Datasheet for Telink

Bluetooth LE SoC

TLSR8208

DS-TLSR8208-E13

Ver 1.0.2

2024/04/03

Keyword

Bluetooth LE; 2.4 GHz

Brief

This datasheet is dedicated for Telink Bluetooth LE SoC TLSR8208.

In this datasheet, function block diagram, key features and typical application of the TLSR8208 are introduced.

Published by

Telink Semiconductor

Bldg 3, 1500 Zuchongzhi Rd,

Zhangjiang Hi-Tech Park, Shanghai, China

Legal Disclaimer

This document is provided as-is. Telink Semiconductor reserves the right to make improvements without further notice to this document or any products herein. This document may contain technical inaccuracies or typographical errors. Telink Semiconductor disclaims any and all liability for any errors, inaccuracies or incompleteness contained herein.

Information

For further information on the technology, product and business term, please contact Telink Semiconductor Company (www.telink-semi.com).

For sales or technical support, please send email to the address of:

telinksales@telink-semi.com

telinksupport@telink-semi.com

Revision History

Version	Change Description
0.1.0	Initial release
0.2.0	• Chapter 1 Overview: Updated the description • Section 1.1 Block Diagram: Updated Figure 1-1 • Added 56-pin package (TLSR8208A) related information, including ordering information, package, pin layout, schematic and BOM, changes involving Section 1.4 Ordering Information, Section 1.5 Package, Section Package dimension of TLSR8208C is shown below. • Added Chapter 14 Reference Design
0.5.0	• Added Chapter 2 Memory and MCU, Chapter 3 BLE/2.4 GHz RF Transceiver, Chapter 4 Clock, Chapter 5 Timers, Chapter 6 Interrupt System, Chapter 7 Interface, Chapter 8 PWM, Chapter 9 Keyscan, Chapter 10 Quadrature Decoder, Chapter 11 SAR ADC, Chapter 12 AES, Chapter 13 Key Electrical Specifications
0.5.1	• Section 14.1 Schematic of TLSR8208A: Updated the schematic of TLSR8208A • Section 14.2 BOM (Bill of Material) of TLSR8208A: Updated the bom of TLSR8208A
0.5.2	• Added 24-pin package (TLSR8208B), 40-pin package (TLSR8208C), 16-pin package (TLSR8208D) related information, including ordering information, package, pin layout, schematic and BOM, changes involving Section 1.4 Ordering Information, Section 1.5 Package, Section 1.6.2 Pin Layout for TLSR8208B, Section 1.6.3 Pin Layout for TLSR8208C, Section 1.6.4 Pin Layout for TLSR8208D, Section 14.3 Schematic of TLSR8208B, Section 14.4 BOM (Bill of Material) of TLSR8208B, Section 14.5 Schematic of TLSR8208C, Section 14.6 BOM (Bill of Material) of TLSR8208C. Section 14.7 Schematic of TLSR8208D, Section 14.8 BOM (Bill of Material) of TLSR8208D • Section 1.2.1 General Features: Updated GPIO number • Section 2.1 Memory: Updated the description of FLASH • Section 7.1 GPIO: Updated the first sentence of Section 7.1 describing GPIO number

Version	Change Description
0.5.3	• Section 1.2.3 Power Management Features: Updated the current value • Section 1.2.4 Bluetooth LE Features: Removed mesh support • Section 1.6.4 Pin Layout for TLSR8208D: Updated pin assignment, changes involving Figure 1-9, Table 1-12, Table 1-13 • Section 7.4 SPI: Added the description for SPI • Section 13.1 Absolute Maximum Ratings: Corrected supply voltage • Section 13.2 Recommended Operating Conditions: Updated power-supply voltage conditions • Section 14.1 Schematic of TLSR8208A: Updated the schematic • Section 14.2 BOM (Bill of Material) of TLSR8208A: Updated the BOM • Section 14.7 Schematic of TLSR8208D: Updated the schematic • Section 14.8 BOM (Bill of Material) of TLSR8208D: Updated the BOM
0.5.4	• Section 1.2.3 Power Management Features: Removed the voltage description of deep sleep • Section 1.6.1 Pin Layout for TLSR8208A: Added the pin functions of PE[2], PE[3] in Table 1-7 • Section 7.1.1.1 GPIO Lookup Table: Added the related Information of PE[2], PE[3] in Table 7-1 and Table 7-2 • Section 7.1.3 Connection Relationship Between GPIO and Related Modules: Added the related Information of PE[2], PE[3] in Table 7-3
0.5.5	• Section 1.6 Pin Layout: Added NOTE for SPI master function pins • Section 7.1.1.1 GPIO Lookup Table: Added NOTE for SPI master function pins • Section 14.5 Schematic of TLSR8208C: Updated the schematic • Section 14.6 BOM (Bill of Material) of TLSR8208C: Updated the BOM
0.5.6	• Section 1.5 Package: Corrected package dimension of TLSR8208D, changes involving Figure 1-5 and Table 1-5
0.5.7	• Section 1.2.1 General Features: Updated the descriptions of clock sources and AES • Section 1.4 Ordering Information: Corrected the minimum order quantity of TLSR8208DEE to 5000

Version	Change Description
1.0.0	• Section 1.2.2 RF Features: Updated the description of RF feature • Section 1.2.4 Bluetooth LE Features: Updated the description of Bluetooth feature • Section 1.4 Ordering Information: Added SRAM and flash size to Table 1-1 • Section 3.1 Block Diagram: Updated the description of the BLE mode • Section 3.3 Baseband: Revised the payload to 1 ~ 63 bytes in Table 3-3, removed BLE location function • Section 4.1 Clock Sources: Updated Figure 4-1 Block Diagram of Clock • Section 7.1.2 GPIO Logic Introduction: Added this section to describe GPIO logic • Section 7.1.4 Pull-Up/Pull-Down Resistor: Revised the pull-up/pull-down resistor typical value in Table 7-4 • Section 13.4 AC Characteristics: Updated Table 13-5 RF Performance Characteristics
1.0.1	• Section 1.6 Pin Layout: Added PWM2 to the functions included in "All functions" • Section 2.5.3 LDO: Added the diagram of LDO module • Section 2.5.4 VBAT and VANT Power-Supply Mode: Updated the description of VBAT and VANT power-supply mode • Section 7.1.1 Basic Configuration: Added PWM2 to the functions included in "All functions" • Section 7.5 UART: Updated the description of UART • Chapter 14 Reference Design: Updated the schematics and BOMs for TLSR8208A, TLSR8208B, TLSR8208C and TLSR8208D
1.0.2	• Section 1.1 Block Diagram: Removed ECC in Figure 1-1 • Chapter 14 Reference Design: Updated the schematics and BOMs for TLSR8208A, TLSR8208B, TLSR8208C and TLSR8208D

Table of Contents

List of Figures

List of Tables

1 Overview

The TLSR8208 is a Telink-developed Bluetooth low energy (LE) and 2.4 GHz multi-standard wireless SoC solution. The embedded 2.4 GHz transceiver supports Bluetooth low energy as well as 2.4 GHz operation. It's completely RoHS-compliant and 100% lead (Pb)-free.

The TLSR8208 combines the radio frequency (RF), digital processing, protocols stack software and profiles for multiple standards into a single SoC. The chip supports standards and industrial alliance specifications including Bluetooth low energy and 2.4 GHz proprietary standards. The TLSR8208 has hardware OTA upgrades support and multiple boot switching, allowing convenient product feature roll outs and upgrades.

The TLSR8208 supports concurrent multi-standards. For some use cases, the TLSR8208 can "concurrently" run two standards, for example, stacks such as Bluetooth LE and 2.4G can run concurrently with one application state but dual radio communication channels for interacting with different devices. The end product working in this mode can maintain active Bluetooth Smart connections to smart phones or other Bluetooth LE devices while control and communicate with other 2.4 GHz devices at the same time. In this case, it's compatible with Bluetooth standard, supports Bluetooth LE specification, allows easy connectivity with Bluetooth Smart Ready mobile phones, tablets, laptops, which supports Bluetooth LE slave and master mode operation, including broadcast, encryption, connection updates, and channel map updates. At the same time, it also supports 2.4G standard, and is perfect for creating interoperable solution for use within the home combined with leading 2.4G software stack. This feature enables products to bridge the smartphone and home automation world with a single chip and no requirement for an external hub.

The TLSR8208 integrates hardware acceleration to support the complicated security operations required by Bluetooth, without the requirement for an external DSP, thereby significantly reducing the product eBOM.

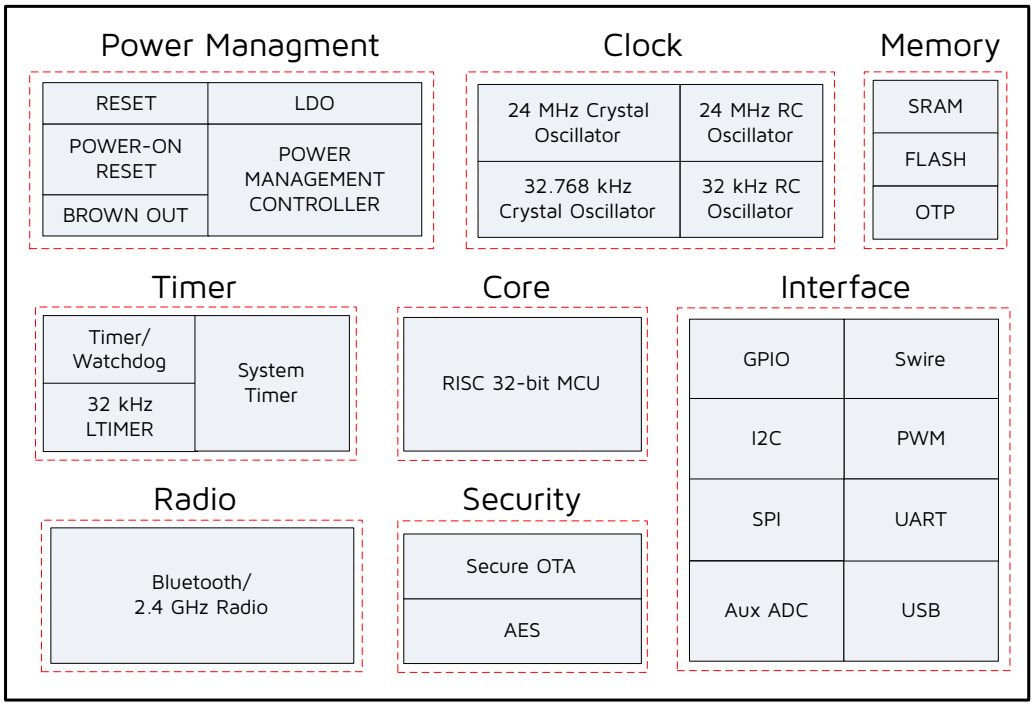
The TLSR8208 includes a full range of on-chip peripherals for interfacing with external components such as LEDs, sensors, touch controllers, keyboards, and motors. This makes it an solution for low cost IoT (Internet of Things) and 2.4 Ghz devices.

The TLSR8208 series is compliant with worldwide radio frequency regulations, including ETSI EN 300 328 and EN 300 440 Class 2 (Europe), FCC CFR47 Part 15 (US) and ARIB STD-T66 (Japan).

1.1 Block Diagram

The TLSR8208 is designed to offer high integration, ultra-low power application capabilities. The system's block diagram is as shown in Figure 1-1.

Figure 1-1 Block Diagram of the System



The TLSR8208 integrates a powerful 32-bit MCU, 2.4 GHz ISM radio, 16 KB OTP, 16 kB retention SRAM, 128 KB flash (TLSR8208B/D), 512 KB flash (TLSR8208C), or external Flash (TLSR8208A), 14-bit Aux ADC, PWM, flexible IO interfaces, and other peripheral blocks required for IoT (Internet of Things) and HID (Human Interface Devices) application development.

The TLSR8208 also includes multi-stage power management design allowing ultra-low power operation and making it the ideal candidate for power-constraint applications.

With the high integration level of the TLSR8208, few external components are needed to satisfy customers' complicated application requirements.

1.2 Key Features

1.2.1 General Features

General features are as follows:

1. 32-bit proprietary microcontroller
   - Maximum running speed up to 48 MHz
2. Memory architecture
   - Program memory: 128 KB flash (TLSR8208B/D), 512 KB flash (TLSR8208C) or external Flash
   - 16 KB retention SRAM
   - Support 16 KB OTP
3. RTC and other timers

- Clock source of 24 MHz & 32.768 kHz Crystal and 32 kHz / 24 MHz embedded RC oscillator, among which the external 24 MHz crystal is to calibrate internal 32 kHz clock, the internal 32 kHz oscillator is for low precision application, the external 32.768 kHz crystal is for high precision application
- Three general 32-bit timers with four selectable modes in active mode
- Watchdog timer
- A low-frequency 32 kHz timer available in low power mode

4. A rich set of digital and analog interfaces
   - Up to 38/16/33/10 GPIOs depending on package option. All digital IOs can be used as GPIOS
   - SPI
   - I2C
   - USB 2.0
   - UART with hardware flow control and 7816 protocol support
   - Swire debug interface
   - Up to 6 channels of differential PWM
   - IR transmitter with DMA
   - One quadrature decoder (QDEC), two-phase input selectable
   - 14-bit auxiliary ADC
5. Embedded hardware AES block cipher with 128 bit keys and software AES CCM
6. Hardware OTA upgrade and multiple boot switch, allowing convenient product feature roll outs and upgrades
7. Support keyboard scan function, maximum matrix is 8*18
8. Operating temperature range: -40°C ~ + 85°C
9. Completely RoHS-compliant package
   - TLSR8208A, 56-pin QFN 7×7 mm
   - TLSR8208B, 24-pin QFN 4x4 mm
   - TLSR8208C, 40-pin QFN 5x5 mm
   - TLSR8208D, 16-pin SOP 9.9x6 mm

1.2.2 RF Features

RF features include:

1. Bluetooth/2.4 GHz RF transceiver in worldwide 2.4 GHz ISM band
2. Bluetooth LE 1 Mbps and 2 Mbps
3. 2.4 GHz proprietary 1 Mbps/2 Mbps/250 kbps/500 kbps mode
4. RX sensitivity: -97 dBm @ Bluetooth LE 1 Mbps mode, -93 dBm @ Bluetooth LE 2 Mbps mode
5. TX output power: -45 to +10 dBm
6. 50 Ω matched single-pin antenna input
7. RSSI monitoring with +/-1 dB resolution
8. Auto acknowledgement, retransmission and flow control
9. Support PTA (Packet Traffic Arbitrator) for Wi-Fi co-existence

1.2.3 Power Management Features

Features of power management module include:

1. Embedded LDO
2. Battery monitor: Support low battery detection
3. Power supply:
   - VDD: 1.8 V ~ 3.6 V
   - VBAT: 1.8 V ~ 4.2 V
   - VBUS: 4.5 V ~ 5.5 V
4. Multiple stage power management to minimize power consumption
   - RF/Digit core working at 1.2 V
5. Low power consumption:
   - Whole chip RX mode: 9.1 mA with LDO
   - Whole chip TX mode @ 0 dBm: 9.5 mA with LDO
   - Deep sleep with external wakeup (without SRAM retention): 0.55 μA
   - Deep sleep with 16 KB SRAM retention: 0.9 μA
   - Deep sleep with external wakeup, with 32K RC oscillator on (without SRAM retention): 1.0 μA
   - Deep sleep with 16 KB SRAM retention, with 32K RC oscillator on: 1.3 μA

1.2.4 Bluetooth LE Features

Bluetooth LE features include:

1. Qualified for Bluetooth 5.3, main features supported include Bluetooth LE 1 Mbps and 2 Mbps

1.2.5 Concurrent Mode Feature

In concurrent mode, the chip supports multiple standard working concurrently. A typical combination: BLE and 2.4G based stacks can run concurrently with one application state but dual radio communication channels for interacting with different devices, BLE and 2.4G concurrent operation.

1.3 Typical Applications

The TLSR8208 can be applied to IoT (Internet of Things) and HID (Human Interface Devices) applications, such as Bluetooth LE smart devices, home automation devices. Its typical applications include, but are not limited to the following:

- ESL (Electronic Shelf Label) devices
- Smart lighting, smart home devices
- RF remote control
- Smartphone and tablet accessories
- Sports and fitness tracking
- Wearable devices
- Wireless toys

- Building automation
- Smart grid
- Intelligent logistics/transportation/city
- Industrial control
- Health care

1.4 Ordering Information

Table 1-1 Ordering Information of TLSR8208

Product Series	Ordering No.	Package Type	SRAM Size	Flash Size	Temperature Range	Packing Method	Minimum Order Quantity
TLSR8208	TLSR8208BER	24-pin QFN 4x4x0.75 mm	16 KB	128 KB	-40°C ~ +85°C	TR[a]	3000
	TLSR8208CER	40-pin QFN 5x5x0.75 mm	16 KB	512 KB	-40°C ~ +85°C	TR	3000
	TLSR8208DEE	16-pin SOP 9.9x6x1.75 mm	16 KB	128 KB	-40°C ~ +85°C	Tube	5000

a. Packing method "TR" means tape and reel. The tape and reel material DO NOT support baking under high temperature.

