<br>(SPI0)<br>0x450<br>(SPI1) | [4:0] | R/W | 0x0 | 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. |
| **DMARDLR: DMA Receive Data Level** | | | | | |
| 0x054<br>(SPI0)<br>0x454<br>(SPI1) | [3:0] | R/W | 0x0 | DMARD | 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 above this field value + 1, and RDMAE=1. |

## *10.13. I2C*

There are two separate I2C interfaces in the RTL8762A. Each I2C interface is comprised of serial data line (SDA) and serial clock (SCL). Both I2C interfaces can be configured to master or slave mode.

Features:

- Supports standard mode (0~100kb/s) and fast mode (less than or equal to 400kb/s)
- 7/10 bit device address
- 7/10 bit combined format transfer
- Bulk transmit mode

Table 25. I2C (Base Address: 0x4004_4000(I2C0) & 0x4004_4400(I2C1))

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| IC_CON: I2C Control Register | | | | | |
| 0x00 | [15:7] | - | - | - | Reserved |
| | [6] | R/W | 0x1 | IC_SLAVE_DISABLE | This bit controls whether I2C has its slave disabled. You have the choice of having the slave enabled or disabled after reset is applied, which means software does not have to configure the slave. By default, the slave is always enabled (in reset state as well).<br>If you need to disable it after reset, set this bit to 1.<br>If this bit is set (slave is disabled), DW_apb_i2c functions only as a master and does not perform any action that requires a slave.<br>0: Slave is enabled<br>1: Slave is disabled |
| | [5] | R/W | 0x1 | IC_RESTART_EN | Determines whether RESTART conditions may be sent when acting as a master. Some older slaves do not support handling RESTART conditions; however, RESTART conditions are used in several DW_apb_i2c operations.<br>0: Disable<br>1: Enable<br>When the RESTART is disabled, the DW_apb_i2c master is incapable of performing the following functions:<br>• Sending a START BYTE<br>• Performing any high-speed mode operation<br>• Performing direction changes in combined format mode<br>• Performing a read operation with a 10-bit address<br>By replacing RESTART condition followed by a STOP and a subsequent START condition, split operations are broken down into multiple DW_apb_i2c transfers. If the above operations are performed, it will result in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. |
| | [4] | R | - | IC_10BITADDR_MASTER_rd_only | The function of this bit is handled by bit 12 of IC_TAR register<br>0: 7-bit addressing<br>1: 10-bit addressing |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [3] | R/W | 0x1 | IC_10BITADDR_SLAVE | When acting as a slave, this bit controls whether the DW_apb_i2c responds to 7- or 10-bit addresses.<br>0: 7-bit addressing. The DW_apb_i2c ignores transactions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the IC_SAR register are compared.<br>1: 10-bit addressing. The DW_apb_i2c responds to only 10-bit addressing transfers that match the full 10 bits of the IC_SAR register. |
| | [2:1] | R/W | 0x3 | SPEED | These bits control at which speed the DW_apb_i2c operates; its setting is relevant only if one is operating the DW_apb_i2c in master mode.<br>1: Standard mode (0 to 100 kbit/s)<br>2: Fast mode (≤ 400 kbit/s) |
| | [0] | R/W | 0x1 | MASTER_MODE | This bit controls whether the DW_apb_i2c master is enabled.<br>0: Master disabled<br>1: Master enabled |
| **IC_TAR: I2C Target Address Register** | | | | | |
| 0x04 | [15:13] | - | - | - | Reserved |
| | [12] | R/W | 0x1 | IC_10BITADDR_MASTER | This bit controls whether the DW_apb_i2c starts its transfers in 7-or 10-bit addressing mode when acting as a master.<br>0: 7-bit addressing<br>1: 10-bit addressing |
| | [11] | R/W | 0x0 | SPECIAL | This bit indicates whether software performs a General Call or START BYTE command.<br>0: Ignore bit 10 GC_OR_START and use IC_TAR normally<br>1: Perform special I2C command as specified in GC_OR_START bit |
| | [10] | R/W | 0x0 | GC_OR_START | If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the DW_apb_i2c.<br>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 DW_apb_i2c remains in General Call mode until the SPECIAL bit value (bit 11) is cleared.<br>1: START BYTE |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [9:0] | R/W | 10'h055 | IC_TAR | This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE, the CPU needs to write only once into these bits. |
| **IC_SAR: I2C Slave Address Register** | | | | | |
| 0x08 | [15:10] | - | - | - | Reserved |
| | [9:0] | R/W | 10'h055 | IC_SAR | The IC_SAR holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only IC_SAR[6:0] is used. |
| **IC_HS_MADDR: I2C High Speed Master Mode Code Address Register** | | | | | |
| 0x0C | [15:3] | - | - | - | Reserved |
| | [2:0] | R/W | 0x1 | IC_HS_MAR | 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. |
| **IC_DATA_CMD:** | | | | | |
| 0x10 | [15:11] | - | - | - | Reserved |
| | [10] | W | - | RESTART | This bit controls whether a RESTART is issued before the byte is sent or received.<br>1: 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.<br>0: RESTART is issued only if the transfer direction is changing from the previous command |
| | [9] | W | - | STOP | This bit controls whether a STOP is issued after the byte is sent or received.<br>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.<br>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. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [8] | W | 0x0 | CMD | This bit controls whether a read or a write is performed. This bit does not control the direction when the DW_apb_i2c acts as a slave. It controls only the direction when it acts as a master.<br>1: Read<br>0: Write<br>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 the data in IC_DATA_CMD is to be transmitted.<br>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 IC_RAW_INTR_STAT register), unless bit 11 (SPECIAL) in the IC_TAR register has been cleared.<br>If a '1' is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs. |
| | [7:0] | R/W | 0x0 | 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 DW_apb_i2c. However, when you read this register, these bits return the value of data<br>received on the DW_apb_i2c interface. |
| **IC_SS_SCL_HCNT :Standard Speed I2C Clock SCL High Count Register** | | | | | |
| 0x14 | [15:0] | R/W | 16'h 0190 | IC_SS_SCL_HCNT | 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. |
| **IC_SS_SCL_LCNT: Standard Speed I2C Clock SCL Low Count Register** | | | | | |
| 0x18 | [15:0] | R/W | 16'h 01d6 | IC_SS_SCL_LCNT | 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. |
| **IC_FS_SCL_HCNT: Fast Speed I2C Clock SCL High Count Register** | | | | | |
| 0x1C | [15:0] | R/W | 16'h 003c | IC_FS_SCL_HCNT | 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. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **IC_FS_SCL_LCNT: Fast Speed I2C Clock SCL Low Count Register** | | | | | |
| 0x20 | [15:0] | R/W | 16'h 0082 | IC_FS_SCL_LCNT | 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. |
| **IC_INTR_STAT: I2C Interrupt Status Register** | | | | | |
| 0x2C | [15:12] | - | - | - | Reserved |
| | [11] | R | 0x0 | R_GEN_CALL | See IC_RAW_INTR_STAT(0x34) for a detailed description of these bits |
| | [10] | R | 0x0 | R_START_DET | |
| | [9] | R | 0x0 | R_STOP_DET | |
| | [8] | R | 0x0 | R_ACTIVITY | |
| | [7] | R | 0x0 | R_RX_DONE | |
| | [6] | R | 0x0 | R_TX_ABRT | |
| | [5] | R | 0x0 | R_RD_REQ | |
| | [4] | R | 0x0 | R_TX_EMPTY | |
| | [3] | R | 0x0 | R_TX_OVER | |
| | [2] | R | 0x0 | R_RX_FULL | |
| | [1] | R | 0x0 | R_RX_OVER | |
| | [0] | R | 0x0 | R_RX_UNDER | |
| **IC_INTR_MASK: I2C Interrupt Mask Register** | | | | | |
| 0x30 | [15:12] | - | - | - | Reserved |
| | [11] | R/W | 12'h8ff | M_GEN_CALL | These bits mask their corresponding interrupt status bits in the IC_INTR_STAT register. |
| | [10] | R/W | | M_START_DET | |
| | [9] | R/W | | M_STOP_DET | |
| | [8] | R/W | | M_ACTIVITY | |
| | [7] | R/W | | M_RX_DONE | |
| | [6] | R/W | | M_TX_ABRT | |
| | [5] | R/W | | M_RD_REQ | |
| | [4] | R/W | | M_TX_EMPTY | |
| | [3] | R/W | | M_TX_OVER | |
| | [2] | R/W | | M_RX_FULL | |
| | [1] | R/W | | M_RX_OVER | |
| | [0] | R/W | | M_RX_UNDER | |
| - | | | | | |
| 0x34 | [15:12] | - | - | - | Reserved |
| | [11] | R | 0x0 | 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 DW_apb_i2c or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. DW_apb_i2c stores the received data in the RX buffer. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [10] | R | 0x0 | START_DET | Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether DW_apb_i2c is operating in slave or master mode. |
| | [9] | R | 0x0 | STOP_DET | Indicates whether a STOP condition has occurred on the I2C interface regardless of whether DW_apb_i2c is operating in slave or master mode. |
| | [8] | R | 0x0 | ACTIVITY | This bit captures DW_apb_i2c activity and stays set until it is cleared. There are four ways to clear it:<br>• Disabling the DW_apb_i2c<br>• Reading the IC_CLR_ACTIVITY register<br>• Reading the IC_CLR_INTR register<br>• System reset<br>Once this bit is set, it stays set unless one of the four methods is used to clear it.<br>Even if the DW_apb_i2c module is idle, this bit remains set until cleared, indicating that there was activity on the bus. |
| | [7] | R | 0x0 | RX_DONE | When the DW_apb_i2c 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. |
| | [6] | R | 0x0 | TX_ABRT | This bit indicates if DW_apb_i2c, 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’.<br>When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. |
| | [5] | R | 0x0 | RD_REQ | This bit is set to 1 when DW_apb_i2c is acting as a slave and another I2C master is attempting to read data from DW_apb_i2c. The DW_apb_i2c 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.<br>The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [4] | R | 0x0 | TX_EMPTY | This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold.<br>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.<br>When there is no longer activity, then with ic_en=0, this bit is set to 0. |
| | [3] | R | 0x0 | TX_OVER | Set during transmit if the transmit buffer is filled to IC_TX_BUFFER_DEPTH and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register.<br>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. |
| | [2] | R | 0x0 | RX_FULL | Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold.<br>If the module is disabled (IC_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 IC_ENABLE bit 0 is programmed with a 0, regardless of ongoing activity. |
| | [1] | R | 0x0 | RX_OVER | Set if the receive buffer is completely filled to IC_RX_BUFFER_DEPTH and an additional byte is received from an external I2C device. The DW_apb_i2c acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_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. |
| | [0] | R | 0x0 | RX_UNDER | Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register.<br>If the module is disabled (IC_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. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **IC_RX_TL: I2C Receive FIFO Threshold Register** | | | | | |
| 0x38 | [15:8] | - | - | - | Reserved |
| | [7:0] | R/W | 0xb | RX_TL | Receive FIFO Threshold Level.<br>Controls the level of entries (or above) that triggers the RX_FULL interrupt (bit 2 in IC_RAW_INTR_STAT register) |
| **IC_TX_TL: I2C Transmit FIFO Threshold Register** | | | | | |
| 0x3C | [15:8] | - | - | - | Reserved |
| | [7:0] | R/W | 0x3 | TX_TL | Transmit FIFO Threshold Level.<br>Controls the level of entries (or below) that trigger the TX_EMPTY interrupt (bit 4 in IC_RAW_INTR_STAT register). |
| **IC_CLR_INTR: Clear Combined and Individual Interrupt Register** | | | | | |
| 0x40 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_INTR | Read this register to clear the combined interrupt, all individual interrupts, and the IC_TX_ABRT_SOURCE register. This bit does not clear hardware clearable interrupts but software clearable interrupts. |
| **IC_CLR_RX_UNDER: Clear RX_UNDER Interrupt Register** | | | | | |
| 0x44 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_RX_UNDER | Read this register to clear the RX_UNDER interrupt (bit 0) of the IC_RAW_INTR_STAT register. |
| **IC_CLR_RX_OVER: Clear RX_OVER Interrupt Register** | | | | | |
| 0x48 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_RX_OVER | Read this register to clear the RX_OVER interrupt (bit 1) of the IC_RAW_INTR_STAT register. |
| **IC_CLR_TX_OVER: Clear TX_OVER Interrupt Register** | | | | | |
| 0x4C | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_TX_OVER | Read this register to clear the TX_OVER interrupt (bit 3) of the IC_RAW_INTR_STAT register. |
| **IC_CLR_RD_REQ: Clear RD_REQ Interrupt Register** | | | | | |
| 0x50 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_RD_REQ | Read this register to clear the RD_REQ interrupt (bit 5) of the IC_RAW_INTR_STAT register. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **IC_CLR_TX_ABRT: Clear TX_ABRT Interrupt Register** | | | | | |
| 0x54 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_TX_ABRT | Read this register to clear the TX_ABRT interrupt (bit 6) of the IC_RAW_INTR_STAT register, and the IC_TX_ABRT_SOURCE register.<br>This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO.<br>Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE. |
| **IC_CLR_RX_DONE: Clear RX_DONE Interrupt Register** | | | | | |
| 0x58 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_RX_DONE | Read this register to clear the RX_DONE interrupt (bit 7) of the IC_RAW_INTR_STAT register. |
| **IC_CLR_ACTIVITY: Clear ACTIVITY Interrupt Register** | | | | | |
| 0x5C | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | 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.<br>The value read from this register gets the status of the ACTIVITY interrupt (bit 8) of the IC_RAW_INTR_STAT register. |
| **IC_CLR_STOP_DET: Clear STOP_DET Interrupt Register** | | | | | |
| 0x60 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_STOP_DET | Read this register to clear the STOP_DET interrupt (bit 9) of the IC_RAW_INTR_STAT register. |
| **IC_CLR_START_DET: Clear START_DET Interrupt Register** | | | | | |
| 0x64 | [15:1] | - | - | - | - |
| | [0] | R | 0x0 | CLR_START_DET | Read this register to clear the START_DET interrupt (bit 10) of the IC_RAW_INTR_STAT register. |
| **IC_CLR_GEN_CALL: Clear GEN_CALL Interrupt Register** | | | | | |
| 0x68 | [15:1] | - | - | - | Reserved |
| | [0] | R | 0x0 | CLR_GEN_CALL | Read this register to clear the GEN_CALL interrupt (bit 11) of IC_RAW_INTR_STAT register. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **IC_ENABLE: I2C Enable Register** | | | | | |
| 0x6C | [15:1] | - | - | - | - |
| | [0] | R/W | 0x0 | ENABLE | Controls whether the DW_apb_i2c is enabled.<br>0: Disables DW_apb_i2c (TX and RX FIFOs are held in an erased state)<br>1: Enables DW_apb_i2c<br>Software can disable DW_apb_i2c while it is active. |
| **IC_STATUS: I2C Status Register** | | | | | |
| 0x70 | [31:7] | - | - | - | Reserved |
| | [6] | R | 0x0 | SLV_ACTIVITY | Slave FSM Activity Status. When the Slave Finite State Machine (FSM) is not in the IDLE state, this bit is set.<br>0: Slave FSM is in IDLE state so the Slave part of DW_apb_i2c is not Active<br>1: Slave FSM is not in IDLE state so the Slave part of DW_apb_i2c is Active |
| | [5] | R | 0x0 | MST_ACTIVITY | Master FSM Activity Status. When the Master Finite State Machine (FSM) is not in the IDLE state, this bit is set.<br>0: Master FSM is in IDLE state so the Master part of DW_apb_i2c is not Active<br>1: Master FSM is not in IDLE state so the Master part of DW_apb_i2c is Active |
| | [4] | R | 0x0 | 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.<br>0: Receive FIFO is not full<br>1: Receive FIFO is full |
| | [3] | R | 0x0 | 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.<br>0: Receive FIFO is empty<br>1: Receive FIFO is not empty |
| | [2] | R | 0x1 | 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.<br>0: Transmit FIFO is not empty<br>1: Transmit FIFO is empty |
| | [1] | R | 0x1 | TFNF | Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full.<br>0: Transmit FIFO is full<br>1: Transmit FIFO is not full |
| | [0] | R | 0x0 | ACTIVITY | I2C Activity Status. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **IC_TXFLR :I2C Transmit FIFO Level Register** | | | | | |
| 0x74 | [31:6] | - | - | - | Reserved |
| | [5:0] | R | 0x0 | TXFLR | Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO. |
| **IC_RXFLR: I2C Receive FIFO Level Register** | | | | | |
| 0x78 | [31:5] | - | - | - | - |
| | [4:0] | R | 0x0 | RXFLR | Receive FIFO Level. Contains the number of valid data entries in the receive FIFO. |
| **IC_SDA_HOLD: I2C SDA Hold Time Length Register** | | | | | |
| 0x7C | [31:16] | - | - | - | Reserved |
| | [15:0] | R/W | 16'h 0001 | IC_SDA_HOLD | Sets the required SDA hold time in units of ic_clk period. |
| **IC_TX_ABRT_SOURCE: I2C Transmit Abort Source Register** | | | | | |
| 0x80 | [31:16] | - | - | - | Reserved |
| | [15] | R | 0x0 | ABRT_SLVRD_INTX | 1: 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 IC_DATA_CMD register. |
| | [14] | R | 0x0 | ABRT_SLV_ARBLOST | 1: Slave lost the bus while transmitting data to a remote master. IC_TX_ABRT_SOURCE[12] is set at the same time. |
| | [13] | R | 0x0 | ABRT_SLVFLUSH_TXFIFO | 1: Slave has received a read command and some data exists in the TX FIFO, so the slave issues a TX_ABRT interrupt to flush old data in TX FIFO. |
| | [12] | R | 0x0 | ARB_LOST | 1: Master has lost arbitration, or if IC_TX_ABRT_SOURCE[14] is also set, then the slave transmitter has lost arbitration. |
| | [11] | R | 0x0 | ABRT_MASTER_DIS | 1: User tried to initiate a Master operation with the Master mode disabled. |
| | [10] | R | 0x0 | ABRT_10B_RD_NORSTRT | 1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5])=0) and the master sends a read command in 10-bit addressing mode. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [9] | R | 0x0 | ABRT_SBYTE_NORSTRT | To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; restart must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]).<br>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.<br>1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5])=0) and the user is trying to send a START Byte. |
| | [8] | R | 0x0 | ABRT_HS_NORSTRT | 1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5])=0) and the user is trying to use the master to transfer data in High Speed mode. |
| | [7] | R | 0x0 | ABRT_SBYTE_ACKDET | 1: Master has sent a START Byte and the START Byte was acknowledged (wrong behavior). |
| | [6] | R | 0x0 | ABRT_HS_ACKDET | 1: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior). |
| | [5] | R | 0x0 | ABRT_GCALL_READ | 1: DW_apb_i2c 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). |
| | [4] | R | 0x0 | ABRT_GCALL_NOACK | 1: DW_apb_i2c in master mode sent a General Call and no slave on the bus acknowledged the General Call. |
| | [3] | R | 0x0 | ABRT_TXDATA_NOACK | 1: This is a master-mode only bit. Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledge from the remote slave(s). |
| | [2] | R | 0x0 | ABRT_10ADDR2_NOACK | 1: Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave. |
| | [1] | R | 0x0 | ABRT_10ADDR1_NOACK | 1: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave. |
| | [0] | R | 0x0 | ABRT_7B_ADDR_NOACK | 1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **IC_SLV_DATA_NACK_ONLY: Generate Slave Data NACK Register** | | | | | |
| 0x84 | [31:1] | - | - | - | Reserved |
| | [0] | R/W | 0x0 | NACK | Generate NACK. This NACK generation only occurs when DW_apb_i2c is a slave-receiver. If this register is set to a value of 1, it can only generate a NACK after a data byte is received; hence, the data transfer is aborted and the data received is not pushed to the receive buffer.<br>When the register is set to a value of 0, it generates NACK/ACK, depending on normal criteria.<br>1: Generate NACK after data byte received<br>0: Generate NACK/ACK normally |
| **IC_DMA_CR: DMA Control Register** | | | | | |
| 0x88 | [31:2] | - | - | - | Reserved |
| | [1] | R/W | 0x0 | TDMAE | Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel.<br>0: Transmit DMA disabled<br>1: Transmit DMA enabled |
| | [0] | R/W | 0x0 | RDMAE | Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel.<br>0: Receive DMA disabled<br>1: Receive DMA enabled |
| **IC_DMA_TDLR: DMA Transmit Data Level Register** | | | | | |
| 0x8C | [31:5] | - | - | - | Reserved |
| | [4:0] | R/W | 0x0 | 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<br>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. |
| **IC_DMA_RDLR: I2C Receive Data Level Register** | | | | | |
| 0x90 | [31:4] | - | - | - | Reserved |
| | [3:0] | R/W | 0x00 | 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.<br>For instance, when DMARDL is 0, then dma_rx_req is asserted when 1 or more data entries are present in the receive FIFO. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **IC_SDA_SETUP: I2C SDA Setup Register** | | | | | |
| 0x94 | [31:8] | - | - | - | Reserved |
| | [7:0] | R/W | 0x64 | SDA_SETUP | SDA Setup. We recommend that if the required delay is 1000ns, then for an ic_clk frequency of 10MHz, IC_SDA_SETUP should be programmed to a value of 11. IC_SDA_SETUP must be programmed with a minimum value of 2. |
| **IC_ACK_GENERAL_CALL: I2C ACK General Call Register** | | | | | |
| 0x98 | [31:1] | - | - | - | Reserved |
| | [0] | R/W | 0x1 | ACK_GEN_CALL | ACK General Call. When set to 1, DW_apb_i2c responds with a ACK (by asserting ic_data_oe) when it receives a General Call. When set to 0, the DW_apb_i2c does not generate General Call interrupts. |
| **IC_EN: ABLE_STATUS: I2C Enable Status Register** | | | | | |
| 0x9C | [31:3] | - | - | - | Reserved |
| | [2] | R | 0x0 | 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. |
| | [1] | R | 0x0 | 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.<br>This bit is set when the CPU writes a 0 to the IC_ENABLE register while: (a) DW_apb_i2c is receiving the address byte of the Slave-Transmitter operation from a remote master; OR, (b) address and data bytes of the Slave-Receiver operation from a remote master. |
| | [0] | R | 0x0 | IC_EN | ic_en Status. This bit always reflects the value driven on the output port ic_en.<br>When read as 1, DW_apb_i2c is deemed to be in an enabled state.<br>When read as 0, DW_apb_i2c is deemed completely inactive. |