1.5 Package

Package dimension of TLSR8208A is shown below.

Figure 1-2 Package of TLSR8208A

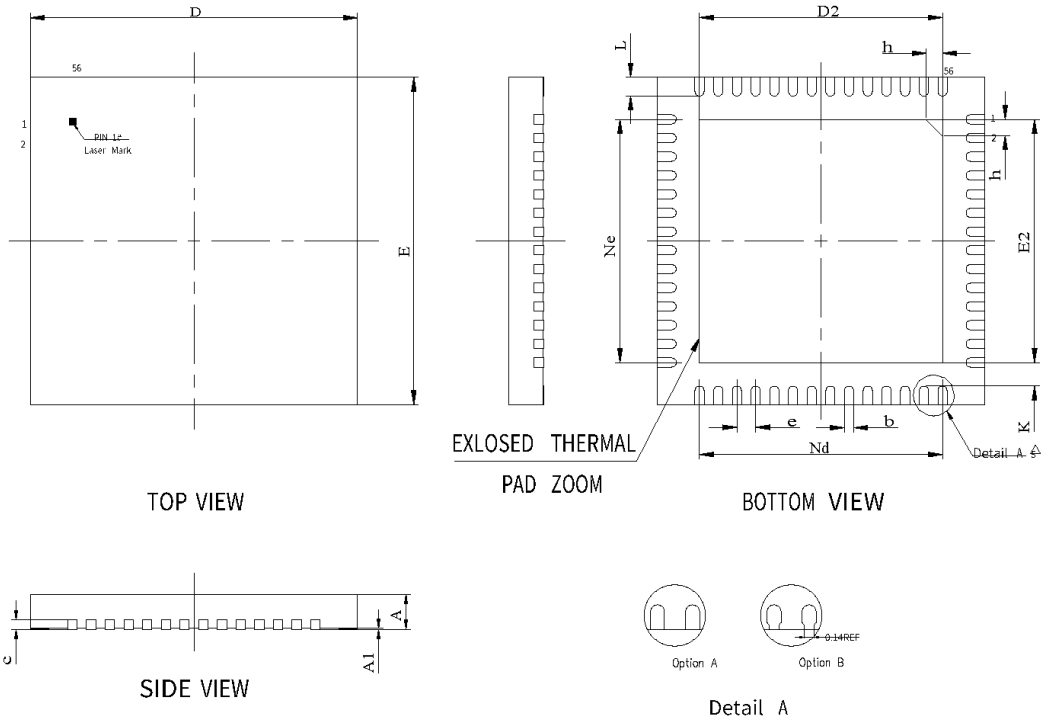


Table 1-2 Mechanical Dimensions of TLSR8208A

Symbol	Millimeter		
	Min	Nom	Max
A	0.70	0.75	0.80
A1	-	0.02	0.05
b	0.15	0.20	0.25
c	0.18	0.20	0.25
D	6.90	7.00	7.10
D2	5.10	5.20	5.30
e	0.40 BSC		
Nd	5.20 BSC		
Ne	5.20 BSC		
E	6.90	7.00	7.10
E2	5.10	5.20	5.30
K	0.20	-	-
L	0.35	0.40	0.45
h	0.30	0.35	0.40

Package dimension of TLSR8208B is shown below.

Figure 1-3 Package of TLSR8208B

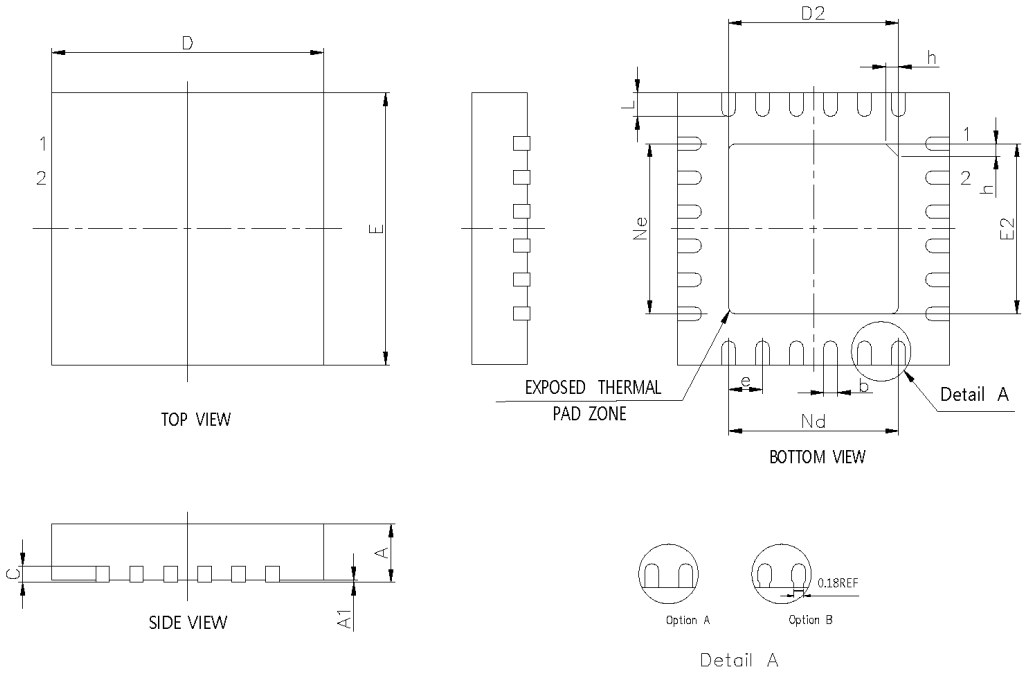


Table 1-3 Mechanical Dimensions of TLSR8208B

Symbol	Millimeter		
	Min	Nom	Max
A	0.70	0.75	0.80
A1	-	0.02	0.05
b	0.18	0.25	0.30
c	0.18	0.20	0.25
D	3.90	4.00	4.10
D2	2.40	2.50	2.60
e	0.50 BSC		
Nd	2.50 BSC		
Ne	2.50 BSC		
E	3.90	4.00	4.10
E2	2.40	2.50	2.60
L	0.35	0.40	0.45
h	0.30	0.35	0.40

Package dimension of TLSR8208C is shown below.

Figure 1-4 Package of TLSR8208C

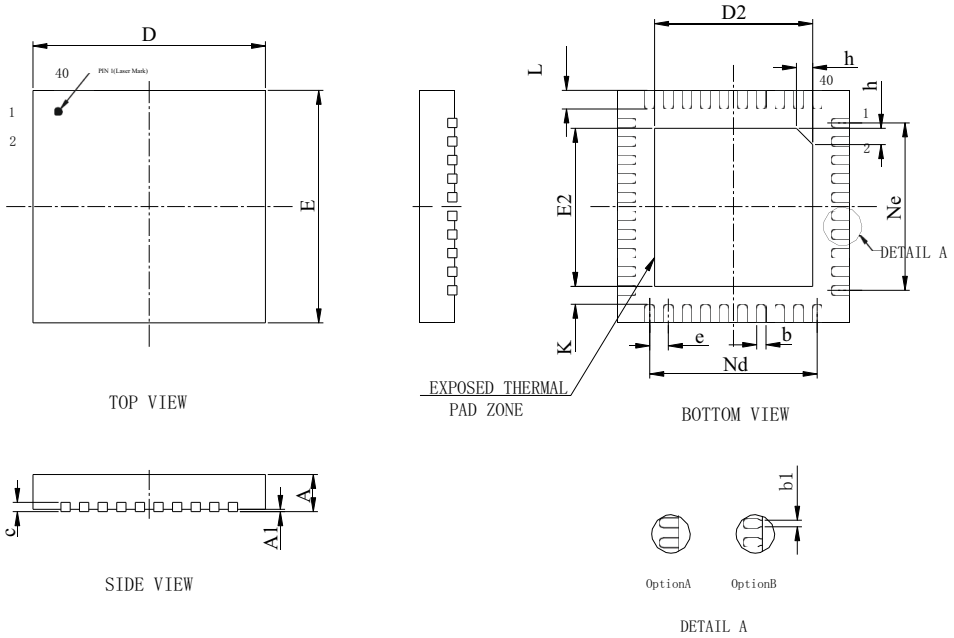


Table 1-4 Mechanical Dimensions of TLSR8208C

Symbol	Millimeter		
	Min	Nom	Max
A	0.70	0.75	0.80
A1	-	0.02	0.05
b	0.15	0.20	0.25
b1	0.14REF		
c	0.18	0.20	0.25
D	4.90	5.00	5.10
D2	3.30	3.40	3.50
e	0.40 BSC		
Nd	3.60 BSC		
Ne	3.60 BSC		
E	4.90	5.00	5.10
E2	3.30	3.40	3.50
L	0.35	0.40	0.45
K	0.20	-	-

Symbol	Millimeter		
	Min	Nom	Max
h	0.30	0.35	0.40

Package dimension of TLSR8208D is shown below.

Figure 1-5 Package of TLSR8208D

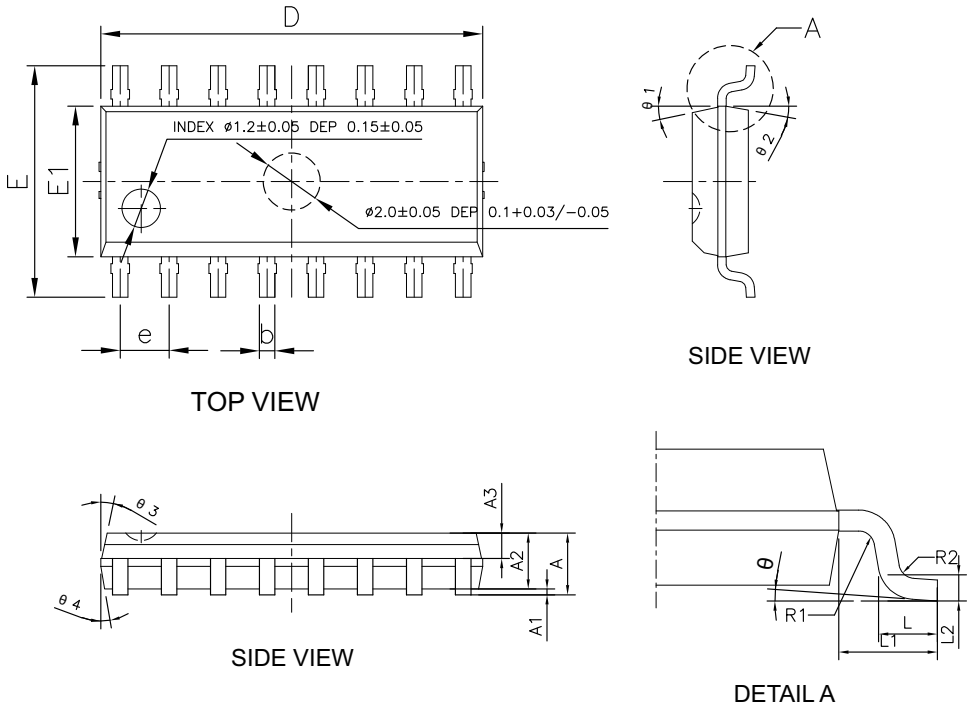


Table 1-5 Mechanical Dimensions of TLSR8208D

Symbol	Millimeter		
	Min	Nom	Max
A	-	-	1.70
A1	0.10	0.15	0.20
A2	1.42	1.45	1.48
A3	0.62	0.65	0.68
b	0.38	-	0.51
D	9.85	9.90	9.95
E	5.90	6.00	6.10
E1	3.87	3.90	3.93
e	1.24	1.27	1.30

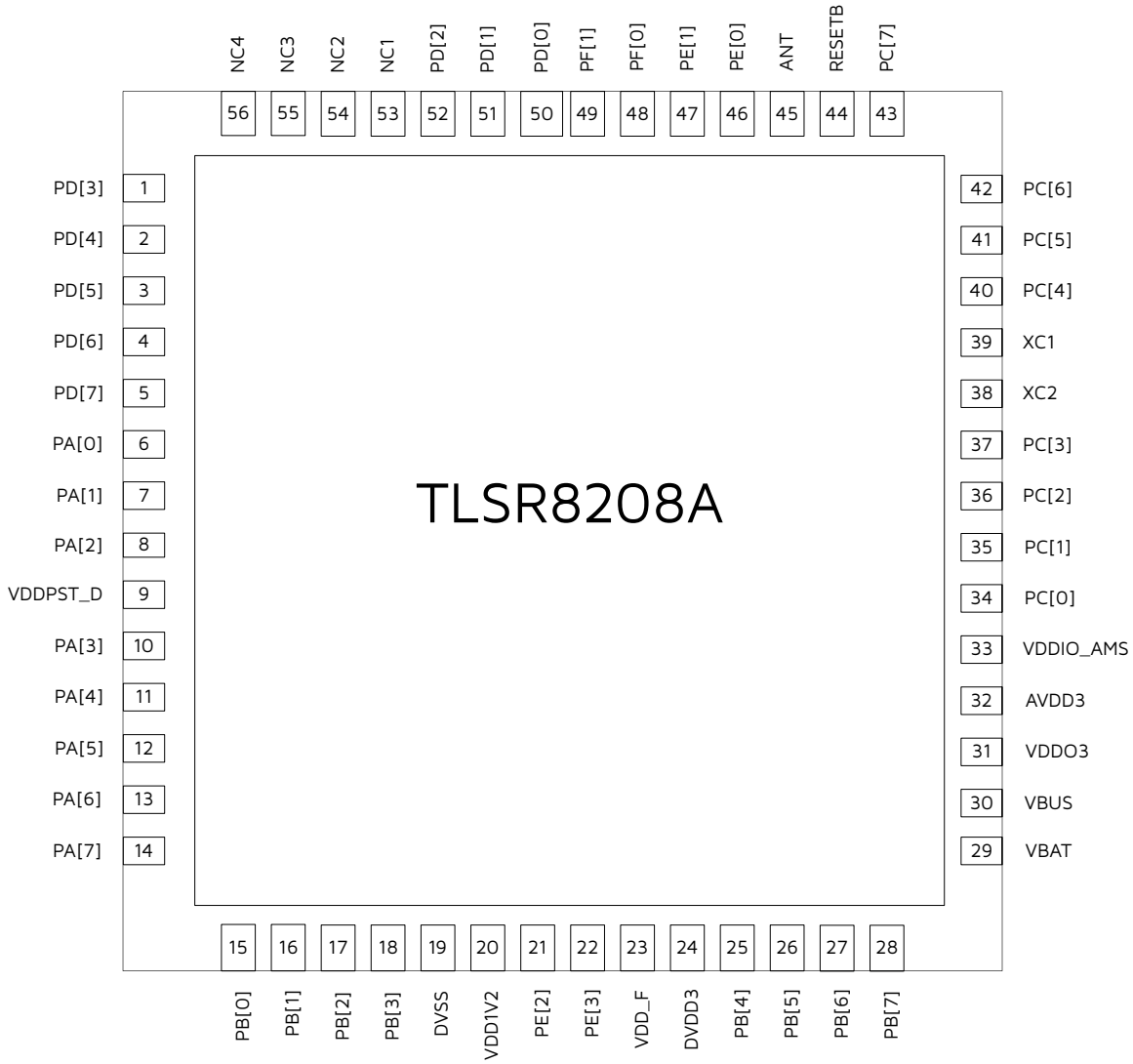
Symbol	Millimeter		
	Min	Nom	Max
L	0.50	0.60	0.70
L1	1.05REF		
L2	0.25REF		
θ	0°	-	8°
θ1 - θ4	12°REF		
R1	0.15REF		
R2	0.15REF		

1.6 Pin Layout

1.6.1 Pin Layout for TLSR8208A

Pin assignment of TLSR8208A is shown below.

Figure 1-6 Pin Assignment of TLSR8208A



Functions of the 56 pins of TLSR8208A are shown in the table below.