# *10.14. UART*

There are three UARTs for different scenarios; LOG_UART, HCI_UART, and DATA_UART. LOG_UART is for firmware debug. HCI_UART is for the MUTE interface, RF test, and image download. DATA_UART is a general UART.

The RTL8762A UART interface is a standard 4-wire interface with RX, TX, CTS, and RTS. The HCI UART interface supports the Bluetooth 2.0 UART HCI H4 and H5 specifications. The default baud rate is 115.2 kbaud.

The RTL8762A provides multiple UART baud-rate configured by register setting. The common band-rate example is shown in Table 26 below. The UART can function between two devices and has a clock error within +-2.5%.

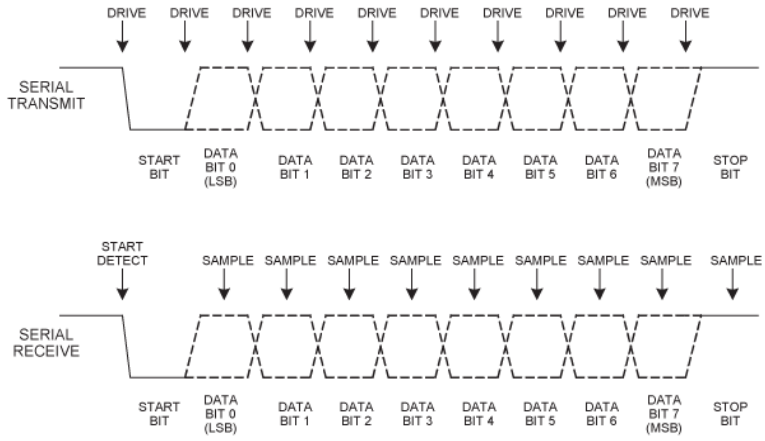


**Figure 15. UART Waveform**

**Table 26. UART BaudRate**

| BaudRate (bps) | Error (%) | BaudRate (bps) | Error (%) |
|---|---|---|---|
| 1200 | -0.11 | 460800 | 0.16 |
| 9600 | 0.26 | 500000 | 0.00 |
| 14400 | 0.28 | 921600 | -0.41 |
| 19200 | -0.31 | 1000000 | 0.00 |
| 28800 | -0.32 | 1382400 | 0.17 |
| 38400 | 0.27 | 1444400 | -0.31 |
| 57600 | -0.32 | 1500000 | 0.00 |
| 76800 | -0.06 | 1843200 | -0.35 |
| 115200 | 0.16 | 2000000 | -0.01 |
| 128000 | -0.44 | 2100000 | 0.24 |
| 153600 | 0.16 | 2764800 | 0.14 |
| 230400 | 0.16 | 3000000 | -0.03 |

# 10.15. GDMA

**Table 27. GDMA (Base Addresses Listed Below)**

GDMA channel 0 Base address: 0x40060000
GDMA channel 1 Base address: 0x40060058
GDMA channel 2 Base address: 0x400600B0
GDMA channel 3 Base address: 0x40060108
GDMA channel 4 Base address: 0x40060160
GDMA channel 5 Base address: 0 x400601B8

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **SARx x can be 0, 1, 2, 3, 4, 5** | | | | | |
| 0x00 | [63:32] | - | - | - | Reserved |
| | [31:0] | R/W | 0x00 | SAR | Source Address of DMA transfer.<br>Updated after each source transfer.<br>The SINC field in the CTLx register determines whether the address increments, decrements, or is left unchanged on every source transfer throughout the block transfer. |
| **DARx x can be 0, 1, 2, 3, 4, 5** | | | | | |
| 0x08 | [63:32] | - | - | - | Reserved |
| | [31:0] | R/W | 0x00 | DAR | Destination address of DMA transfer.<br>Updated after each destination transfer.<br>The DINC field in the CTLx register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. |
| **Reserved** | | | | | |
| 0x10 | [63:0] | - | - | - | Reserved |
| **CTLx x can be 0, 1, 2, 3, 4, 5** | | | | | |
| 0x18 | [63:32] | - | - | - | Reserved |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [b:32] | R/W | - | BLOCK_TS | Block Transfer Size.<br>When the RTK_ocp_dmac is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size.<br>The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single OCP beat.<br>Width: The width of the single transaction is determined by CTLx.SRC_TR_WIDTH.<br>Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of the flow controller.<br>When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAC_CHx_MAX_BLK_SIZE, but the actual block size<br>can be greater.<br>b=log2(DMAC_CHx_MAX_BLK_SIZE + 1) + 31<br>Bits 43:b+1 do not exist and return 0 on a read. |
| | [31:23] | - | - | - | Reserved |
| | [22:20] | R/W | - | TT_FC | Transfer Type and Flow Control. The following transfer types are<br>Supported:<br>• Memory to Memory<br>• Memory to Peripheral<br>• Peripheral to Memory<br>• Peripheral to Peripheral<br>Flow Control can be assigned to the RTK_ocp_dmac, the source peripheral, or the destination peripheral |
| | [19:17] | - | - | - | Reserved |
| | [16:14] | R/W | - | SRC_MSIZE | Source Burst Transaction Length.<br>Number of data items, each of width CTLx.SRC_TR_WIDTH, to be read from the source each time a source burst transaction request is made from either the corresponding hardware or software handshaking interface |
| | [13:11] | R/W | - | DEST_MSIZE | Destination Burst Transaction Length.<br>Number of data items, each of width CTLx.DST_TR_WIDTH, to be written to the destination each time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [10:9] | R/W | - | SINC | Source Address Increment. Indicates whether to increment the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to 'No change'.<br>00: Increment<br>01: Decrement<br>1x: No change |
| | [8:7] | R/W | - | DINC | Destination Address Increment. Indicates whether to increment destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to 'No change'.<br>00: Increment<br>01: Decrement<br>1x: No change |
| | [6:4] | R/W | - | SRC_TR_WIDTH | Source Transfer Width. |
| | [3:1] | R/W | - | DST_TR_WIDTH | Destination Transfer Width. |
| | [0] | R/W | - | INT_EN | Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTLx.INT_EN=0. |
| **Reserved** | | | | | |
| 0x20 | [63:0] | - | - | - | Reserved |
| **Reserved** | | | | | |
| 0x28 | [63:0] | - | - | - | Reserved |
| **Reserved** | | | | | |
| 0x30 | [63:0] | - | - | - | Reserved |
| **Reserved** | | | | | |
| 0x38 | [63:0] | - | - | - | Reserved |
| **CFGx x can be 0, 1, 2, 3, 4, 5** | | | | | |
| 0x40 | [63:43] | - | - | - | Reserved |
| | [b:43] | R/W | - | DEST_PER | Assigns a hardware handshaking interface (0 - DMAC_NUM_HS_INT-1) to the destination of channel x if the CFGx.HS_SEL_DST field is 0; otherwise, this field is ignored.<br>The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [b:39] | R/W | - | SRC_PER | Assigns a hardware handshaking interface (0 - DMAC_NUM_HS_INT-1) to the source of channel x if the CFGx.HS_SEL_SRC field is 0; otherwise, this field is ignored.<br>The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. |
| | [31] | R/W | - | RELOAD_DST | Automatic Destination Reload. The DARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated.<br>This register does not exist if the configuration parameter DMAC_CHx_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. |
| | [30] | R/W | - | RELOAD_SRC | Automatic Source Reload. The SARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated.<br>This field does not exist if the configuration parameter<br>DMAC_CHx_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. |
| | [29:20] | - | - | - | Reserved |
| | [19] | R/W | - | SRC_HS_POL | Source Handshaking Interface Polarity.<br>0: Active high<br>1: Active low |
| | [18] | R/W | - | DST_HS_POL | Destination Handshaking Interface Polarity.<br>0: Active high<br>1: Active low |
| | [17:12] | - | - | - | Reserved |
| | [11] | - | - | HS_SEL_SRC | Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces – hardware or software – is active for source requests on this channel.<br>0: Hardware handshaking interface. Software-initiated<br>transaction requests are ignored.<br>1: Software handshaking interface. Hardware-initiated transaction requests are ignored.<br>If the source peripheral is memory, then this bit is ignored. |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [10] | - | - | HS_SEL_DST | Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces – hardware or software – is active for destination requests on this channel.<br>0: Hardware handshaking interface. Software-initiated transaction requests are ignored.<br>1: Software handshaking interface. Hardware-initiated transaction requests are ignored.<br>If the destination peripheral is memory, then this bit is ignored. |
| | [9] | R/W | - | FIFO_EMPTY | Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFGx.CH_SUSP to cleanly disable a channel.<br>1: Channel FIFO empty<br>0: Channel FIFO not empty |
| | [8] | - | - | - | Reserved |
| | [7:5] | - | - | CH_PRIOR | Channel priority. A priority of 7 is the highest priority, and 0 is the lowest. This field must be programmed within the following range: 0: (DMAH_NUM_CHANNELS – 1) |
| | [4:0] | - | - | - | Reserved |

**Table 28. GDMA (Base Address: 0x4004_4000(I2C0) & 0x4006_02C0)**

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **StatusTfr** | | | | | |
| 0x2e8 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAC_NUM_CHANNELS–1:0] | R | - | STATUS | Interrupt status |
| **StatusBlock** | | | | | |
| 0x2f0 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAC_NUM_CHANNELS–1:0] | R | - | STATUS | Interrupt status |
| **StatusSrcTran** | | | | | |
| 0x2f8 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAC_NUM_CHANNELS–1:0] | R | - | STATUS | Interrupt status |
| **StatusDstTran** | | | | | |
| 0x300 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAC_NUM_CHANNELS–1:0] | R | - | STATUS | Interrupt status |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **StatusErr** | | | | | |
| 0x308 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAC_NUM_CHANNELS–1:0] | R | - | STATUS | Interrupt status |
| **MaskTfr** | | | | | |
| 0x310 | [63:8+dnc] | - | - | - | Reserved |
| | [7+dnc:8] | W | - | INT_MASK_WE | Interrupt Mask Write Enable<br>0: Write disabled<br>1: Write enabled<br>dnc=DMAH_NUM_CHANNELS<br>Reset Value: 0x0 |
| | [7:dnc] | - | - | - | Reserved |
| | [dnc–1:0] | R/W | - | INT_MASK | Interrupt Mask<br>0: Masked<br>1: Unmasked<br>dnc=DMAH_NUM_CHANNELS<br>Reset Value: 0x0 |
| **MaskBlock** | | | | | |
| 0x318 | [63:8+dnc] | - | - | - | Reserved |
| | [7+dnc:8] | W | - | INT_MASK_WE | Interrupt Mask Write Enable<br>0: Write disabled<br>1: Write enabled<br>dnc= DMAH_NUM_CHANNELS<br>Reset Value: 0x0 |
| | [7:dnc] | - | - | - | Reserved |
| | [dnc–1:0] | R/W | - | INT_MASK | Interrupt Mask<br>0: Masked<br>1: Unmasked<br>dnc= DMAH_NUM_CHANNELS<br>Reset Value: 0x0 |
| **MaskSrcTran** | | | | | |
| 0x320 | [63:8+dnc] | - | - | - | Reserved |
| | [7+dnc:8] | W | - | INT_MASK_WE | Interrupt Mask Write Enable<br>0: Write disabled<br>1: Write enabled<br>dnc= DMAH_NUM_CHANNELS<br>Reset Value: 0x0 |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [7:dnc] | - | - | - | Reserved |
| | [dnc–1:0] | R/W | - | INT_MASK | Interrupt Mask<br>0: Masked<br>1: Unmasked<br>dnc= DMAH_NUM_ CHANNELS<br>Reset Value: 0x0 |
| **MaskDstTran** | | | | | |
| 0x328 | [63:8+dnc] | - | - | - | Reserved |
| | [7+dnc:8] | W | - | INT_MASK_WE | Interrupt Mask Write Enable<br>0: Write disabled<br>1: Write enabled<br>dnc= DMAH_NUM_ CHANNELS<br>Reset Value: 0x0 |
| | [7:dnc] | - | - | - | Reserved |
| | [dnc–1:0] | R/W | - | INT_MASK | Interrupt Mask<br>0: Masked<br>1: Unmasked<br>dnc= DMAH_NUM_ CHANNELS<br>Reset Value: 0x0 |
| **MaskErr** | | | | | |
| 0x330 | [63:8+dnc] | - | - | - | Reserved |
| | [7+dnc:8] | W | - | INT_MASK_WE | Interrupt Mask Write Enable<br>0: Write disabled<br>1: Write enabled<br>dnc= DMAH_NUM_CHANNEL S<br>Reset Value: 0x0 |
| | [7:dnc] | - | - | - | Reserved |
| | [dnc–1:0] | R/W | - | INT_MASK | Interrupt Mask<br>0: Masked<br>1: Unmasked<br>dnc= DMAH_NUM_CHANNEL S<br>Reset Value: 0x0 |
| **ClearTfr** | | | | | |
| 0x338 | [63:DMAH_NUM_ CHANNELS] | - | - | - | Reserved |
| | [DMAH_NUM_ CHANNELS–1:0] | W | - | CLEAR | Interrupt clear.<br>0: No effect<br>1: Clear interrupt |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| **ClearBlock** | | | | | |
| 0x340 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAH_NUM_CHANNELS–1:0] | W | - | CLEAR | Interrupt clear.<br>0: No effect<br>1: Clear interrupt |
| **ClearSrcTran** | | | | | |
| 0x348 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAH_NUM_CHANNELS–1:0] | W | - | CLEAR | Interrupt clear.<br>0: No effect<br>1: Clear interrupt |
| **ClearDstTran** | | | | | |
| 0x350 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAH_NUM_CHANNELS–1:0] | W | - | CLEAR | Interrupt clear.<br>0: No effect<br>1: Clear interrupt |
| **ClearErr** | | | | | |
| 0x358 | [63:DMAH_NUM_CHANNELS] | - | - | - | Reserved |
| | [DMAH_NUM_CHANNELS–1:0] | W | - | CLEAR | Interrupt clear.<br>0: No effect<br>1: Clear interrupt |
| **Reserved** | | | | | |
| 0x360~ 0x390 | [630] | - | - | - | Reserved |
| **DmaCfgReg** | | | | | |
| 0x398 | [63:1] | - | - | - | Reserved |
| | [0] | R/W | 0x00 | DMA_EN | DMA Configuration Register 63.<br>1: Undefined N/A 0x0 Reserved<br>0: DMA_EN R/W 0x0 RTK_ocp_dmac Enable bit.<br>0: RTK_ocp_dmac Disabled<br>1: RTK_ocp_dmac Enabled |
| **ChEnReg** | | | | | |
| 0x3A0 | [6314] | - | - | - | Reserved |
| | [13:8] | W | - | CH_EN_WE | Channel enable write enable |
| | - | - | - | - | - |