Table 1-6 Pin Function of TLSR8208A

No.	Pin Name	Type	Description
1	PD[3]	GPIO	GPIO PD[3], refer to Table 1-7 for pin mux function.
2	PD[4]	GPIO	GPIO PD[4], refer to Table 1-7 for pin mux function.
3	PD[5]	GPIO	GPIO PD[5], refer to Table 1-7 for pin mux function.
4	PD[6]	GPIO	GPIO PD[6], refer to Table 1-7 for pin mux function.
5	PD[7]	GPIO	GPIO PD[7], refer to Table 1-7 for pin mux function.
6	PA[0]	GPIO	GPIO PA[0], refer to Table 1-7 for pin mux function.
7	PA[1]	GPIO	GPIO PA[1], refer to Table 1-7 for pin mux function.

No.	Pin Name	Type	Description
8	PA[2]	GPIO	GPIO PA[2], refer to Table 1-7 for pin mux function.
9	VDDPST_D	PWR	Power supply.
10	PA[3]	GPIO	GPIO PA[3], refer to Table 1-7 for pin mux function.
11	PA[4]	GPIO	GPIO PA[4], refer to Table 1-7 for pin mux function.
12	PA[5]	GPIO	GPIO PA[5], refer to Table 1-7 for pin mux function.
13	PA[6]	GPIO	GPIO PA[6], refer to Table 1-7 for pin mux function.
14	PA[7]	GPIO	GPIO PA[7], refer to Table 1-7 for pin mux function.
15	PB[0]	GPIO	GPIO PB[0], refer to Table 1-7 for pin mux function.
16	PB[1]	GPIO	GPIO PB[1], refer to Table 1-7 for pin mux function.
17	PB[2]	GPIO	GPIO PB[2], refer to Table 1-7 for pin mux function.
18	PB[3]	GPIO	GPIO PB[3], refer to Table 1-7 for pin mux function.
19	DVSS	GND	Digital Ground.
20	VDD1V2	PWR	Digital core supply.
21	PE[2]	GPIO	GPIO PE[2], refer to Table 1-7 for pin mux function.
22	PE[3]	GPIO	GPIO PE[3], refer to Table 1-7 for pin mux function.
23	VDD_F	PWR	Internally generated power supply to flash. Connect to GND via external capacitor.
24	DVDD3	PWR	Power supply input as AA battery application, output as Li/USB application.
25	PB[4]	GPIO	GPIO PB[4], refer to Table 1-7 for pin mux function.
26	PB[5]	GPIO	GPIO PB[5], refer to Table 1-7 for pin mux function.
27	PB[6]	GPIO	GPIO PB[6], refer to Table 1-7 for pin mux function.
28	PB[7]	GPIO	GPIO PB[7], refer to Table 1-7 for pin mux function.
29	VBAT	PWR	Lion-Battery power supply.
30	VBUS	PWR	5V VBUS power supply of USB.
31	VDDO3	PWR	Power supply.
32	AVDD3	PWR	Power supply.
33	VDDIO_AMS	PWR	Power supply.
34	PC[0]	GPIO	GPIO PC[0], refer to Table 1-7 for pin mux function.
35	PC[1]	GPIO	GPIO PC[1], refer to Table 1-7 for pin mux function.

No.	Pin Name	Type	Description
36	PC[2]	GPIO	GPIO PC[2], refer to Table 1-7 for pin mux function.
37	PC[3]	GPIO	GPIO PC[3], refer to Table 1-7 for pin mux function.
38	XC2	Analog	Crystal oscillator pin.
39	XC1	Analog	Crystal oscillator pin.
40	PC[4]	GPIO	GPIO PC[4], refer to Table 1-7 for pin mux function.
41	PC[5]	GPIO	GPIO PC[5], refer to Table 1-7 for pin mux function.
42	PC[6]	GPIO	GPIO PC[6], refer to Table 1-7 for pin mux function.
43	PC[7]	GPIO	GPIO PC[7], refer to Table 1-7 for pin mux function.
44	RESETB	Reset	Power on reset, active low.
45	ANT	Analog	Pin to connect to the Antenna through the matching network.
46	PE[0]	GPIO	GPIO PE[0], refer to Table 1-7 for pin mux function.
47	PE[1]	GPIO	GPIO PE[1], refer to Table 1-7 for pin mux function.
48	PF[0]	GPIO	GPIO PF[0], refer to Table 1-7 for pin mux function.
49	PF[1]	GPIO	GPIO PF[1], refer to Table 1-7 for pin mux function.
50	PD[0]	GPIO	GPIO PD[0], refer to Table 1-7 for pin mux function.
51	PD[1]	GPIO	GPIO PD[1], refer to Table 1-7 for pin mux function.
52	PD[2]	GPIO	GPIO PD[2], refer to Table 1-7 for pin mux function.
53	NC1	-	-
54	NC2	-	-
55	NC3	-	-
56	NC4	-	-

GPIO pin mux functions of TLSR8208A are shown in the table below.

Table 1-7 GPIO Pin Mux of TLSR8208A

Pad	Default	Func1	Func2	Func3	Analog Func
PA[0]	GPIO	All functions[a]	PA_KS0_IO	UART_CTS_I	-
PA[1]	DM	-	UART_RX_I	DM_IO	-
PA[2]	DP(SWS)	-	UART_TX	DP_IO	-
PA[3]	SWS	-	-	SWS_IO	sar_aio<9>

Pad	Default	Func1	Func2	Func3	Analog Func
PA[4]	GPIO	All functions	PA_KS4_IO	SWM_IO	-
PA[5]	GPIO	All functions	PA_KS5_IO	UART_RTS	-
PA[6]	GPIO	All functions	PA_KS6_IO	UART_TX	-
PA[7]	GPIO	All functions	PA_KS7_IO	UART_RX_I	-
PB[0]	SPI_CN	-	PB_KS0_IO	SPI_CN_IO	sar_aio<0>
PB[1]	SPI_CK	-	PB_KS1_IO	SPI_CK_IO	sar_aio<1>
PB[2]	GPIO	All functions	PB_KS2_IO	PWM0	sar_aio<2>
PB[3]	SPI_IO2	-	PB_KS3_IO	SPI_IO2_IO	sar_aio<3>
PB[4]	GPIO	All functions	PB_KS4_IO	PWM0_N	sar_aio<4>
PB[5]	GPIO	All functions	PB_KS5_IO	PWM1	sar_aio<5>
PB[6]	GPIO	All functions	PB_KS6_IO	PWM2	sar_aio<6>
PB[7]	GPIO	All functions	PB_KS7_IO	PWM3	sar_aio<7>
PC[0]	GPIO	All functions	PC_KS0_IO	ANT_SEL0	-
PC[1]	GPIO	All functions	PC_KS1_IO	ANT_SEL1	-
PC[2]	GPIO	All functions	PC_KS2_IO	ANT_SEL2	xtl_32k_out/diag_hv_ana/ atb_p_sar
PC[3]	GPIO	All functions	PC_KS3_IO	BLE_ACTIVITY	xtl_32k_in/atb_n_sar
PC[4]	GPIO	All functions	PC_KS4_IO	BLE_STATUS	sar_aio<8>
PC[5]	GPIO	All functions	PC_KS5_IO	WIFI_DENY_I	-
PC[6]	GPIO	All functions	PC_KS6_IO	RX_CYC2LNA	-
PC[7]	GPIO	All functions	PC_KS7_IO	TX_CYC2PA	-
PD[0]	GPIO	All functions	PD_KS0_IO	I2C_SCL_IO	-
PD[1]	GPIO	All functions	PD_KS1_IO	I2C_SDA_IO	-
PD[2]	GPIO	All functions	PD_KS2_IO	PWM4	-
PD[3]	GPIO	All functions	PD_KS3_IO	PWM5	-
PD[4]	SPI_IO3	-	PD_KS4_IO	SPI_IO3_IO	-
PD[5]	GPIO	All functions	PD_KS5_IO	PWM1_N	-
PD[6]	GPIO	All functions	PD_KS6_IO	CLK_7816	-

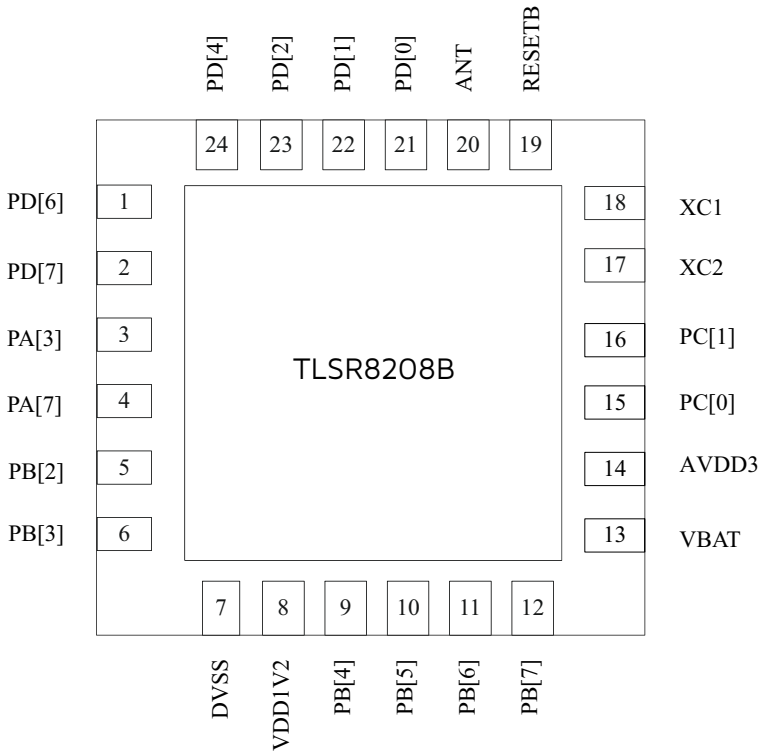
Pad	Default	Func1	Func2	Func3	Analog Func
PD[7]	GPIO	All functions	PD_KS7_IO	UART_RTX_IO	-
PE[0]	MSDO	-	-	MOSI_IO	-
PE[1]	MCLK	-	-	MCLK	-
PE[2]	MSCN	-	-	MSCN	-
PE[3]	MSDI	-	-	MISO_IO	-
PF[0]	SPI_MOSI	-	PA_KS1_IO	SPI_MOSI_IO	-
PF[1]	SPI_MISO	-	PA_KS2_IO	SPI_MISO_IO	-

a. "All functions" include 32 functions: WIFI_DENY_I, BLE_STATUS, BLE_ACTIVITY, SPI_CN_IO, SPI_CK_IO, SPI_MOSI_IO, SPI_MISO_IO, SWM_IO, TX_CYC2PA, RX_CYC2LNA, ANT_SEL2, ANT_SEL1, ANT_SEL0, UART_RTX_IO, CLK_7816, I2C_SDA_IO, I2C_SCL_IO, UART_RX_I, UART_TX, UART_RTS, UART_CTS_I, PWM5_N, PWM4_N, PWM3_N, PWM2_N, PWM1_N, PWM0_N, PWM5, PWM4, PWM3, PWM2, PWM1, PWM0

1.6.2 Pin Layout for TLSR8208B

Pin assignment of TLSR8208B is shown below.

Figure 1-7 Pin Assignment of TLSR8208B



Functions of the 24 pins of TLSR8208B are shown in the table below.

Table 1-8 Pin Function of TLSR8208B

No.	Pin Name	Type	Description
1	PD[6]	GPIO	GPIO PD[6], refer to Table 1-9 for pin mux function.
2	PD[7]	GPIO	GPIO PD[7], refer to Table 1-9 for pin mux function.
3	PA[3]	GPIO	GPIO PA[3], refer to Table 1-9 for pin mux function.
4	PA[7]	GPIO	GPIO PA[7], refer to Table 1-9 for pin mux function.
5	PB[2]	GPIO	GPIO PB[2], refer to Table 1-9 for pin mux function.
6	PB[3]	GPIO	GPIO PB[3], refer to Table 1-9 for pin mux function.
7	DVSS	GND	Digital Ground.
8	VDD1V2	PWR	Digital core supply.
9	PB[4]	GPIO	GPIO PB[4], refer to Table 1-9 for pin mux function.
10	PB[5]	GPIO	GPIO PB[5], refer to Table 1-9 for pin mux function.
11	PB[6]	GPIO	GPIO PB[6], refer to Table 1-9 for pin mux function.
12	PB[7]	GPIO	GPIO PB[7], refer to Table 1-9 for pin mux function.
13	VBAT	PWR	Lion-Battery power supply.
14	AVDD3	PWR	Power supply.
15	PC[0]	GPIO	GPIO PC[0], refer to Table 1-9 for pin mux function.
16	PC[1]	GPIO	GPIO PC[1], refer to Table 1-9 for pin mux function.
17	XC2	Analog	Crystal oscillator pin.
18	XC1	Analog	Crystal oscillator pin.
19	RESETB	Reset	Power on reset, active low.
20	ANT	Analog	Pin to connect to the Antenna through the matching network.
21	PD[0]	GPIO	GPIO PD[0], refer to Table 1-9 for pin mux function.
22	PD[1]	GPIO	GPIO PD[1], refer to Table 1-9 for pin mux function.
23	PD[2]	GPIO	GPIO PD[2], refer to Table 1-9 for pin mux function.
24	PD[4]	GPIO	GPIO PD[4], refer to Table 1-9 for pin mux function.

GPIO pin mux functions of TLSR8208B are shown in the table below.

Table 1-9 GPIO Pin Mux of TLSR8208B

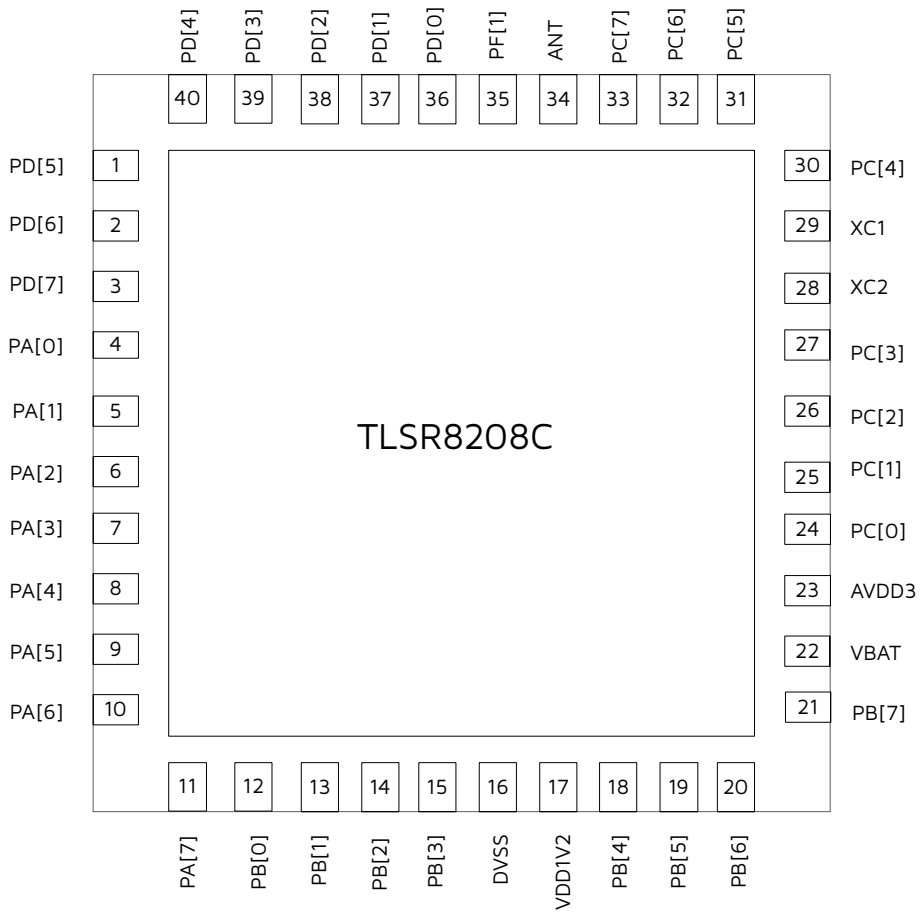
Pad	Default	Func1	Func2	Func3	Analog Func
PA[3]	SWS	-	-	SWS_IO	sar_aio<9>
PA[7]	GPIO	All functions[a]	PA_KS7_IO	UART_RX_I	-
PB[2]	GPIO	All functions	PB_KS2_IO	PWM0	sar_aio<2>
PB[3]	SPI_IO2	-	PB_KS3_IO	SPI_IO2_IO	sar_aio<3>
PB[4]	GPIO	All functions	PB_KS4_IO	PWM0_N	sar_aio<4>
PB[5]	GPIO	All functions	PB_KS5_IO	PWM1	sar_aio<5>
PB[6]	GPIO	All functions	PB_KS6_IO	PWM2	sar_aio<6>
PB[7]	GPIO	All functions	PB_KS7_IO	PWM3	sar_aio<7>
PC[0]	GPIO	All functions	PC_KS0_IO	ANT_SEL0	-
PC[1]	GPIO	All functions	PC_KS1_IO	ANT_SEL1	-
PD[0]	GPIO	All functions	PD_KS0_IO	I2C_SCL_IO	-
PD[1]	GPIO	All functions	PD_KS1_IO	I2C_SDA_IO	-
PD[2]	GPIO	All functions	PD_KS2_IO	PWM4	-
PD[4]	SPI_IO3	-	PD_KS4_IO	SPI_IO3_IO	-
PD[6]	GPIO	All functions	PD_KS6_IO	CLK_7816	-
PD[7]	GPIO	All functions	PD_KS7_IO	UART_RTX_IO	-

a. "All functions" include 32 functions: WIFI_DENY_I, BLE_STATUS, BLE_ACTIVITY, SPI_CN_IO, SPI_CK_IO, SPI_MOSI_IO, SPI_MISO_IO, SWM_IO, TX_CYC2PA, RX_CYC2LNA, ANT_SEL2, ANT_SEL1, ANT_SEL0, UART_RTX_IO, CLK_7816, I2C_SDA_IO, I2C_SCL_IO, UART_RX_I, UART_TX, UART_RTS, UART_CTS_I, PWM5_N, PWM4_N, PWM3_N, PWM2_N, PWM1_N, PWM0_N, PWM5, PWM4, PWM3, PWM2, PWM1, PWM0

1.6.3 Pin Layout for TLSR8208C

Pin assignment of TLSR8208C is shown below.