| Offset | Bit | Access | INI | Symbol | Description |
|---|---|---|---|---|---|
| | [5:0] | R/W | - | CH_EN | Enables/Disables the channel. Setting this bit enables a channel; clearing this bit disables the channel.<br>0: Disable the Channel<br>1: Enable the Channel<br>The ChEnReg.CH_EN bit is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has completed. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer |

# 11. Electrical and Thermal Characteristics

## 11.1. Temperature Limit Ratings

**Table 29. Temperature Limit Ratings**

| Parameter | Minimum | Maximum | Units |
|---|---|---|---|
| Storage Temperature | -55 | +125 | °C |
| Ambient Operating Temperature | 0 | +70 | °C |
| Junction Temperature | 0 | +125 | °C |

## 11.2. Power Supply DC Characteristics

**Table 30. Power Supply DC Characteristics**

| Symbol | Parameter | Minimum | Typical | Maximum | Units |
|---|---|---|---|---|---|
| VBAT | Single power source for whole chip | 1.8 | 3 | 3.6 | V |
| VDD_CORE<br>VD12_PA<br>VD12_RTX<br>VD12_SYN | 1.2V Core and RFAFE Supply Voltage | 1.10 | 1.2 | 1.32 | V |
| VDD_IO[Note] | Power for digital IO PADs | 1.8 | - | 3.6 | V |
| HVD | Power for switching regulator | 1.8 | - | 3.6 | V |

*Note: VDD_IO≤VBAT*

## 11.3. Embedded Flash Characteristics

**Table 31. Embedded Flash Characteristics**

| - | Condition | Min | Typical | Max |
|---|---|---|---|---|
| Memory Organization | - | - | 64K x 32 | - |
| Endurance | - | 10,000 | - | - |
| Data retention life time (year) | at 85 degree Celsius | 10 | - | - |

## 11.4. Switching Regulator Characteristics

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 32. Switching Regulator Characteristics**

| Parameter | Condition | Min | Typical | Max |
|---|---|---|---|---|
| Input voltage (V) | - | 1.8 | 3 | 3.6 |
| Output voltage (V) | - | - | 1.2 | - |
| Output current (mA) | Only for RTL8762A internal use | - | - | 50 |
| Over current protection (mA) | - | - | 680 | - |
| Recommended input capacitor (μF) | X5R | 4.7 | - | 10 |
| Recommended output capacitor (μF) | X5R | 4.7 | - | 10 |
| Recommended output inductor (μH) | 1. Power inductor<br>2. +-20%<br>3. Rated current> 0.8A | - | 4.7 | - |

## 11.5. RTX LDO Characteristics

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 33. RTX LDO Characteristics**

| Parameter | Condition | Min | Typical | Max |
|---|---|---|---|---|
| Input voltage (V) | - | 1.1 | - | 1.32 |
| Output voltage (V) | - | - | 1 | - |
| Output voltage accuracy (%) | - | -5 | - | 5 |
| Output current (mA) | Only for RTL8762A internal use | - | - | 6 |
| Quiescent current (μA) | - | - | 40 | |
| PSRR | At 1kHz tone, vin= 1.2V | 30 | 40 | 50 |

## 11.6. Synthesizer LDO Characteristics

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 34. Synthesizer LDO Characteristics**

| Parameter | Condition | Min | Typical | Max |
|---|---|---|---|---|
| Input voltage (V) | - | 1.1 | - | 1.32 |
| Output voltage (V) | - | - | 1 | - |
| Output voltage accuracy (%) | - | -5 | - | 5 |
| Output current (mA) | Only for RTL8762A internal use | - | - | 10 |
| Quiescent current (μA) | - | - | 40 | - |
| PSRR | At 1kHz tone, vin= 1.2V | 30 | 40 | 50 |

## 11.7. Retention LDO Characteristics

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 35. Retention LDO Characteristics**

| Parameter | Condition | Min | Typical | Max |
|---|---|---|---|---|
| Input voltage (V) | - | 1.8 | | 3.6 |
| Output voltage (V) | - | 0.65 | 0.75 | 1.2 |
| Output voltage accuracy (%) | - | -5 | - | 5 |
| Output current (mA) | Only for RTL8762A internal use | - | - | 2 |
| Quiescent current (μA) | - | - | 0.1 | - |

## 11.8. ESD Characteristics

**Table 36. ESD Characteristics**

| Parameter | Condition | Minimum | Typical | Maximum |
|---|---|---|---|---|
| HBM | All pins, test method: JESD22 | - | - | ±3500V |
| MM | All pins, test method: JESD22 | - | - | ±100V |
| CDM | All pins, test method: JESD22 | - | - | ±500V |

## 11.9. AUXADC Characteristics

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 37. AUXADC Characteristics**

| Parameter | Condition | Min | Typical | Max |
|---|---|---|---|---|
| **For Analog MIC Mode** | | | | |
| Sample rate (kHz) | - | - | 39.0625 | - |
| Input 3dB bandwidth (kHz) | - | - | 19.53 | - |
| SNR (A-weighting) (dB) | - | - | 75 | - |
| PGA gain (dB) | - | 0/ 14/ 20/ 24/ 30 /35 /40 /44 /50 | | |
| Gain error (dB) | - | - | ±0.25 | - |
| Frequency flatness (dB) | 1. PGA+ADC<br>2. Frequency range from 100Hz to 4kHz<br>3. PGA gain=30dB | - | ±1.5 | - |
| Total harmonic distortion (%) | PGA gain=24dB, ADC input level= -6dBFS | - | 0.16% | - |
| MIC Bias voltage (Volt) | VBAT=3.3V | 1.25/1.47/2.5/2.94 | | |
| **AUX Mode** | | | | |
| Input impedance (kOhm) | - | - | 44 | - |
| Input voltage range (Volt) | - | 0 | - | VDDIO |
| Full scaling signal swing (Volt) | - | - | - | 3.3 |
| Number of external channels | P2_0~P2_7 | - | 8 | - |
| Offset (mV) | With In-house calibration | - | TBD | - |
| Gain error (%) | With In-house calibration | - | TBD | - |
| Internal reference voltage(V) | - | - | 0.9 | - |
| Conversion time (µs) | - | - | 128 | - |
| Signal to noise ratio (dB) | 1. Does not take into account harmonic<br>2. -6dBFS<br>3. Test at 1kHz tone<br>4. Differential mode | - | 71.9 | - |

# 11.10. Radio Characteristics

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 38. General Radio Characteristics**

| Parameter | Condition | Minimum | Typical | Maximum |
|---|---|---|---|---|
| Frequency Range (MHz) | - | 2402 | - | 2480 |

**Table 39. RX Performance**

| Parameter | Condition | Minimum | Typical | Maximum |
|---|---|---|---|---|
| Sensitivity (dBm) | PER ≤30.8% | -94.5(note) | - | - |
| Maximum Input Level (dBm) | PER ≤ 30.8% | - | -1 | - |
| C/I | $C/I_{co\text{-}channel}$ (dB) | - | 6 | - |
| | $C/I_{+1MHz}$ (dB) | - | -9 | - |
| | $C/I_{-1MHz}$ (dB) | - | -8 | - |
| | $C/I_{+2MHz}$ (dB) | - | -38 | - |
| | $C/I_{-2MHz}$ (dB) | - | -32 | - |
| | $C/I_{+3MHz}$ (dB) | - | -41 | - |
| | $C/I_{Image}$ (dB) | - | -26 | - |
| | $C/I_{Image+1MHz}$ (dB) | - | -32 | - |
| | $C/I_{Image-1MHz}$ (dB) | - | -42 | - |
| Blocker Power (dBm) | 30~2000MHz, Wanted signal level =-67dBm | -30 | - | - |
| | 2003~2399MHz, Wanted signal level =-67dBm | -35 | - | - |
| | 2484~2997MHz, Wanted signal level =-67dBm | -35 | - | - |
| | 3000MHz~12.75GHz, Wanted signal level =-67dBm | -30 | - | - |
| Max PER Report Integrity | Wanted signal: -30dBm | - | 50% | - |
| Max Intermodulation level (dBm) | Wanted signal (f0): -64dBm<br>Worst intermodulation level @2f1-f2=f0, \|f1-f2\|=n MHz, n=3, 4, 5… | -50 | - | - |

*Note: 1. Does not include spur channel*

*Note: 2. Depends on PCB design and registers setting*

**Table 40. TX Performance**

| Parameter | Condition | Minimum | Typical | Maximum |
|---|---|---|---|---|
| Maximum Output Power (dBm) | - | - | 0 | - |
| Adjacent Channel Power Ratio (dBm) | +2MHz | - | - | -20 |
| | -2MHz | - | - | -20 |
| | >=+3MHz | - | - | -30 |
| | <=-3MHz | - | - | -30 |
| Modulation Characteristics | $\Delta f1_{avg}$ (KHz) | - | 250 | - |
| | Δf2max (KHz) | - | 200 | - |
| | $\Delta f2_{max}$ Pass Rate (%) | - | 100 | - |
| | $\Delta f2_{avg} / \Delta f1_{avg}$ | - | 0.88 | - |
| Carrier Frequency Offset and Drift | Average Fn (KHz) | - | 12.5 | - |
| | Drift Rate (KHz/50μs) | - | 7.4 | - |
| | Avg Drift (KHz/50μs) | - | 8 | - |
| | Max Drift (KHz/50μs) | - | 9 | - |
| Output power of second harmonic(dBm) | - | - | -50(note) | - |
| Output power of third harmonic(dBm) | - | - | -50(note) | - |
| Output power of fourth harmonic(dBm) | - | - | -50(note) | - |
| Output power of fifth harmonic (dBm) | - | - | -50(note) | - |

*Note: Tested by EVB with RF PI network.*

## 11.11. Digital IO Pin DC Characteristics

**Table 41. Digital IO Pin DC Characteristics**

| Parameter | Condition | Min | Typical | Max |
|---|---|---|---|---|
| Input high voltage (V) | VDDIO=3.3V | 2 | 3.3 | 3.6 |
| Input low voltage (V) | VDDIO=3.3V | - | 0 | 0.9 |
| Output high voltage (V) | VDDIO=3.3V | 2.97 | - | 3.3 |
| Output low voltage (V) | VDDIO=3.3V | 0 | - | 0.33 |
| Input high voltage (V) | VDDIO=2.8V | 1.8 | 2.8 | 3.1 |
| Input low voltage (V) | VDDIO=2.8V | - | 0 | 0.8 |
| Output high voltage (V) | VDDIO=2.8V | 2.5 | - | 3.1 |
| Output low voltage (V) | VDDIO=2.8V | 0 | - | 0.28 |
| Pull high and pull low resister (KOhm) | - | - | 10 | - |
| Driving current (mA) (output high and low current) | P0_2, P0_3, P0_4, P0_5 @VDDIO=3.3V | - | 20 | - |
| | Other PADs @VDDIO=3.3V | - | 8 | - |

| Parameter | Condition | Min | Typical | Max |
|---|---|---|---|---|
| Input high current (μA) | PAD configured as input mode | - | - | 0.1 |
| Input low current (μA) | PAD configured as input mode | - | - | 0.1 |

## *11.12. Boot Sequence*

The RTL8762A embeds a Power On Reset Circuit (POR), and power on sequence finite state machine to boot the system. A Power on timing diagram is shown as figure below. The RTL8762A also supports reset by an external reset IC through pin 'HW_RST_N'. The HW_RST_N boot sequence is shown in Figure 17, page 113.