Figure 1-8 Pin Assignment of TLSR8208C



Functions of the 40 pins of TLSR8208C are shown in the table below.

Table 1-10 Pin Function of TLSR8208C

No.	Pin Name	Type	Description
1	PD[5]	GPIO	GPIO PD[5], refer to Table 1-11 for pin mux function.
2	PD[6]	GPIO	GPIO PD[6], refer to Table 1-11 for pin mux function.
3	PD[7]	GPIO	GPIO PD[7], refer to Table 1-11 for pin mux function.
4	PA[0]	GPIO	GPIO PA[0], refer to Table 1-11 for pin mux function.
5	PA[1]	GPIO	GPIO PA[1], refer to Table 1-11 for pin mux function.
6	PA[2]	GPIO	GPIO PA[2], refer to Table 1-11 for pin mux function.
7	PA[3]	GPIO	GPIO PA[3], refer to Table 1-11 for pin mux function.
8	PA[4]	GPIO	GPIO PA[4], refer to Table 1-11 for pin mux function.
9	PA[5]	GPIO	GPIO PA[5], refer to Table 1-11 for pin mux function.

No.	Pin Name	Type	Description
10	PA[6]	GPIO	GPIO PA[6], refer to Table 1-11 for pin mux function.
11	PA[7]	GPIO	GPIO PA[7], refer to Table 1-11 for pin mux function.
12	PB[0]	GPIO	GPIO PB[0], refer to Table 1-11 for pin mux function.
13	PB[1]	GPIO	GPIO PB[1], refer to Table 1-11 for pin mux function.
14	PB[2]	GPIO	GPIO PB[2], refer to Table 1-11 for pin mux function.
15	PB[3]	GPIO	GPIO PB[3], refer to Table 1-11 for pin mux function.
16	DVSS	GND	Digital Ground.
17	VDD1V2	PWR	Digital core supply.
18	PB[4]	GPIO	GPIO PB[4], refer to Table 1-11 for pin mux function.
19	PB[5]	GPIO	GPIO PB[5], refer to Table 1-11 for pin mux function.
20	PB[6]	GPIO	GPIO PB[6], refer to Table 1-11 for pin mux function.
21	PB[7]	GPIO	GPIO PB[7], refer to Table 1-11 for pin mux function.
22	VBAT	PWR	Lion-Battery power supply.
23	AVDD3	PWR	Power supply.
24	PC[0]	GPIO	GPIO PC[0], refer to Table 1-11 for pin mux function.
25	PC[1]	GPIO	GPIO PC[1], refer to Table 1-11 for pin mux function.
26	PC[2]	GPIO	GPIO PC[2], refer to Table 1-11 for pin mux function.
27	PC[3]	GPIO	GPIO PC[3], refer to Table 1-11 for pin mux function.
28	XC2	Analog	Crystal oscillator pin.
29	XC1	Analog	Crystal oscillator pin.
30	PC[4]	GPIO	GPIO PC[4], refer to Table 1-11 for pin mux function.
31	PC[5]	GPIO	GPIO PC[5], refer to Table 1-11 for pin mux function.
32	PC[6]	GPIO	GPIO PC[6], refer to Table 1-11 for pin mux function.
33	PC[7]	GPIO	GPIO PC[7], refer to Table 1-11 for pin mux function.
34	ANT	Analog	Pin to connect to the Antenna through the matching network.
35	PF[1]	GPIO	GPIO PF[1], refer to Table 1-11 for pin mux function.
36	PD[0]	GPIO	GPIO PD[0], refer to Table 1-11 for pin mux function.
37	PD[1]	GPIO	GPIO PD[1], refer to Table 1-11 for pin mux function.

No.	Pin Name	Type	Description
38	PD[2]	GPIO	GPIO PD[2], refer to Table 1-11 for pin mux function.
39	PD[3]	GPIO	GPIO PD[3], refer to Table 1-11 for pin mux function.
40	PD[4]	GPIO	GPIO PD[4], refer to Table 1-11 for pin mux function.

GPIO pin mux functions of TLSR8208C are shown in the table below.

Table 1-11 GPIO Pin Mux of TLSR8208C

Pad	Default	Func1	Func2	Func3	Analog Func
PA[0]	GPIO	All functions[a]	PA_KS0_IO	UART_CTS_I	-
PA[1]	DM	-	UART_RX_I	DM_IO	-
PA[2]	DP(SWS)	-	UART_TX	DP_IO	-
PA[3]	SWS	-	-	SWS_IO	sar_aio<9>
PA[4]	GPIO	All functions	PA_KS4_IO	SWM_IO	-
PA[5]	GPIO	All functions	PA_KS5_IO	UART_RTS	-
PA[6]	GPIO	All functions	PA_KS6_IO	UART_TX	-
PA[7]	GPIO	All functions	PA_KS7_IO	UART_RX_I	-
PB[0]	SPI_CN	-	PB_KS0_IO	SPI_CN_IO	sar_aio<0>
PB[1]	SPI_CK	-	PB_KS1_IO	SPI_CK_IO	sar_aio<1>
PB[2]	GPIO	All functions	PB_KS2_IO	PWM0	sar_aio<2>
PB[3]	SPI_IO2	-	PB_KS3_IO	SPI_IO2_IO	sar_aio<3>
PB[4]	GPIO	All functions	PB_KS4_IO	PWM0_N	sar_aio<4>
PB[5]	GPIO	All functions	PB_KS5_IO	PWM1	sar_aio<5>
PB[6]	GPIO	All functions	PB_KS6_IO	PWM2	sar_aio<6>
PB[7]	GPIO	All functions	PB_KS7_IO	PWM3	sar_aio<7>
PC[0]	GPIO	All functions	PC_KS0_IO	ANT_SEL0	-
PC[1]	GPIO	All functions	PC_KS1_IO	ANT_SEL1	-
PC[2]	GPIO	All functions	PC_KS2_IO	ANT_SEL2	xtl_32k_out/diag_hv_ana/ atb_p_sar
PC[3]	GPIO	All functions	PC_KS3_IO	BLE_ACTIVITY	xtl_32k_in/atb_n_sar
PC[4]	GPIO	All functions	PC_KS4_IO	BLE_STATUS	sar_aio<8>

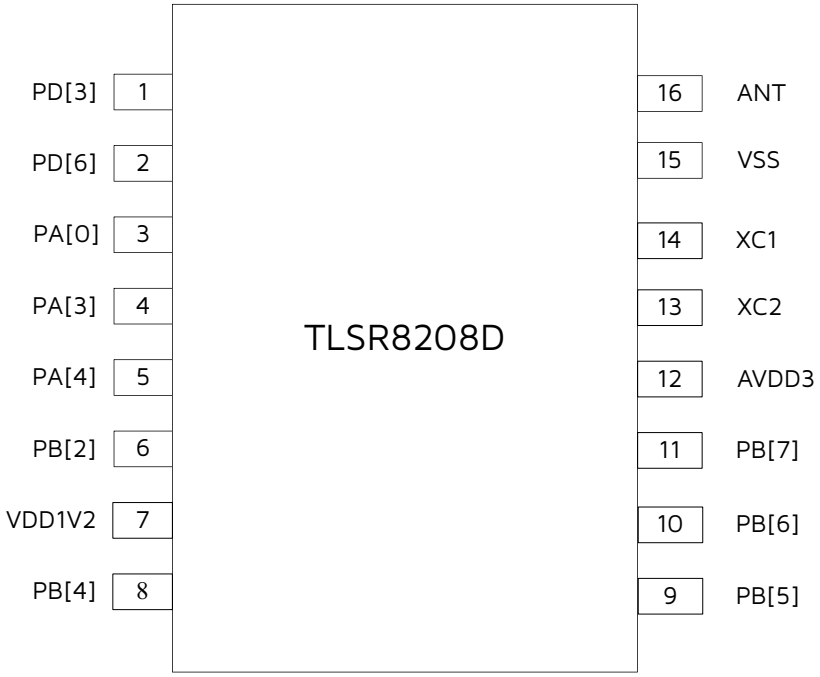
Pad	Default	Func1	Func2	Func3	Analog Func
PC[5]	GPIO	All functions	PC_KS5_IO	WIFI_DENY_I	-
PC[6]	GPIO	All functions	PC_KS6_IO	RX_CYC2LNA	-
PC[7]	GPIO	All functions	PC_KS7_IO	TX_CYC2PA	-
PD[0]	GPIO	All functions	PD_KS0_IO	I2C_SCL_IO	-
PD[1]	GPIO	All functions	PD_KS1_IO	I2C_SDA_IO	-
PD[2]	GPIO	All functions	PD_KS2_IO	PWM4	-
PD[3]	GPIO	All functions	PD_KS3_IO	PWM5	-
PD[4]	SPI_IO3	-	PD_KS4_IO	SPI_IO3_IO	-
PD[5]	GPIO	All functions	PD_KS5_IO	PWM1_N	-
PD[6]	GPIO	All functions	PD_KS6_IO	CLK_7816	-
PD[7]	GPIO	All functions	PD_KS7_IO	UART_RTX_IO	-
PF[1]	SPI_MISO	-	PA_KS2_IO	SPI_MISO_IO	-

a. "All functions" include 32 functions: WIFI_DENY_I, BLE_STATUS, BLE_ACTIVITY, SPI_CN_IO, SPI_CK_IO, SPI_MOSI_IO, SPI_MISO_IO, SWM_IO, TX_CYC2PA, RX_CYC2LNA, ANT_SEL2, ANT_SEL1, ANT_SEL0, UART_RTX_IO, CLK_7816, I2C_SDA_IO, I2C_SCL_IO, UART_RX_I, UART_TX, UART_RTS, UART_CTS_I, PWM5_N, PWM4_N, PWM3_N, PWM2_N, PWM1_N, PWM0_N, PWM5, PWM4, PWM3, PWM2, PWM1, PWM0

1.6.4 Pin Layout for TLSR8208D

Pin assignment of TLSR8208D is shown below.

Figure 1-9 Pin Assignment of TLSR8208D



Functions of the 16 pins of TLSR8208D are shown in the table below.

Table 1-12 Pin Function of TLSR8208D

No.	Pin Name	Type	Description
1	PD[3]	GPIO	GPIO PD[3], refer to Table 1-13 for pin mux function.
2	PD[6]	GPIO	GPIO PD[6], refer to Table 1-13 for pin mux function.
3	PA[0]	GPIO	GPIO PA[0], refer to Table 1-13 for pin mux function.
4	PA[3]	GPIO	GPIO PA[3], refer to Table 1-13 for pin mux function.
5	PA[4]	GPIO	GPIO PA[4], refer to Table 1-13 for pin mux function.
6	PB[2]	GPIO	GPIO PB[2], refer to Table 1-13 for pin mux function.
7	VDD1V2	PWR	Digital core supply.
8	PB[4]	GPIO	GPIO PB[4], refer to Table 1-13 for pin mux function.
9	PB[5]	GPIO	GPIO PB[5], refer to Table 1-13 for pin mux function.
10	PB[6]	GPIO	GPIO PB[6], refer to Table 1-13 for pin mux function.
11	PB[7]	PWR	GPIO PB[7], refer to Table 1-13 for pin mux function.
12	AVDD3	PWR	Power supply input as AA battery application, output as Li/USB application.
13	XC2	Analog	Crystal oscillator pin.
14	XC1	Analog	Crystal oscillator pin.

No.	Pin Name	Type	Description
15	VSS	GND	Ground for the RF front end.
16	ANT	Analog	Pin to connect to the Antenna through the matching network.

GPIO pin mux functions of TLSR8208D are shown in the table below.

Table 1-13 GPIO Pin Mux of TLSR8208D

Pad	Default	Func1	Func2	Func3	Analog Func
PA[0]	GPIO	All functions[a]	PA_KS0_IO	UART_CTS_I	-
PA[3]	SWS	-	-	SWS_IO	sar_aio<9>
PA[4]	GPIO	All functions	PA_KS4_IO	SWM_IO	-
PB[2]	GPIO	All functions	PB_KS2_IO	PWM0	sar_aio<2>
PB[4]	GPIO	All functions	PB_KS4_IO	PWM0_N	sar_aio<4>
PB[5]	GPIO	All functions	PB_KS5_IO	PWM1	sar_aio<5>
PB[6]	GPIO	All functions	PB_KS6_IO	PWM2	sar_aio<6>
PB[7]	GPIO	All functions	PB_KS7_IO	PWM3	sar_aio<7>
PD[3]	GPIO	All functions	PD_KS3_IO	PWM5	-
PD[6]	GPIO	All functions	PD_KS6_IO	CLK_7816	-

a. "All functions" include 32 functions: WIFI_DENY_I, BLE_STATUS, BLE_ACTIVITY, SPI_CN_IO, SPI_CK_IO, SPI_MOSI_IO, SPI_MISO_IO, SWM_IO, TX_CYC2PA, RX_CYC2LNA, ANT_SEL2, ANT_SEL1, ANT_SEL0, UART_RTX_IO, CLK_7816, I2C_SDA_IO, I2C_SCL_IO, UART_RX_I, UART_TX, UART_RTS, UART_CTS_I, PWM5_N, PWM4_N, PWM3_N, PWM2_N, PWM1_N, PWM0_N, PWM5, PWM4, PWM3, PWM2, PWM1, PWM0

NOTE:

- The initial status of PB[0], PB[1], PB[3], PD[4], PF[0] and PF[1] at power up or after sleep wake-up is SPI function, so these pins are not recommend to use as wakeup source.

2 Memory and MCU

2.1 Memory

The TLSR8208 embeds 16 KB SRAM with retention in deep sleep as data memory, 16 KB OTP, and 128/512 KB internal or external FLASH as program memory.

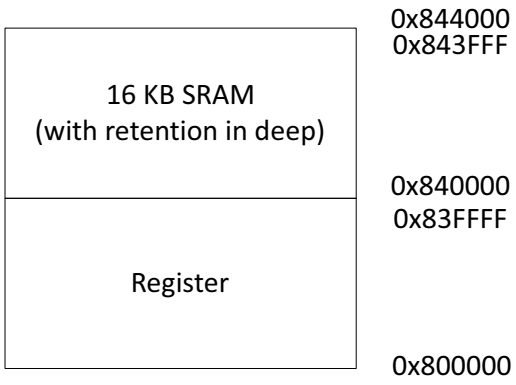
NOTE:

- TLSR8208B/D embeds 128 KB FLASH.
- TLSR8208C embeds 512 KB FLASH.