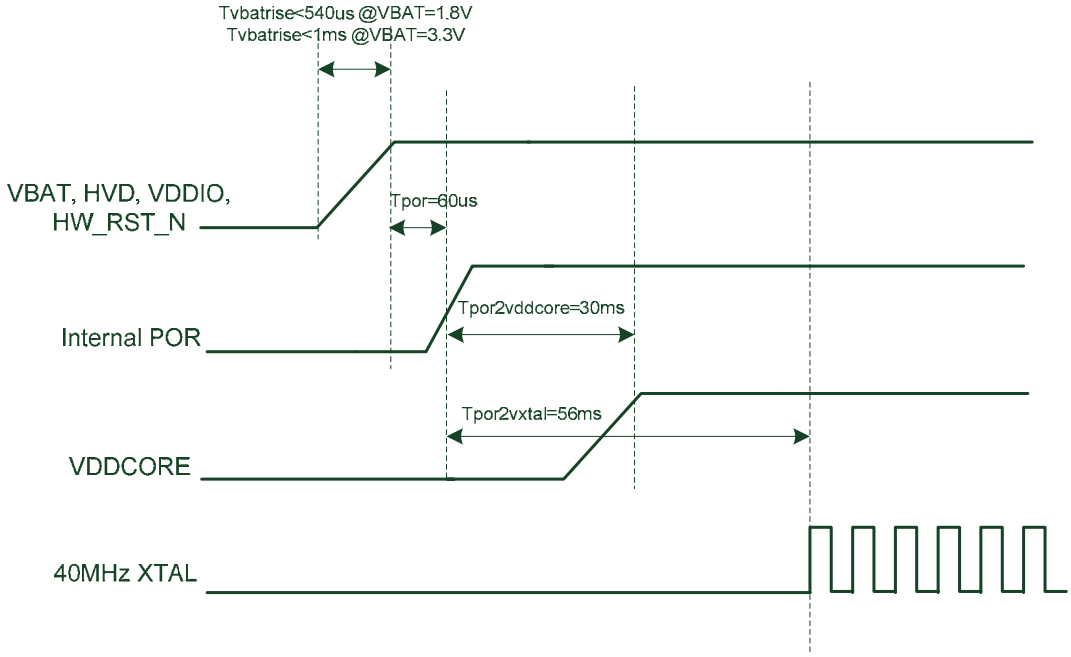


**Figure 16. Boot Up By Internal Power On Reset Circuit**

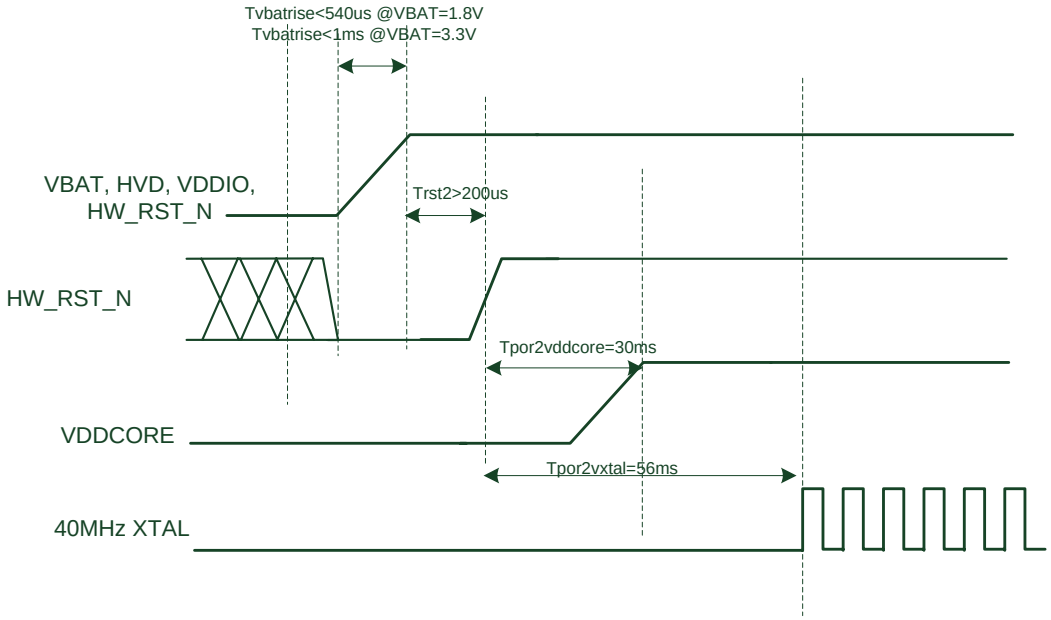


**Figure 17. Boot Up By HW_RST_N Pin**

## *11.13. UART Characteristics*

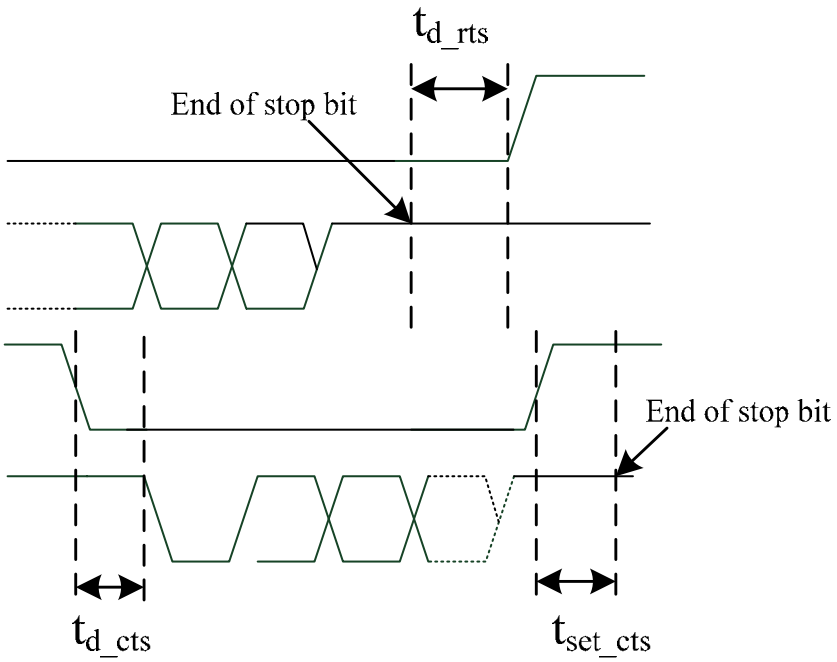


**Figure 18. UART Characteristics**

**Table 42. UART Timing Characteristics**

| Parameter | Symbol | Min | Typical | Max |
|---|---|---|---|---|
| Timing between RX Stop bit and RTS go high when RX FIFO is full (symbol time) | $t_{d\_rts}$ | - | - | 0.5 |
| Timing between CTS go low and device send first bit (ns) | $t_{d\_cts}$ | - | - | 25 |
| Timing between CTS go high and TX send stop bit (ns) | $t_{set\_cts}$ | 75 | - | - |

## 11.14. I2C Timing Characteristics

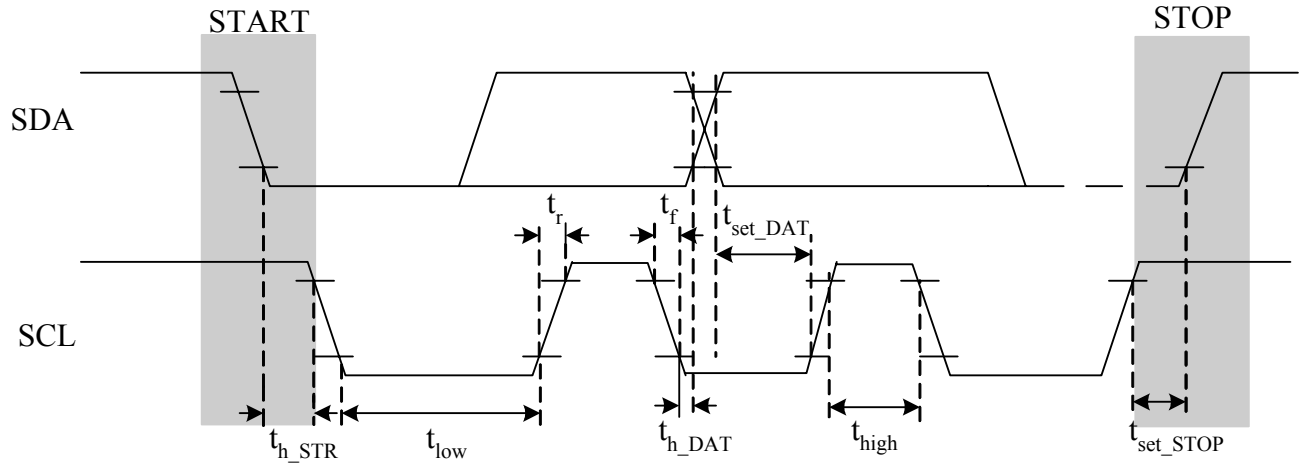


**Figure 19. I2C Interface Timing Diagram**

**Table 43. I2C Timing Characteristics**

| Parameter | Symbol | Min | Typical | Max |
|---|---|---|---|---|
| SCL clock frequency (kHz) | - | - | - | 400 |
| High period of SCL (ns) | $t_{high}$ | 600 | - | - |
| Low period of SCL (ns) | $t_{low}$ | 1300 | - | - |
| Hold time of START (ns) | $t_{h\_STR}$ | 600 | - | - |
| Hold time of DATA (ns) | $t_{h\_DAT}$ | 0 | - | - |
| Setup time of STOP (ns) | $t_{set\_STOP}$ | 600 | - | - |
| Setup time of DATA (ns) | $t_{set\_DAT}$ | 100 | - | - |
| Rise time of SCL and SDA (ns) (with 4.7k ohm resistor pulled high) | $t_r$ | See note | - | - |
| Fall time of SCA and SDA (ns) | $t_f$ | See note | - | - |

*Note: Depends on the external bus pull up resistor.*

# *11.15. Power Consumption*

## 11.15.1. Low Power Mode

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 44. Low Power Mode (Typical)**

| Power Mode | Always on Registers | 32k RCOSC/XTAL | Retention SRAM | CPU | Wakeup Method | Current Consumption |
|---|---|---|---|---|---|---|
| Power down | ON | OFF | OFF | OFF | Wakeup by GPIO | 200nA |
| Deep LPS | ON | ON | Retention | OFF | Wakeup by GPIO, timer | 1.1μA |

## 11.15.2. Active Mode

Condition: VBAT=3V, VDDIO=3V, Temperature: 25°C

**Table 45. Active Mode (Typical)**

| Power Mode | Current Consumption |
|---|---|
| Active RX mode | 6.9mA |
| Active TX mode (TX power: 0dBm) | 5.2mA |

# 12. Mechanical Dimensions

## 12.1. RTL8762AR/AG: Plastic Quad Flat No Lead Package 32 Leads 5mm$^2$ Outline

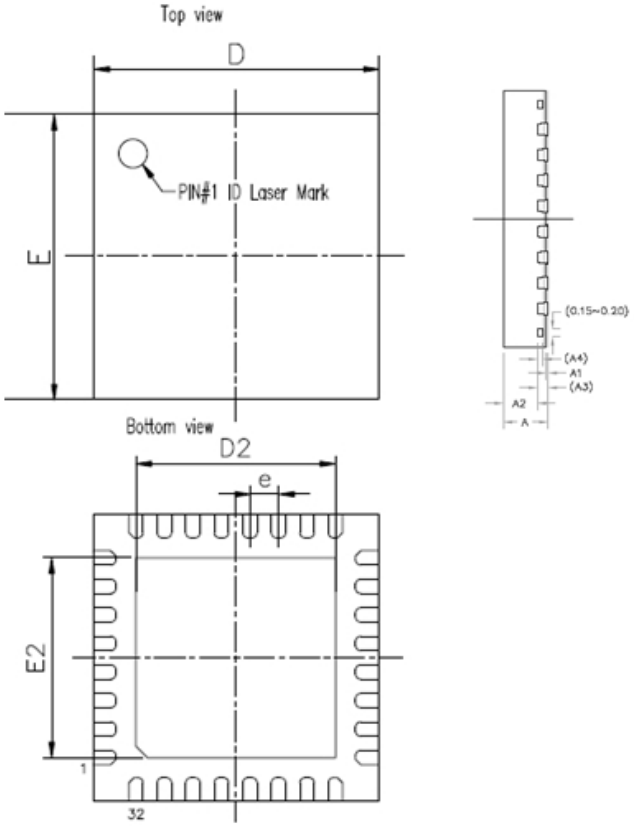


**Figure 20. RTL8762AR/AG: Plastic Quad Flat No Lead Package 32 Leads 5mm$^2$ Outline**

## 12.2. RTL8762AR/AG Mechanical Dimensions Notes

| Symbol | Dimension in mm | | | Dimension in inch | | |
|---|---|---|---|---|---|---|
| - | Min | Nom | Max | Min | Nom | Max |
| A | 0.80 | 0.85 | 0.90 | 0.031 | 0.033 | 0.035 |
| A1 | 0.00 | 0.02 | 0.05 | 0.000 | 0.001 | 0.002 |
| A3 | 0.20 REF | | | 0.008 REF | | |
| A4 | 0.10 REF | | | 0.004 REF | | |
| b | 0.18 | 0.25 | 0.30 | 0.007 | 0.010 | 0.012 |
| D/E | 5.00 BSC | | | 0.020 BSC | | |
| D2/E2 | 3.25 | 3.50 | 3.75 | 0.128 | 0.138 | 0.148 |
| e | 0.50 BSC | | | 0.020 BSC | | |
| L | 0.30 | 0.40 | 0.50 | 0.012 | 0.016 | 0.020 |

*Note 1: CONTROLLING DIMENSION: MILLIMETER (mm).*
*Note 2: REFERENCE DOCUMENT: JEDEC MO-220.*

## 12.3. RTL8762AJ: Plastic Quad Flat No Lead Package 40 Leads 5mm$^2$ Outline

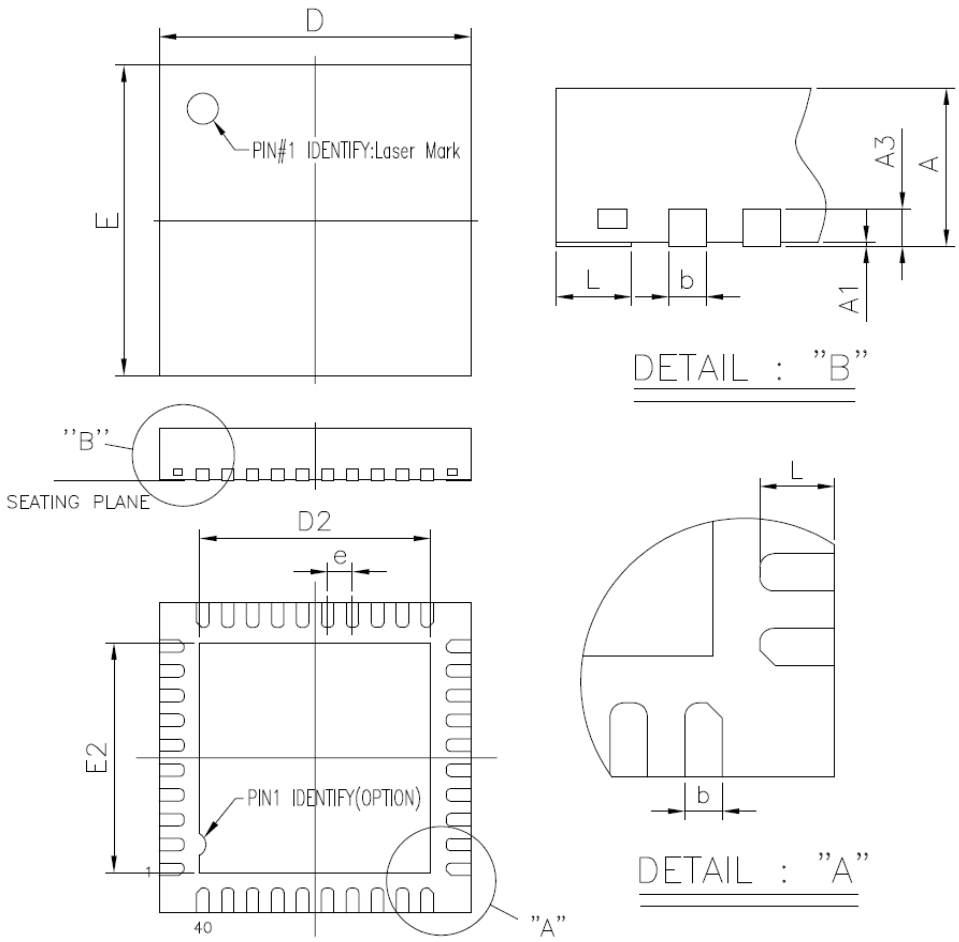


**Figure 21. RTL8762AJ: Plastic Quad Flat No Lead Package 40 Leads 5mm2 Outline**

## 12.4. RTL8762AJ Mechanical Dimensions Notes

| Symbol | Dimension in mm | | | Dimension in inch | | |
|---|---|---|---|---|---|---|
| | Min | Nom | Max | Min | Nom | Max |
| A | 0.80 | 0.85 | 0.90 | 0.031 | 0.033 | 0.035 |
| A1 | 0.00 | 0.02 | 0.05 | 0.000 | 0.001 | 0.002 |
| A3 | 0.20 REF | | | 0.008 REF | | |
| b | 0.15 | 0.20 | 0.25 | 0.006 | 0.008 | 0.010 |
| D/E | 5.00 BSC | | | 0.197 BSC | | |
| D2/E2 | 3.45 | 3.60 | 3.75 | 0.136 | 0.142 | 0.148 |
| e | 0.40 BSC | | | 0.016 BSC | | |
| L | 0.30 | 0.40 | 0.50 | 0.012 | 0.016 | 0.020 |

*Note 1: CONTROLLING DIMENSION: MILLIMETER (mm).*

*Note 2: REFERENCE DOCUMENT: JEDEC MO-220.*

## 12.5. RTL8762AK: Plastic Quad Flat No Lead Package 56 Leads 7mm² Outline

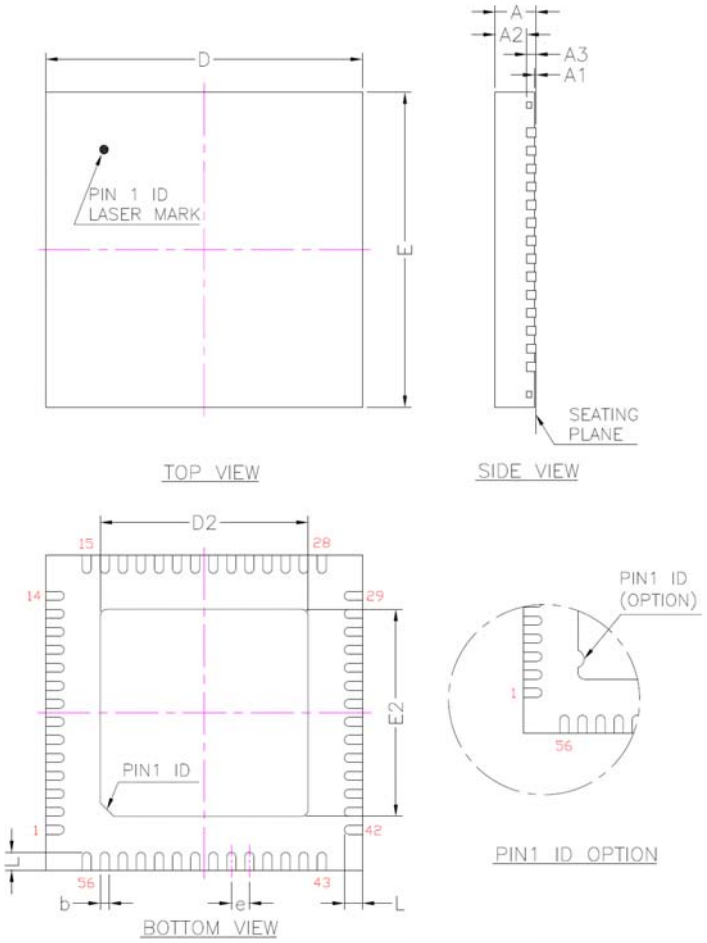


**Figure 22. RTL8762AK: Plastic Quad Flat No Lead Package 56 Leads 7mm² Outline**

## 12.6. RTL8762AK Mechanical Dimensions Notes

| Symbol | Dimension in mm | | | Dimension in inch | | |
|---|---|---|---|---|---|---|
| - | Min | Nom | Max | Min | Nom | Max |
| A | 0.80 | 0.85 | 0.90 | 0.031 | 0.033 | 0.035 |
| A1 | 0.00 | 0.02 | 0.05 | 0.000 | 0.001 | 0.002 |
| A3 | - | | | 0.65 | | |
| A4 | 0.2 REF | | | 0.008 REF | | |
| b | 0.15 | 0.20 | 0.25 | 0.006 | 0.008 | 0.010 |
| D/E | 7.00 BSC | | | 0.276 BSC | | |
| D2/E2 | 4.35 | 4.60 | 4.85 | 0.171 | 0.181 | 0.191 |
| e | 0.40 BSC | | | 0.016 BSC | | |
| L | 0.30 | 0.40 | 0.50 | 0.012 | 0.016 | 0.020 |

*Note 1: CONTROLLING DIMENSION: MILLIMETER (mm).*
*Note 2: REFERENCE DOCUMENT: JEDEC MO-220.*

# 13. Ordering Information

**Table 46. Ordering Information**

| Part Number | Package | Status |
|---|---|---|
| RTL8762AG-CG | QFN-32, 5mm$^2$ Outline; 'Green' Package | MP |
| RTL8762AR-CG | QFN-32, 5mm$^2$ Outline; 'Green' Package | MP |
| RTL8762AJ-CG | QFN-40, 5mm$^2$ Outline; 'Green' Package | MP |
| RTL8762AK-CG | QFN-56, 7mm$^2$ Outline; 'Green' Package | MP |

*Note: See section 5 Pin Assignments, page 6 for package identification.*

**Realtek Semiconductor Corp.**
**Headquarters**
No. 2, Innovation Road II, Hsinchu Science Park,
Hsinchu 300, Taiwan, R.O.C.
Tel: 886-3-5780211 Fax: 886-3-5776047
www.realtek.com