2.1.1 SRAM/Register

The SRAM/register memory map is shown as follows:

Figure 2-1 Phyical Memory Map

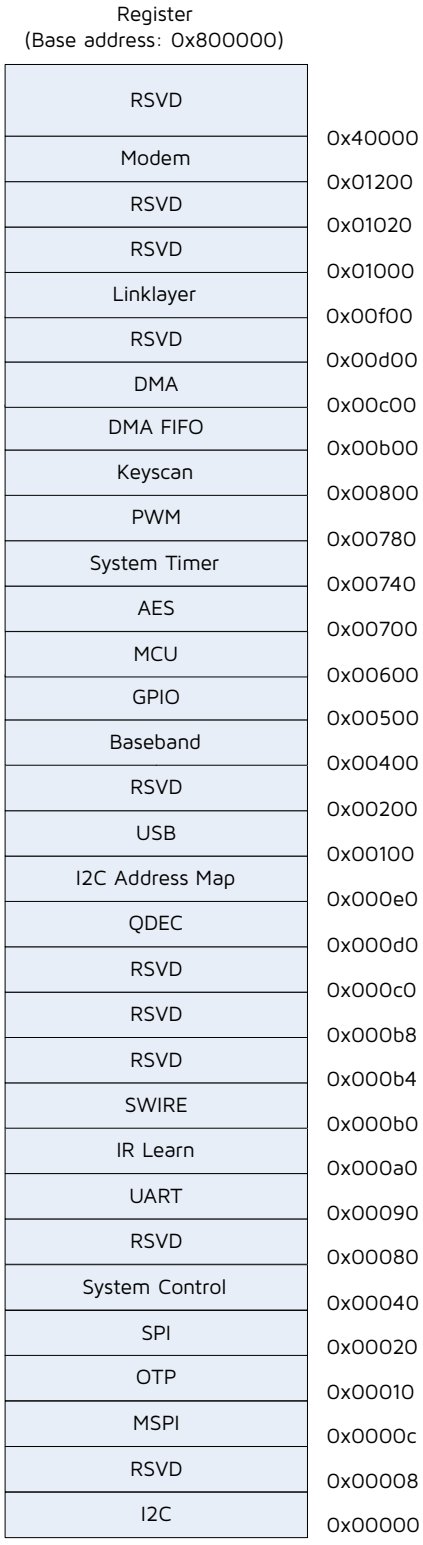


Register address: 0x800000 ~ 0x83FFFF.

Address for 16 KB SRAM with retention in deep sleep: 0x840000 ~ 0x843FFF.

Both register and SRAM address can be accessed (read or write) via debugging interface (SWS/SWM, SPI/I2C interface).

Figure 2-2 Register Space



2.1.2 Flash

For TLSR8208B/C/D, the internal Flash mainly supports page program, sector/block/chip erase operations, and deep power down operation. Please refer to the corresponding SDK for Flash memory operation details.

MCU uses the system frequency to load instructions, and adopts flash driver to access (read/write) Flash with the speed of half of the system clock.

TLSR8208A embeds a MSPI interface for external Flash.

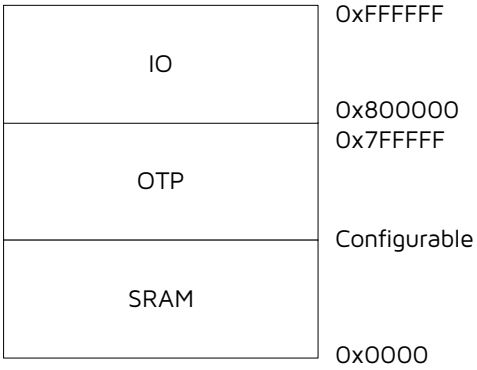
2.1.3 OTP

The TLSR8208 embeds a 4K x 32 bits OTP device with the following features:

- 1-bit program operation
- Build in 1 bit charge pump
- Build in ECC scheme
- Data retention: > 10 years
- Bit program time: 20 μs (min)

OTP address mapping is configurable. The MCU memory map is shown below.

Figure 2-3 MCU Memory Map



The OTP section is preloaded with OTP configuration shown as below.

Table 2-1 OTP Definition

OTP Information	MAC Address	Die Trace Info			Internal Information
		Internal Information	Wafer No.	Lot No.	
	Byte 7F48 ~ 7F4F	Byte 7F50 ~ 7F51	Byte 7F52 ~ 7F52	Byte 7F52 ~ 7F53	Byte 7F54 ~ 7FFC

2.1.4 Unique ID

For chip identification and traceability, the flash is preloaded with 128-bit Unique ID (UID). This UID can be read via the interface in SDK.

2.2 MCU

The TLSR8208 integrates a powerful 32-bit MCU developed by Telink. The digital core is based on 32-bit RISC, and the length of instructions is 16 bits; four hardware breakpoints are supported.

2.3 Working Modes

The TLSR8208 supports six working modes, including Active, Idle, Suspend, Deep Sleep with SRAM retention, Deep Sleep without SRAM retention, and Shutdown.

- The Power Management (PM) module is always active in all working modes.
- For modules such as MCU, RF transceiver (Radio), and SRAM, the state depends on working mode, as shown below.

Table 2-2 Working Modes

Mode	Active	Idle	Suspend	Deep Sleep with SRAM Retention	Deep Sleep Without SRAM Retention	Shutdown
MCU	active	stall	stall	off	off	off
Radio	available	available	off	off	off	off
USB	available	available	off	off	off	off
Wakeup time to Active mode	-	0 μs	100 μs	Shorter than Deep Sleep without retention, almost same as Suspend	1 ms	10 ms
retention SRAMs (with retention in deep sleep)	full	full	full	full	off	off
Wakeup on RTC (32K Timer wakeup)	-	-	available	available	available	off
Wakeup on pin (IO wakeup)	-	-	available	available	available	off
Wakeup on interrupt	-	available	-	-	-	-
Wakeup on reset pin (RESETB)	-	available	available	available	available	on
Current	Please refer to Section 13.3.					

NOTE:

- "active": MCU is at working state.
- "stall": In Idle and Suspend mode, MCU does not work, while its clock is still running.
- "available" for Modules: It's selectable to be at working state, or stall/be powered down if it does not need to work.
- "available"/"on" for wakeup: Corresponding wakeup method is supported.
- "off" for wakeup: Corresponding wakeup method is not supported.
- "on"/"off"/"full" for SRAMs:
  - "on": The 16 KB SRAM is powered on and works normally (can be accessed) in Active, Idle and Suspend mode.
  - "full": Full speed. In Active, Idle and Suspend mode, the 16 KB retention SRAM is powered on and work normally (can be accessed); in Deep Sleep with SRAM retention, the retention SRAM is powered on, however, the contents of the retention SRAM can be retained and cannot be accessed.
  - "off": The 16 KB SRAM is powered down in two Deep Sleep modes and Shutdown mode. The retention SRAMs are powered down in Deep Sleep without SRAM retention and Shutdown mode.
- Current:
  - In Deep Sleep without SRAM retention, only the PM module is active, all digital and analog modules are powered down, thus the power consumption is largely decreased.
  - In Deep Sleep with SRAM retention, the PM module is active, all analog and digital modules except for the retention SRAM is powered down, thus the power consumption is a little higher than in Deep Sleep without SRAM retention, but much lower than in Suspend.

Table 2-3 Retention Analog Registers in Deep Sleep

Address	R/W	Description	Default Value
afe_0x35	RW	buffer, clean at watch dog reset	0x20
afe_0x36	RW	buffer, clean at watch dog reset	0x00
afe_0x37	RW	buffer, clean at watch dog reset	0x00
afe_0x38	RW	buffer, clean at watch dog reset	0x00
afe_0x39	RW	buffer, clean at watch dog reset	0xff
afe_0x3a	RW	buffer, clean at power on reset	0x00
afe_0x3b	RW	buffer, clean at power on reset	0x00
afe_0x3c	RW	buffer, clean at power on reset	0x0f

Analog registers (0x35 ~ 0x3c) as shown in the table above are retained in deep sleep mode and can be used to store program state information across deep sleep cycles.

- Analog registers 0x3a ~ 0x3c are non-volatile even when chip enters deep sleep or chip is reset by watchdog or software, i.e. the contents of these registers won't be changed by deep sleep or watchdog reset or chip software reset.

- Analog registers 0x35 ~ 0x39 are non-volatile in deep sleep, but will be cleared by watchdog reset or chip software reset.
- After POR (Power-On-Reset), all registers will be cleared to their default values, including these analog registers.

User can set flag in these analog registers correspondingly, so as to check the booting source by reading the flag.

For chip software reset, please refer to Section 2.4.

2.4 Reset

The chip supports three types of reset methods, including POR (Power-On-Reset), watchdog reset and software reset.

1. POR: After power on, the whole chip will be reset, and all registers will be cleared to their default values.
2. Watchdog reset: A programmable watchdog is supported to monitor the system. If watchdog reset is triggered, registers except for the retention analog registers 0x3a ~ 0x3c will be cleared.
3. Software reset: It is also feasible to carry out software reset for the whole chip or some modules.
   - Setting address 0x6f[5] as 1'b1 is to reset the whole chip. Similar to watchdog reset, the retention analog registers 0x3a ~ 0x3c are non-volatile, while other registers including 0x35 ~ 0x39 will be cleared by chip software reset.
   - Addresses 0x60 ~ 0x62 serve to reset individual modules: if some bit is set to logic "1", the corresponding module is reset.

Table 2-4 Register Configuration for Software Reset

Address	Name	R/W	Description	Default Value
0x60	RST0	RW	Reset control, 1 for reset, 0 for clear [0] SPI [1] I2C [2] RS232, i.e. UART [3] USB [4] PWM0 [5] QDEC [6] IR_LEARN [7] SWIRE	0x7c

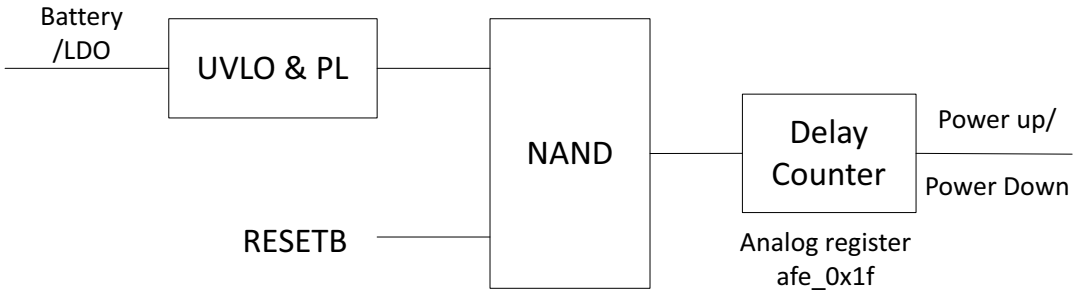
Address	Name	R/W	Description	Default Value
0x61	RST1	RW	[0] ZB, i.e. Baseband [1] System Timer [2] DMA [3] ALGM [4] AES [5] ADC [6] ALG [7] RSVD (Public Key Engine)	0xff
0x62	RST2	RW	[0] AIF [1] RSVD (AUDIO) [2] DFIFO [3] RSVD (TRNG) [4] RISC [5] MCIC [6] RISC1 (R) [7] EOTP	0x4f
0x6f	PWDNEN	RW	[0] suspend enable (RW) [4] clear ramcrc enable (W1C) [5] reset all (act as watchdog reset) (W) [6] RSVD (mcu low power mode) (W) [7] stall mcu trig. If bit[0] set 1, then system will go to suspend. Or only stall mcu (W)	0x00

2.5 Power Management

The multiple-stage Power Management (PM) module is flexible to control power state of the whole chip or individual functional blocks such as MCU, RF Transceiver, and peripherals.

2.5.1 Power-On-Reset (POR) and Brown-Out Detect

Figure 2-4 Control Logic for Power Up/Down



The whole chip power up and down is controlled by the UVLO (Ultra-low Voltage Lockout) & PL (Power Logic) module and the external RESETB pin via the logic shown in the above diagram. UVLO takes the external power supply as input and releases the lock only when the power supply voltage is higher than a preset threshold. The RESETB pin has an internal pull-up resistor; an external Cap can be connected on the RESETB pin to control the POR delay.

After both UVLO and RESETB release, there is a further configurable delay before the system reset signal ("Sysrst") is released. The delay is adjusted by analog register afe_0x1f. Since the content of afe_0x1f is reset to default only after power cycle, watchdog reset, or software reset, the delay change using afe_0x1f is only applicable when the chip has not gone through these reset conditions. For example, after deep sleep wakeup, the setting in afe_0x1f will take effect.

Table 2-5 Analog Register to Control Delay Counters

Address	Name	R/W	Description	Default Value
afe_0x1f	r_dly	RW	Wait for Boost LDO ready (based on 16 kHz count decrement counter)	0x80

Figure 2-5 Initial Power-Up Sequence

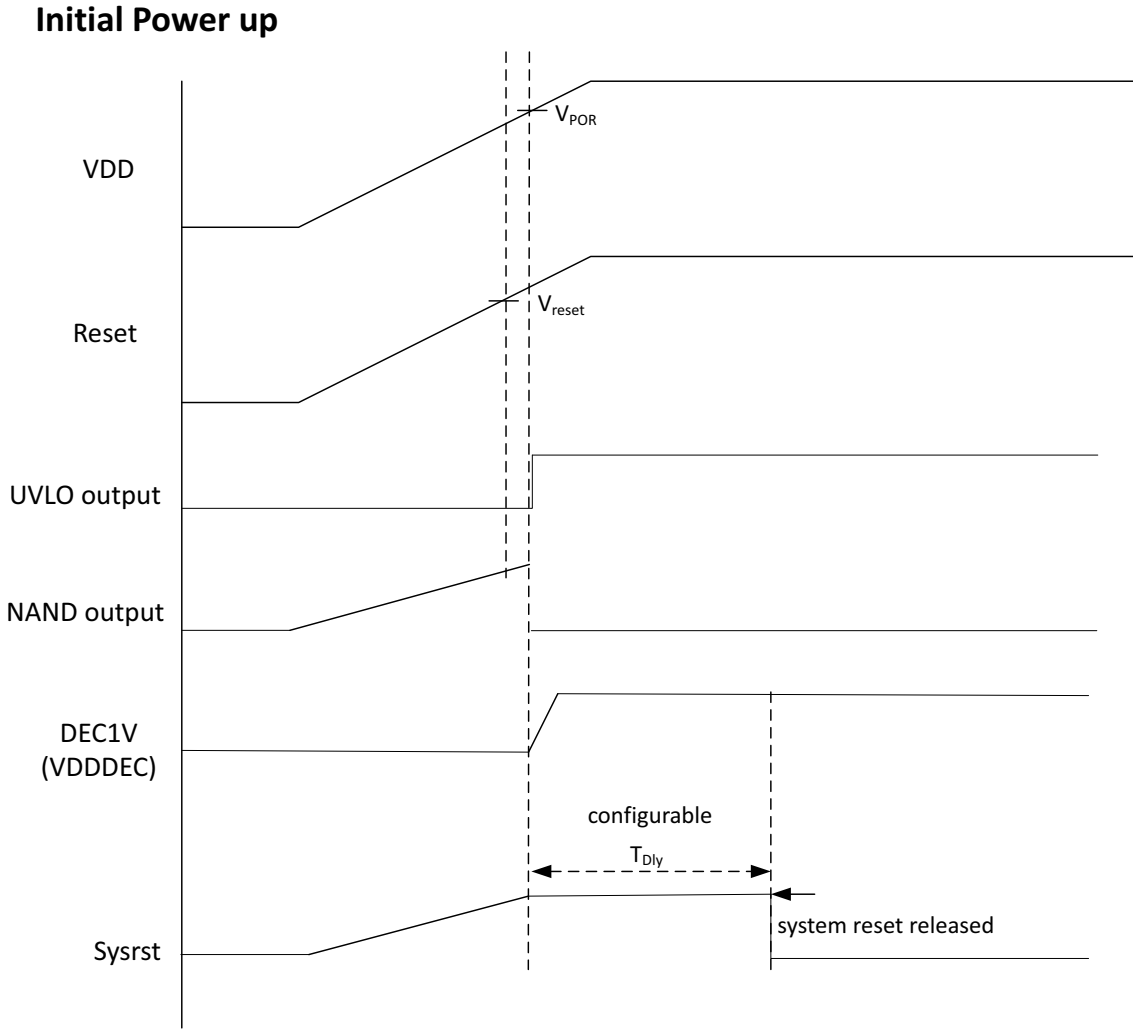


Figure 2-6 Power-Down Sequence

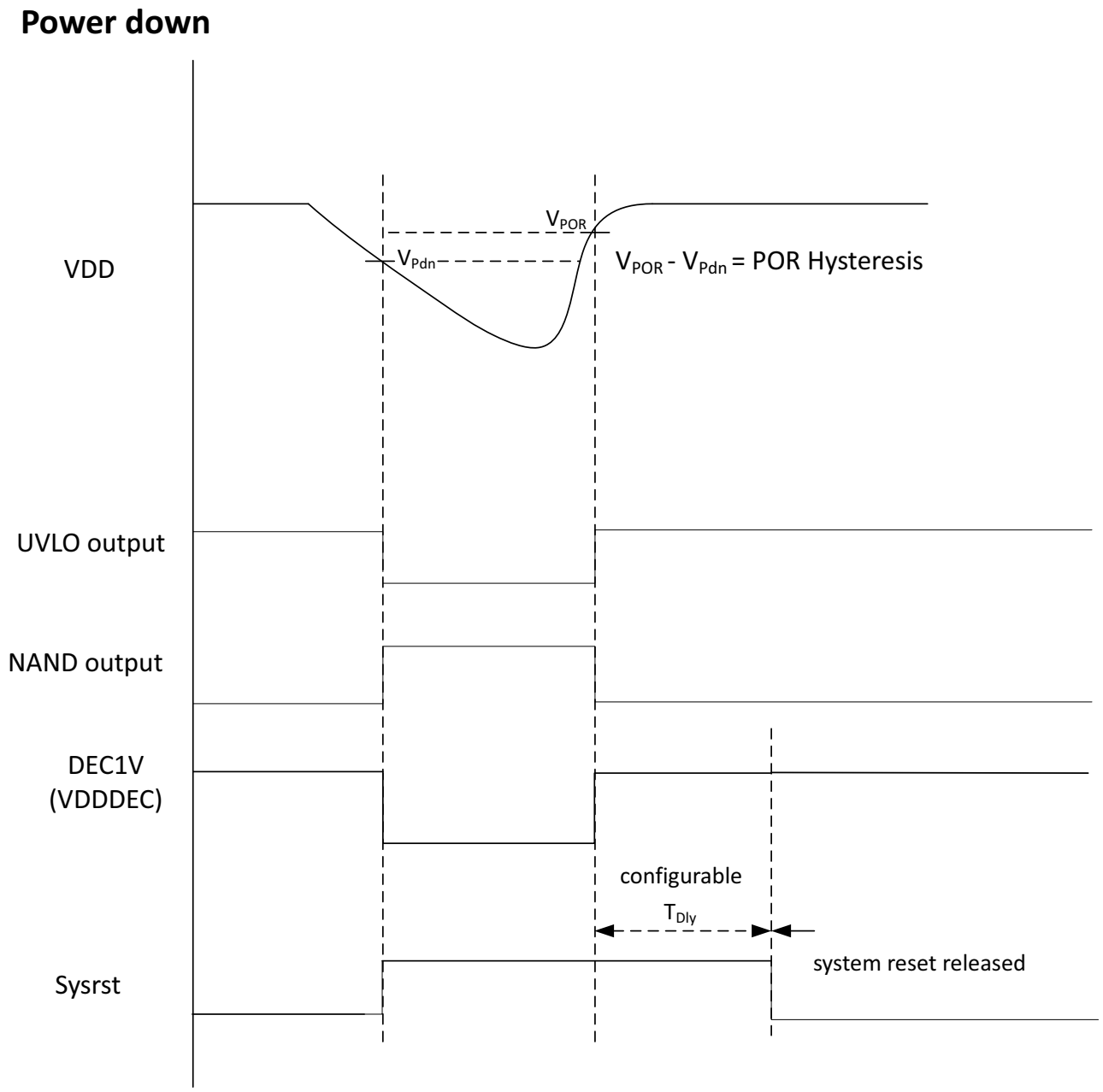


Table 2-6 Characteristics of Initial Power-Up/Power-Down Sequence

Symbol	Parameter	Min	Typ	Max	Unit
V_{POR}	VDD voltage when V_{UVLO} turns to high level	-	1.62	-	V
V_{Pdn}	VDD voltage when V_{UVLO} turns to low level	-	1.55	-	V
T_{Dly}	Delay counter value	Configurable via analog register afe_0x1f			

2.5.2 Working Mode Switch

In Active mode, MCU is active, all SRAMs are accessible, and other modules are selectable whether to be at working state.

The chip can switch to Idle mode to stall the MCU. In this mode, all SRAMs are still accessible, modules such as RF transceiver, are still selectable whether to be at working state. The chip can be triggered to Active mode by interrupt or RESETB pin, and the time to switch to Active mode is negligible.

To decrease power consumption to different levels, the chip can switch to power saving mode (Suspend, Deep Sleep with SRAM retention, Deep Sleep without SRAM retention, Shutdown) correspondingly. (Please refer to Table 2-2.)

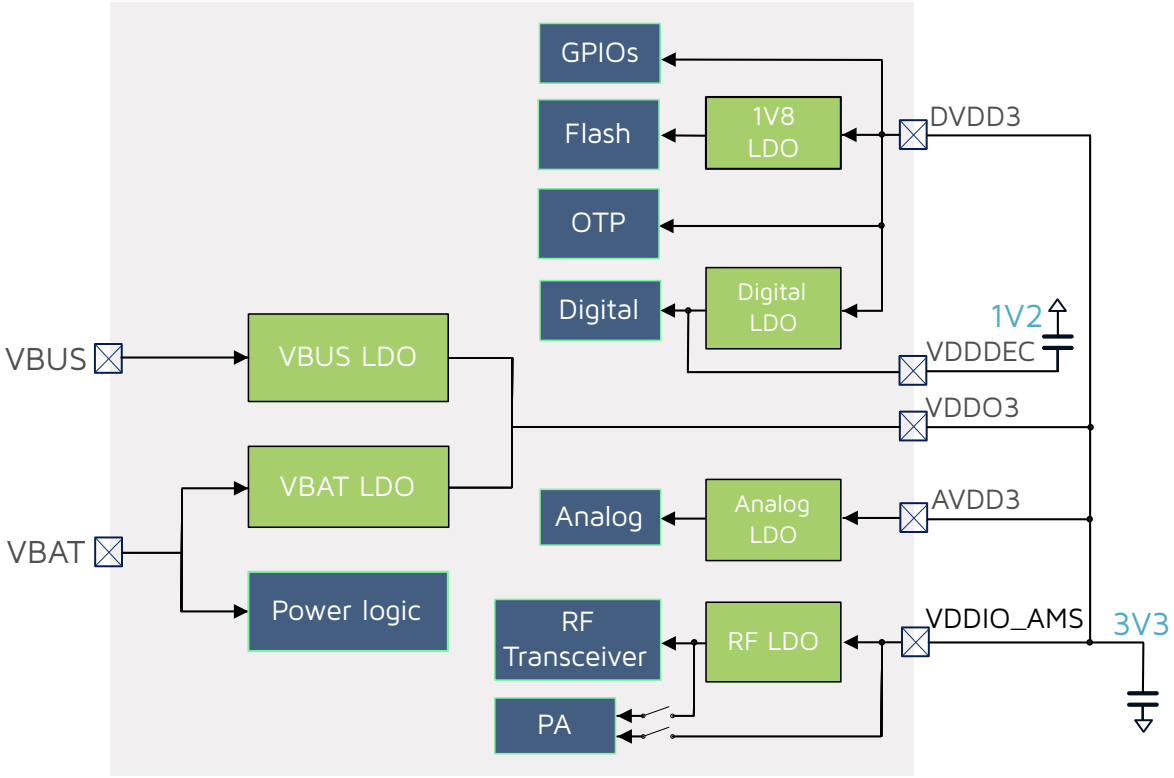
- In Suspend mode, MCU stalls, all SRAMs are still accessible, the PM module is active, modules such as RF transceiver and USB are powered down. The chip can be triggered to Active mode by 32K Timer, IO pin or RESETB pin. It takes 100 µs or so to switch from Suspend mode to Active mode.
- In Deep Sleep with SRAM retention, the PM module is active, analog and digital modules except for the 16 KB retention SRAM is powered down, while the retention SRAM can be retained and not accessible. The chip can be triggered to Active mode by 32K Timer, IO pin or RESETB pin. The time to switch to Active mode is shorter than Deep Sleep without SRAM retention and close to Suspend.
- In Deep Sleep without SRAM retention, only the PM module is active, while analog and digital modules including the retention SRAM is powered down. The chip can be triggered to Active mode by 32K Timer, IO pin or RESETB pin. The time to switch to Active mode is 1 ms or so.
- In Shutdown mode, all digital and analog modules are powered down, and only the PM module is active. The chip can be triggered to Active mode by RESETB pin only. The time to switch to Active mode is 10 ms or so.

User can directly invoke corresponding library function to switch working mode of the chip. If certain module doesn't need to work, user can power down this module in order to save power.

2.5.3 LDO

The diagram of LDO module is shown as following.

Figure 2-7 LDO



The chip embedded LDO, can generate 1.8 V output voltage for internal flash; this LDO block also generates 1.4 V output voltage.

Another embedded LDO regulator takes the 1.4 V voltage output from the LDO, and generates 1.2 V regulated voltage to supply power for 1.2 V digital core and analog modules in Active/Idle mode. The RF block is supplied by the 1.4 V output from the LDO, the power amplifier (PA) of RF can be either powered by 1.4 V or directly from battery depending on VANT or VBAT mode, respectively.

2.5.4 VBAT and VANT Power-Supply Mode

The RF PA module has two power-supply modes including VBAT mode and VANT mode.

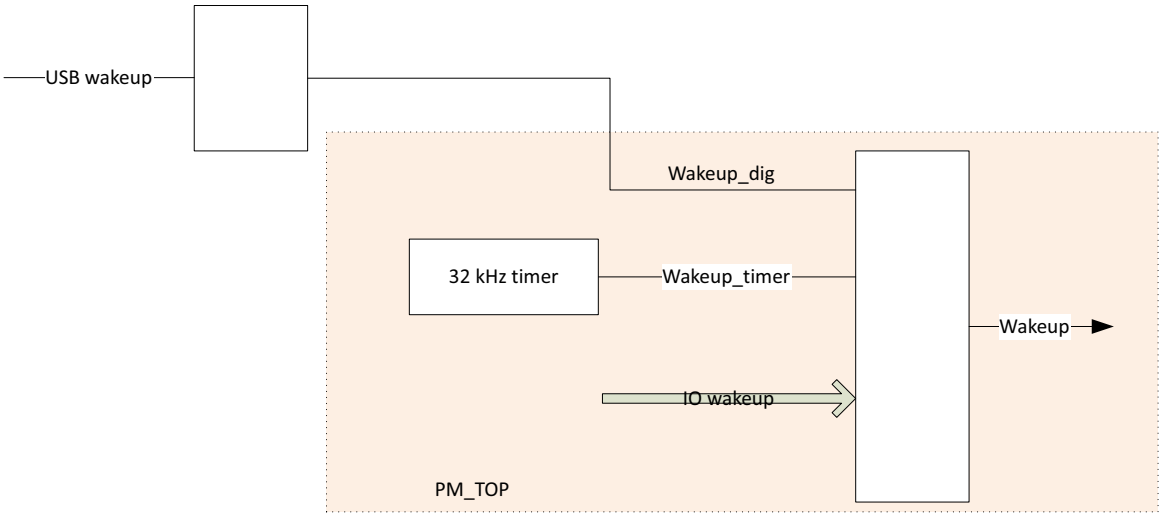
- In VBAT mode, the RF PA module is supplied by 3.3 V voltage regulated from 4.2V lithium battery or directly from two AA/AAA batteries in series. The maximum output power is related to power supply voltage of RF PA, for example, the maximum output power is 10 dBm at 3.3 V power supply, and 6 dBm at 1.8 V.
- In VANT mode, the RF PA module is supplied with 1.4 V voltage by the embedded LDO. In this mode, the output power won't change with AVDD which is converted from VBAT voltage, and the power stays constantly around 4 dBm.

Comparing to the VBAT mode, the VANT mode is more power-saving at the same TX power.

When the chip works in VBAT mode, it can be configured to the maximum output power. However, as the VBAT/VDD supply decreases below 3.0 V, the maximum transmit power of TX is then slightly attenuated. The detailed RF transmit power level refers to the code comments in the corresponding driver SDK, in which the RF transmit power level under VBAT mode is the result tested in 3.3 V VBAT voltage.

2.6 Wakeup Sources

Figure 2-8 Wakeup Sources



2.6.1 Wakeup Source - USB

This wakeup source can only wake up the system from suspend mode.

First, set the digital register 0x6e bit[2] as 1'b1.

To activate this mode, analog register afe_0x2a[1] should also be set as 1'b1.

Once USB host sends out resuming signal, the system will be woke up.

2.6.2 Wakeup Source - 32 kHz Timer

This wakeup source is able to wake up the system from suspend mode or two deep sleep modes.

To enable the wakeup source from 32 kHz timer, analog register afe_0x2a[2] should be set as 1'b1.

2.6.3 Wakeup Source - IO

This wakeup source is able to wake up the system from suspend mode or two deep sleep modes. And IO wakeup supports high level or low level wakeup which is configurable via polarity control registers.

Analog register afe_0x2a[0] should be set as 1'b1 to enable IO wakeup source.

Enabling control analog registers: PA[7:0] enabling control register is afe_0x25[7:0], PB[7:0] enabling control register is afe_0x26[7:0], PC[7:0] enabling control register is afe_0x27[7:0], PD[7:0] enabling control register is afe_0x28[7:0], and PF[1:0] enabling control register is afe_0x29[1:0]. Total wakeup pins can be up to 34.

Polarity control registers: PA[7:0] polarity control register is afe_0x20[7:0], PB[7:0] polarity control register is afe_0x21[7:0], PC[7:0] polarity control register is afe_0x22[7:0], PD[7:0] polarity control register is afe_0x23[7:0], and PF[1:0] polarity control register is afe_0x24[1:0].

The corresponding driver is available so that user can directly invoke it to use IO wakeup source.

Analog register 0x44[2:0] indicates the wakeup source which triggers system wakeup. After wakeup, the corresponding wakeup status will be set as 1'b1 automatically, and it's needed to write 1 to manually clean the status.

2.6.4 Register Table

Table 2-7 Analog Registers for Wakeup

Address	R/W	Description	Default Value
afe_0x20	RW	Polarity control registers for IO wakeup 0: high level wakeup, 1: low level wakeup	0x00
afe_0x21	RW		0x00
afe_0x22	RW		0x00
afe_0x23	RW		0x00
afe_0x24	RW		0x00
afe_0x25	RW	Enabling control registers for IO wakeup	0x00
afe_0x26	RW		0x00
afe_0x27	RW		0x00
afe_0x28	RW		0x00
afe_0x29	RW		0x00
afe_0x2a	RW	[0] IO (pad) wakeup enable [1] Digital core wakeup enable [2] 32 kHz timer wakeup enable [3] RSVD [4] RSVD [5] RSVD [6] RSVD [7] shutdown wakeup enable	0x00
afe_0x44	R	Write 1 to clean the status: [0] IO (pad) wakeup status [1] Digital core wakeup status [2] 32 kHz timer wakeup status [3] RSVD [4] RSVD [5] RSVD [6] RSVD [7] RSVD	-

Table 2-8 Digital Register for Wakeup

Address	R/W	Description	Default Value
0x6e	RW	Wakeup enable [0] enable wakeup from I2C host [1] enable wakeup from SPI host [2] enable wakeup from USB [3] enable wakeup from GPIO [4] enable wakeup from I2C synchronous interface System resume control [5] enable GPIO remote wakeup [6] If set to 1, system will issue USB resume signal on USB bus [7] sleep wakeup reset system enable	0x1f

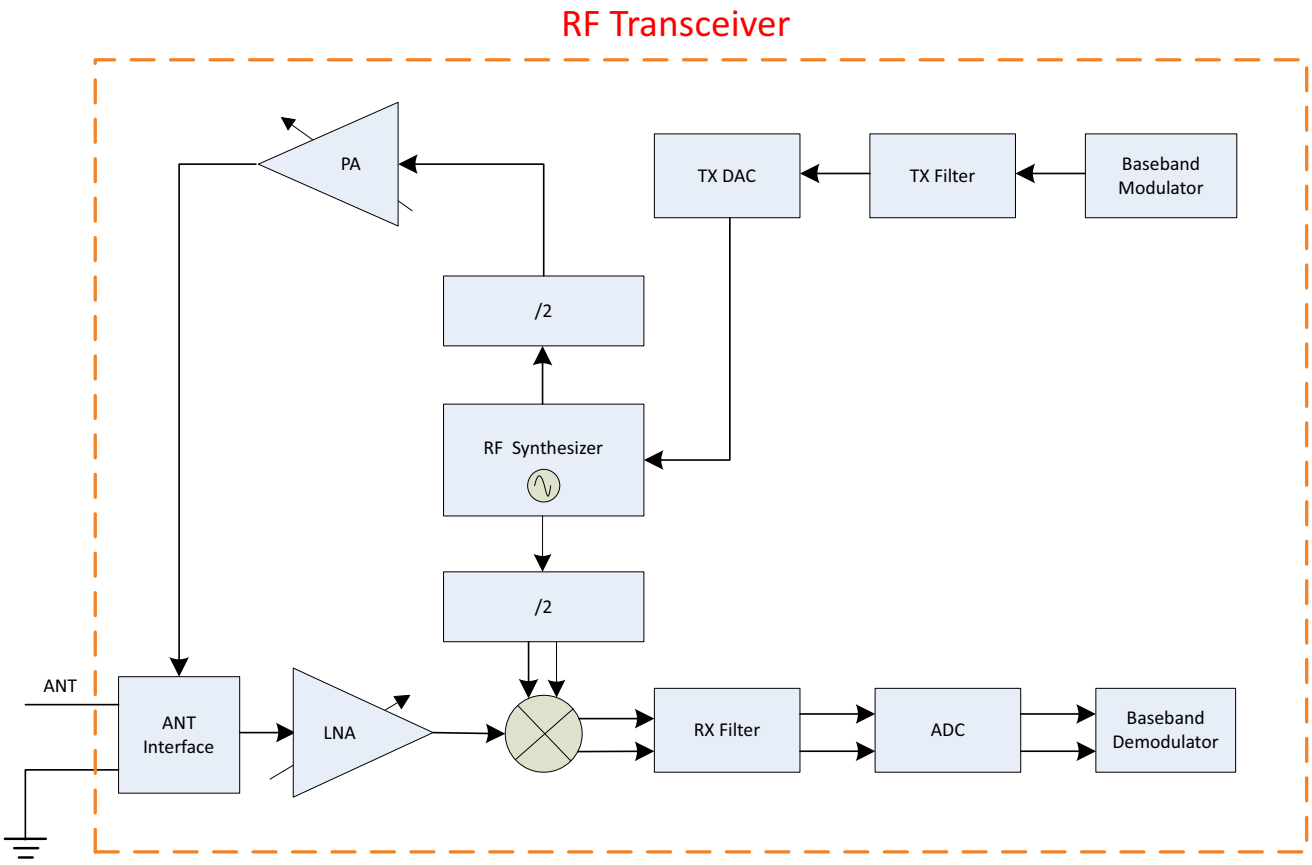
3 BLE/2.4 GHz RF Transceiver

3.1 Block Diagram

The TLSR8208 integrates an advanced BLE/2.4 GHz RF transceiver. The RF transceiver works in the worldwide 2.4 GHz ISM (Industrial Scientific Medical) band.

The transceiver consists of a fully integrated RF synthesizer, a power amplifier (PA), a low noise amplifier (LNA), a TX filter, a RX filter, a TX DAC, an ADC, a modulator and a demodulator. The transceiver can be configured to work in standard-compliant 1 Mbps BLE mode, 2 Mbps enhancement BLE mode and proprietary 1 Mbps, 2 Mbps, 250 kbps and 500 kbps mode.

Figure 3-1 Block Diagram of RF Transceiver



The internal PA can deliver a maximum 5 dBm output power, avoiding the need for an external RF PA.

3.2 Air Interface Data Rate and RF Channel Frequency

Air interface data rate, the modulated signaling rate for RF transceiver when transmitting and receiving data, is configurable via related register setting: 250 kbps, 500 kbps, 1 Mbps, 2 Mbps.

For the TLSR8208, RF transceiver can operate with frequency ranging from 2.400 GHz to 2.4835 GHz. The RF channel frequency setting determines the center of the channel.

3.3 Baseband

The baseband is disabled by default. The corresponding API is available for user to power on/down the baseband and enable/disable clock, so that the baseband can be turned on/off flexibly.

The baseband contains dedicated hardware logic to perform fast AGC control, access code correlation, CRC checking, data whitening, encryption/decryption and frequency hopping logic.

The baseband supports all features required by Bluetooth specification.

3.3.1 Packet Format

Packet format in standard 1 Mbps BLE mode is shown in Table 3-1.

Table 3-1 Packet Format in Standard 1 Mbps BLE Mode[a]

LSB MSB

Preamble (1 octet)	Access Address (4 octets)	PDU (2 ~ 257 octets)	CRC (3 octets)

a. Packet length 80 bit ~ 2120 bit (80 ~ 2120 μs @ 1 Mbps).

Packet format in standard 2 Mbps BLE mode is shown in Table 3-2.

Table 3-2 Packet Format in Standard 2 Mbps BLE Mode

LSB MSB

Preamble (2 octets)	Access Address (4 octets)	PDU (2 ~ 257 octets)	CRC (3 octets)

Packet format in 2.4 GHz proprietary mode is shown in Table 3-3.

Table 3-3 Packet Format in Proprietary Mode

LSB MSB

Preamble (8 bits)	Address code (configurable 3 ~ 5 bytes)	Packet Controller + Payload (1 ~ 63 bytes)	CRC (1 ~ 2 bytes)

3.3.2 RSSI and Frequency Offset

The TLSR8208 provides accurate RSSI (Receiver Signal Strength Indicator) and frequency offset indication.

- RSSI can be read from the 1 byte at the tail of each received data packet.
- If no data packet is received (e.g. to perform channel energy measurement when no desired signal is present), real-time RSSI can also be read from specific registers which will be updated automatically.
- RSSI monitoring resolution can reach +/-1 dB.
- Frequency offset can be read from the 2 bytes at the tail of the data packet. Valid bits of actual frequency offset may be less than 16 bits, and different valid bits correspond to different tolerance range.

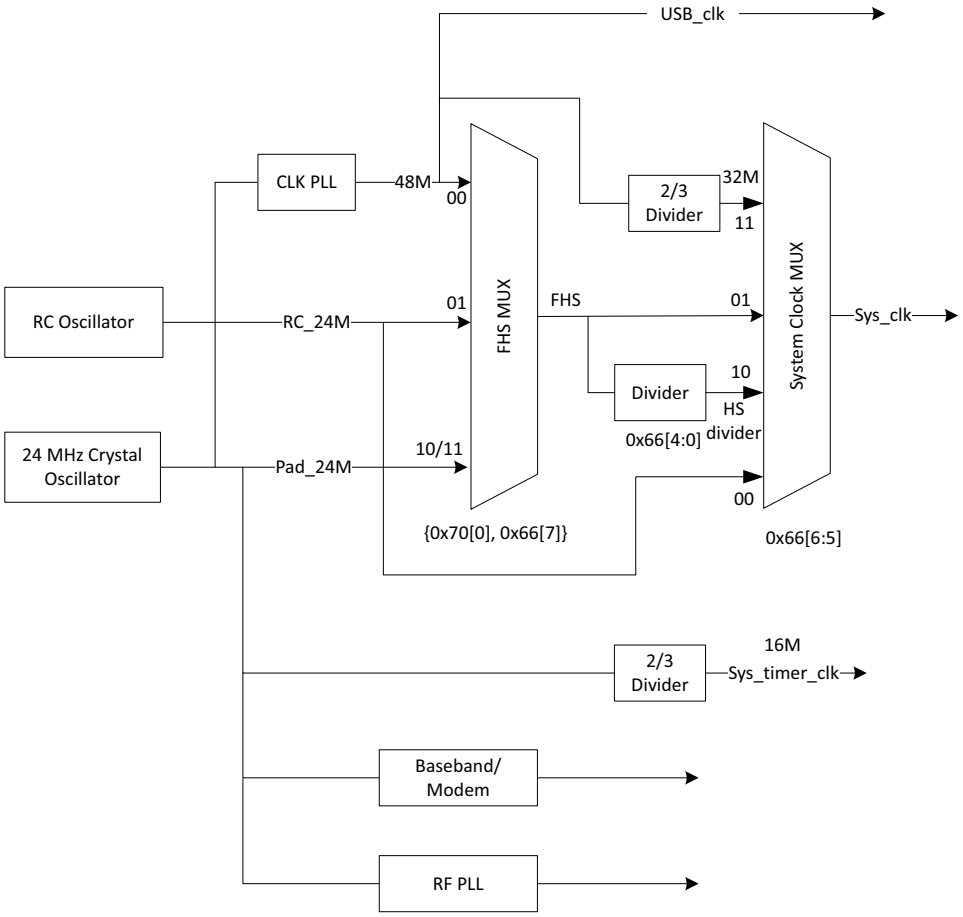
Telink supplies corresponding drivers for user to read RSSI and frequency offset as needed.

4 Clock

4.1 Clock Sources

The TLSR8208 embeds a 24 MHz RC oscillator which can be used as clock source for system. External 24 MHz crystal is available via pin XC1 and XC2, which can provide a Pad_24MHz clock source for system and System Timer, and generate a 48M clock via a frequency doubler to provide clock source for USB. The block diagram of the TLSR8208 clock is shown below.

Figure 4-1 Block Diagram of Clock



4.2 System Clock

There are four selectable clock sources for MCU system clock: RC_24M derived from 24 MHz RC oscillator, High speed clock "FHS", HS divider clock (derived from "FHS" via a frequency divider), 32 MHz clock derived from 48 MHz clock via a 2/3 frequency divider (The 48M clock is derived from 24M crystal oscillator via a frequency doubler).

The high speed clock (FHS) is selectable via address {0x70[0], 0x66[7]} from the following sources: 48 MHz clock (derived from 24M crystal oscillator via a frequency doubler), RC_24M (derived from 24 MHz RC oscillator), and Pad_24M (derived from 24M crystal oscillator).

The digital register CLKSEL (address 0x66) serves to set system clock: System clock source is selectable via bit[6:5].

If address 0x66[6:5] is set to 2'b10 to select the HS divider clock, system clock frequency is adjustable via address 0x66[4:0]. The formula is shown as below:

$$F_{System\ clock} = F_{FHS} / (system\ clock\ divider\ value\ in\ address\ 0x66[4:0])$$

NOTE: *Address 0x66[4:0] should not be set as 0 or 1.*

4.3 Module Clock

Registers CLKEN0 ~ CLKEN2 (address 0x63 ~ 0x65) are used to enable or disable clock for various modules. By disabling the clocks of unused modules, current consumption could be reduced.

4.3.1 System Timer Clock

System Timer clock is derived from 24M crystal oscillator via a 2/3 frequency divider. The clock frequency is fixed as 16 MHz.

4.3.2 USB Clock

USB clock is derived from 48M clock. The 48M clock is derived from 24M crystal oscillator via a frequency doubler.

4.4 Register Table

Table 4-1 Clock Register Table

Address	R/W	Description	Default Value
0x63	RW	Clock enable control: 1 - enable; 0 - disable [0] SPI [1] I2C [2] UART (RS232) [3] USB [4] PWM [5] QDEC [6] IR_LEARN [7] Swire	0x83

Address	R/W	Description	Default Value
0x64	RW	[0] ZB [1] System Timer [2] DMA [3] ALGM [4] AES [5] KS_32K [6] KS [7] RSVD (Public Key Engine)	0x00
0x65	RW	[0] DBGEN [1] RSVD (AUDIO) [2] DFIFO [3] RSVD (TRNG) [4] MC [5] MCIC [6] MC1 [7] EOTP	0xb0
0x66	RW	System clock select [4:0] system clock divider (must exceed 1). If 0x66[6:5] is set as 2'b10, $F_{Sysclk} = F_{FHS} / (CLKSEL[4:0])$. [6:5] select system clock source 2'b00: RC_24M from RC oscillator 2'b01: FHS 2'b10: HS divider (see 0x66[4:0]) 2'b11: 16M clock (24M * 2/3 divider) [7] FHS select	0x06
0x70	RW	[0] Clock select 1: 32k, 0: 7816 clk	0x00
0x73	RW	[0] 32k clock select 0: select RC_32k from RC oscillator 1: select Pad_32k from 32k crystal oscillator[3] stimer clock select	0x04

5 Timers

5.1 Timer0 ~ Timer2

The TLSR8208 supports three timers: Timer0 ~ Timer2. The three timers all support four modes: Mode 0 (System Clock Mode), Mode 1 (GPIO Trigger Mode), Mode 2 (GPIO Pulse Width Mode) and Mode 3 (Tick Mode), which are selectable via the register TMR_CTRL0 (address 0x620) ~ TMR_CTRL1 (address 0x621).

Timer2 can also be configured as "watchdog" to monitor firmware running.

5.1.1 Register Table

Table 5-1 Register Configuration for Timer0 ~ Timer2

Address	R/W	Description	Default Value
0x72	W1C	[0] watch dog status: verify whether it is power reset (1'b0) or watch dog reset (1'b1), write 1 to clear.	0x00
0x620	RW	[0] Timer0 enable [2:1] Timer0 mode 0: using sclk, 1: using gpio, 2: count width of gpi, 3: tick [3] Timer1 enable [5:4] Timer1 mode [6] Timer2 enable [7] Bit of timer2 mode	0x00
0x621	RW	[0] Bit of timer2 mode [7:1] Low bits of watch dog capture	0x00
0x622	RW	[6:0] High bits of watch dog capture. It is compared with [31:18] of timer2 ticker [7] watch dog capture	0x00
0x623	W1C	[0] timer0 status, write 1 to clear [1] timer1 status, write 1 to clear [2] timer2 status, write 1 to clear [3] watch dog status, write 1 to clear (If Watchdog is enabled, need to clear it periodically to avoid triggering watchdog reset)	0x00
0x624	RW	Byte 0 of timer0 capture	0x00
0x625	RW	Byte 1 of timer0 capture	0x00

Address	R/W	Description	Default Value
0x626	RW	Byte 2 of timer0 capture	0x00
0x627	RW	Byte 3 of timer0 capture	0x00
0x628	RW	Byte 0 of timer1 capture	0x00
0x629	RW	Byte 1 of timer1 capture	0x00
0x62a	RW	Byte 2 of timer1 capture	0x00
0x62b	RW	Byte 3 of timer1 capture	0x00
0x62c	RW	Byte 0 of timer2 capture	0x00
0x62d	RW	Byte 1 of timer2 capture	0x00
0x62e	RW	Byte 2 of timer2 capture	0x00
0x62f	RW	Byte 3 of timer2 capture	0x00
0x630	RW	Byte 0 of timer0 ticker	0x00
0x631	RW	Byte 1 of timer0 ticker	0x00
0x632	RW	Byte 2 of timer0 ticker	0x00
0x633	RW	Byte 3 of timer0 ticker	0x00
0x634	RW	Byte 0 of timer1 ticker	0x00
0x635	RW	Byte 1 of timer1 ticker	0x00
0x636	RW	Byte 2 of timer1 ticker	0x00
0x637	RW	Byte 3 of timer1 ticker	0x00
0x638	RW	Byte 0 of timer2 ticker	0x00
0x639	RW	Byte 1 of timer2 ticker	0x00
0x63a	RW	Byte 2 of timer2 ticker	0x00
0x63b	RW	Byte 3 of timer2 ticker	0x00

5.1.2 Mode 0 (System Clock Mode)

In Mode 0, system clock is employed as clock source.

After Timer is enabled, Timer Tick (i.e. counting value) is increased by 1 on each positive edge of system clock from preset initial Tick value. Generally the initial Tick value is set to 0.

Once current Timer Tick value matches the preset Timer Capture (i.e. timing value), an interrupt is generated, Timer stops counting and Timer status is updated.

Steps of setting Timer0 for Mode 0 is taken as an example.

Step 1 Set initial Tick value of Timer0

Set Initial value of Tick via registers TMR_TICK0_0 ~ TMR_TICK0_3 (address 0x630 ~ 0x633). Address 0x630 is lowest byte and 0x633 is highest byte. It's recommended to clear initial Timer Tick value to 0.

Step 2 Set Capture value of Timer0

Set registers TMR_CAPT0_0 ~ TMR_CAPT0_3 (address 0x624 ~ 0x627). Address 0x624 is lowest byte and 0x627 is highest byte.

Step 3 Set Timer0 to Mode 0 and enable Timer0

Set register TMR_CTRL0 (address 0x620) [2:1] to 2'b00 to select Mode 0; Meanwhile set address 0x620[0] to 1'b1 to enable Timer0. Timer0 starts counting upward, and Tick value is increased by 1 on each positive edge of system clock until it reaches Timer0 Capture value.

5.1.3 Mode 1 (GPIO Trigger Mode)

In Mode 1, GPIO is employed as clock source. The "m0"/"m1"/"m2" register specifies the GPIO which generates counting signal for Timer0/Timer1/Timer2.

After Timer is enabled, Timer Tick (i.e. counting value) is increased by 1 on each positive/negative (configurable) edge of GPIO from preset initial Tick value. Generally the initial Tick value is set to 0. The "Polarity" register specifies the GPIO edge when Timer Tick counting increases.

NOTE: Refer to Section 7.1.3 *for corresponding "**m0**", "**m1**", "**m2**" and "**Polarity**" register address.*

Once current Timer Tick value matches the preset Timer Capture (i.e. timing value), an interrupt is generated and timer stops counting.

Steps of setting Timer1 for Mode 1 is taken as an example.

Step 1 Set initial Tick value of Timer1

Set Initial value of Tick via registers TMR_TICK1_0 ~ TMR_TICK1_3 (address 0x634 ~ 0x637). Address 0x634 is lowest byte and 0x637 is highest byte. It's recommended to clear initial Timer Tick value to 0.

Step 2 Set Capture value of Timer1

Set registers TMR_CAPT1_0 ~ TMR_CAPT1_3 (address 0x628 ~ 0x62b). Address 0x628 is lowest byte and 0x62b is highest byte.

Step 3 Select GPIO source and edge for Timer1

Select certain GPIO to be the clock source via setting "m1" register.

Select positive edge or negative edge of GPIO input to trigger Timer1 Tick increment via setting "Polarity" register.

Step 4 Set Timer1 to Mode 1 and enable Timer1

Set address 0x620[5:4] to 2'b01 to select Mode 1; Meanwhile set address 0x620[3] to 1'b1 to enable Timer1. Timer1 starts counting upward, and Timer1 Tick value is increased by 1 on each positive/negative (specified during Step 3) edge of GPIO until it reaches Timer1 Capture value.

5.1.4 Mode 2 (GPIO Pulse Width Mode)

In Mode 2, system clock is employed as the unit to measure the width of GPIO pulse. The "m0"/"m1"/"m2" register specifies the GPIO which generates control signal for Timer0/Timer1/Timer2.

After Timer is enabled, Timer Tick is triggered by a positive/negative (configurable) edge of GPIO pulse. Then Timer Tick (i.e. counting value) is increased by 1 on each positive edge of system clock from preset initial Tick value. Generally the initial Tick value is set to 0. The "Polarity" register specifies the GPIO edge when Timer Tick starts counting.

NOTE: Refer to Section 7.1.3 *for corresponding "**m0**", "**m1**", "**m2**" and "**Polarity**" register address.*

While a negative/positive edge of GPIO pulse is detected, an interrupt is generated and timer stops counting. The GPIO pulse width could be calculated in terms of tick count and period of system clock.

Steps of setting Timer2 for Mode 2 are taken as an example.

Step 1 Set initial Timer2 Tick value

Set Initial value of Tick via registers TMR_TICK2_0 ~ TMR_TICK2_3 (address 0x638 ~ 0x63b). Address 0x638 is lowest byte and 0x63b is highest byte. It's recommended to clear initial Timer Tick value to 0.

Step 2 Select GPIO source and edge for Timer2

Select certain GPIO to be the clock source via setting "m2" register.

Select positive edge or negative edge of GPIO input to trigger Timer2 counting start via setting "Polarity" register.

Step 3 Set Timer2 to Mode 2 and enable Timer2

Set address 0x620[7:6] to 2'b01 and address 0x621 [0] to 1'b1.

Timer2 Tick is triggered by a positive/negative (specified during Step 2) edge of GPIO pulse. Timer2 starts counting upward and Timer2 Tick value is increased by 1 on each positive edge of system clock.

While a negative/positive edge of GPIO pulse is detected, an interrupt is generated and Timer2 tick stops.

Step 4 Read current Timer2 Tick value to calculate GPIO pulse width

Read current Timer2 Tick value from address 0x638 ~ 0x63b.

Then GPIO pulse width is calculated as follows:

$$GPIO\ pulse\ width = System\ clock\ period * (current\ Timer2\ Tick - intial\ Timer2\ Tick)$$

For initial Timer2 Tick value is set to the recommended value of 0, then:

$$GPIO\ pulse\ width = System\ clock\ period * current\ Timer2\ Tick$$

5.1.5 Mode 3 (Tick Mode)

In Mode 3, system clock is employed.

After Timer is enabled, Timer Tick starts counting upward, and Timer Tick value is increased by 1 on each positive edge of system clock.

This mode could be used as time indicator. There will be no interrupt generated. Timer Tick keeps rolling from 0 to 0xffffffff. When Timer tick overflows, it returns to 0 and starts counting upward again.

Steps of setting Timer0 for Mode 3 is taken as an example.

Step 1 Set initial Tick value of Timer0

Set Initial value of Tick via address 0x630 ~ 0x633. Address 0x630 is lowest byte and address 0x633 is highest byte. It's recommended to clear initial Timer Tick value to 0.

Step 2 Set Timer0 to Mode 3 and enable Timer0

Set address 0x620[2:1] to 2'b11 to select Mode 3, meanwhile set address 0x620[0] to 1'b1 to enable Timer0. Timer0 Tick starts to roll.

Step 3 Read current Timer0 Tick value

Current Timer0 Tick value can be read from address 0x630 ~ 0x633.

5.1.6 Watchdog Timer

Programmable watchdog could reset chip from unexpected hang up or malfunction.

Only Timer2 supports Watchdog.

Timer2 Tick has 32 bits. Watchdog Capture has only 14 bits, which consists of TMR_CTRL2 (address 0x622) [6:0] as higher bits and TMR_CTRL1 (address 0x621) [7:1] as lower bits. Chip will be reset when the Timer2 Tick[31:18] matches Watch dog capture.

Step 1 Clear Timer2 Tick value

Clear registers TMR_TICK2_0 ~TMR_TICK2_3 (address 0x638 ~ 0x63b). Address 0x638 is lowest byte and 0x63b is highest byte.

Step 2 Enable Timer2

Set register TMR_CTRL0 (address 0x620) [6] to 1'b1 to enable Timer2.

Step 3 Set 14-bit Watchdog Capture value and enable Watchdog

Set address 0x622[6:0] as higher bits of watchdog capture and 0x621[7:1] as lower bits. Meanwhile set address 0x622[7] to 1'b1 to enable Watchdog.

Then Timer2 Tick starts counting upwards from 0.

If bits[31:18] of Timer2 Tick value read from address 0x638 ~ 0x63b reaches watchdog capture, the chip will be reset, and the status bit in address 0x72[0] will be set as 1'b1 automatically. User can read the watchdog status bit after chip reset to check if the reset source is watchdog, and needs to write 1'b1 to this bit to manually clear the flag.

5.2 32K LTIMER

The TLSR8208 also supports a low frequency (32 kHz) LTIMER in suspend mode or deep sleep mode. This timer can be used as one kind of wakeup source.

5.3 System Timer

The TLSR8208 also supports a System Timer. As introduced in Section 4.3.1, the clock frequency for System Timer is fixed as 16 MHz irrespective of system clock.

In Suspend mode, both System Timer and Timer0 ~ Timer2 stop counting, and 32k Timer starts counting. When the chip restores to Active mode, Timer0 ~ Timer2 will continue counting from the number when they

stops; in contrast, System Timer will continue counting from an adjusted number which is a sum of the number when it stops and an offset calculated from the counting value of 32k Timer during Suspend mode.

Table 5-2 Register Table for System Timer

Address	R/W	Description	Default Value
0x740	RW	[7:3] Byte 0 of system timer counter, write to set initial value. This is the system timer counter. The sys_timer is running @16 MHz, The [2:0] is invalid, therefore, the resolution is 0.5 μs.	0x00
0x741	RW	[7:0] Byte 1 of system timer counter, write to set initial value. This is the system timer counter.	0x00
0x742	RW	[7:0] Byte 2 of system timer counter, write to set initial value. This is the system timer counter.	0x00
0x743	RW	[7:0] Byte 3 of system timer counter, write to set initial value. This is the system timer counter.	0x00
0x744	RW	Byte 0 Of system timer counter pulse irq trig value	0xf0
0x745	RW	Byte 1 Of system timer counter pulse irq trig value	0x0f
0x746	RW	Byte 2 Of system timer counter pulse irq trig value	0x0f
0x747	RW	Byte 3 Of system timer counter pulse irq trig value	0x0e
0x748	RW	[2] level irq mask [1] read 32k timer done irq mask [0] calibration done irq mask	0x00
0x749	W1C	[1] read 32k timer done irq status [0] calibration done irq status	0x00

Address	R/W	Description	Default Value
0x74a	RW	[7:4] 32 kHz clock calibration mode (cycles of 32k clock) 4'h0: 65536 (2048 ms) 4'h1: 32768 (1024 ms) 4'h2: 16384 (512 ms) 4'h3: 8192 (256 ms) 4'h4: 4096 (128 ms) 4'h5: 2048 (64 ms) 4'h6: 1024 (32 ms) 4'h7: 512 (16 ms) 4'h8: 256 (8 ms) 4'h9: 128 (4 ms) 4'ha: 64 (2 ms) 4'hb: 32 (1 ms) 4'hc: 16 (500 µs) 4'hd: 8 (250 µs) 4'he: 4 (125 µs) 4'hf: 2 (62.5 µs) [3] calibration enable [2] timer auto mode [1] enable system timer [0] write/read mode of 32 kHz timer 1'b1: write; 1'b0: read	0xc1
0x74b	-	[6] read busy status (R) [5] read update status (W1C) [4] state machine status[1] (R) [3] W: Start 32k count write/read; R: state machine status[0] (RW) [2] cmd_set_tgl (R) [1] W: Stop 16m systimer when using auto mode; R: cmd_sync_tgl (RW) [0] W: Run 16m systimer when using auto mode; R: timer_en status (RW)	0x00
0x74c	RW	Byte 0 of 32 kHz Timer write value	0x00
0x74d	RW	Byte 1 of 32 kHz Timer write value	0x00

Address	R/W	Description	Default Value
0x74e	RW	Byte 2 of 32 kHz Timer write value	0x00
0x74f	RW	Byte 3 of 32 kHz Timer write value	0x00
0x750	R	Byte 0 of 32 kHz Timer read value	0x00
0x751	R	Byte 1 of 32 kHz Timer read value	0x00
0x752	R	Byte 2 of 32 kHz Timer read value	0x00
0x753	R	Byte 3 of 32 kHz Timer read value	0x00
0x754	R	Byte 0 of 32 kHz clock calibration result (representing 16 MHz clock cycle number)	0x00
0x755	R	Byte 1 of 32 kHz clock calibration result (representing 16 MHz clock cycle number)	0x00
0x756	R	Byte 2 of 32 kHz clock calibration result (representing 16 MHz clock cycle number)	0x00
0x757	R	Byte 3 of 32 kHz clock calibration result (representing 16 MHz clock cycle number)	0x00

6 Interrupt System

6.1 Interrupt Structure

The interrupt function is applied to manage dynamic program sequencing based on real-time events triggered by timers, pins and etc.

For the TLSR8208, there are 24 interrupt sources in all: 16 types are level-triggered interrupt sources (listed in address 0x640 ~ 0x641), and 8 types are edge-triggered interrupt sources (listed in address 0x642).

When CPU receives an interrupt request (IRQ) from certain interrupt source, it will determine whether to respond to the IRQ. If CPU decides to respond, it pauses current routine and starts to execute interrupt service subroutine. Program will jump to certain code address and execute IRQ handling commands. After finishing interrupt service subroutine, CPU returns to the breakpoint and continues to execute main function.

6.2 Register Configuration

Table 6-1 Register Table for Interrupt System

Address	R/W	Description	Default Value
0x640	RW	Byte 0 interrupt mask, level-triggered type {irq_mix, irq_uart, irq_dfifo, irq_dma, usb_pwdn, time2, time1, time0} [7] irq_mix, i.e. irq_host_cmd [6] irq_uart [5] irq_dfifo [4] irq_dma [3] usb_pwdn [2] time2 [1] time1 [0] time0	0x00

Address	R/W	Description	Default Value
0x641	RW	Byte 1 interrupt mask, level-triggered type {irq_gpio_group, irq_pwm, irq_zb_rt, irq_udc[4:0]} [7] irq_gpio_group [6] irq_pwm [5] irq_zb_rt [4] irq_udc[4] [3] irq_udc[3] [2] irq_udc[2] [1] irq_udc[1] [0] irq_udc[0]	0x00
0x642	RW	Byte 2 interrupt mask, edge-triggered type {RSVD, gpio2risc[2:0], irq_stimer, pm_irq, irq_gpio, usb_reset, usb_250μs} [7] gpio2risc[2] [6] gpio2risc[1] [5] gpio2risc[0] [4] irq_stimer [3] pm_irq_tm [2] irq_gpio [1] usb_reset [0] usb_250μs	0x00
0x643	RW	[0] Interrupt enable [1] Reserved (Multi-address enable)	0x00
0x644	RW	Byte 0 of priority 1: High priority; 0: Low priority	0x00
0x645	RW	Byte 1 of priority	0x00
0x646	RW	Byte 2 of priority	0x00
0x648	R	Byte 0 of interrupt source	0x00
0x649	R	Byte 1 of interrupt source	0x00
0x64a	R	Byte 2 of interrupt source	0x00

6.2.1 Enable/Mask Interrupt Sources

Various interrupt sources could be enabled or masked by the registers MASK_0 ~ MASK_2 (address 0x640 ~ 0x642).

Interrupt sources of level-triggered type:

- irq_mix (0x640[7]): I2C Slave mapping mode or SPI Slave interrupt (irq_host_cmd)
- irq_uart (0x640[6]): UART interrupt
- irq_dfifo (0x640[5]): DFIFO interrupt
- irq_dma (0x640[4]): DMA interrupt
- usb_pwdn (0x640[3]): USB Host has sent power down signal
- time2, time1, time0 (0x640[2] ~ 0x640[0]): Timer2 ~ Timer0 interrupt
- irq_gpio_group (0x641[7]): GPIO group interrupt, please refer to Section 7.1.3
- irq_pwm (0x641[6]): PWM interrupt
- irq_zb_rt (0x641[5]): Baseband interrupt
- irq_udc[4:0] (0x641[4:0]): USB device interrupt

Interrupt sources of edge-triggered type:

- gpio2risc[2:0] (0x642[7] ~ 0x642[5]): gpio2risc[2] ~ gpio2risc[0] interrupt, please refer to Section 7.1.3.
- irq_stimer (0x642[4]): System timer interrupt
- pm_irq_tm (0x642[3]): 32 kHz timer wakeup interrupt
- irq_gpio (0x642[2]): GPIO interrupt, please refer to Section 7.1.3
- usb_reset (0x642[1]): USB Host has sent reset command
- usb_250us (0x642[0]): USB has been in idle status for 250 μs

6.2.2 Interrupt Mode and Priority

Interrupt mode is typically-used mode. Register IRQMODE (address 0x643)[0] should be set as 1'b1 to enable interrupt function.

IRQ tasks could be set as High or Low priority via the registers PRIO_0 ~ PRIO_2 (address 0x644 ~ 0x646). When two or more interrupt sources assert interrupt requests at the same time, CPU will respond depending on respective interrupt priority levels. It's recommended not to modify priority setting.

6.2.3 Interrupt Source Flag

Three bytes in the registers IRQSRC_0 ~ IRQSRC_2 (address 0x648 ~ 0x64a) serve to indicate IRQ sources. Once IRQ occurs from certain source, the corresponding IRQ source flag will be set as "1". User could identify IRQ source by reading address 0x648 ~ 0x64a.

When handling edge-triggered type interrupt, the corresponding IRQ source flag needs to be cleared via address 0x64a. Take the interrupt source usb_250μs for example: First enable the interrupt source by setting address 0x642 bit[0] as 1'b1; then set address 0x643 bit[0] as 1'b1 to enable the interrupt. In interrupt handling function, 24-bit data is read from address 0x648 ~ 0x64a to check which IRQ source is valid; if data bit[16] is 1, it means the usb_250μs IRQ source is valid. Clear this interrupt source by setting address 0x64a bit[0] as 1'b1.

As for level-type interrupt, IRQ interrupt source status needs to be cleared by setting corresponding module status register. Take Timer0 IRQ interrupt source for example: First enable the interrupt source by setting address 0x640 bit[0] as 1'b1; then set address 0x643 bit[0] as 1'b1 to enable the interrupt. In interrupt handling function, 24-bit data is read from address 0x648~0x64a to check which IRQ source is valid; if data

bit[0] is 1, it means the Timer0 IRQ source is valid. Register TMR_STATUS (address 0x623) [0] should be written with 1'b1 to manually clear Timer0 status (refer to Section 5.1.1).

7 Interface

7.1 GPIO

The TLSR8208A supports up to 38 GPIOs, TLSR8208B supports up to 16 GPIOs, TLSR8208C supports up to 33 GPIOs, TLSR8208D supports up to 10 GPIOs. All digital IOs can be used as general purpose IOs.

All GPIOs have configurable pull-up/pull-down resistor. Please refer to Section 7.1.4 for details.

7.1.1 Basic Configuration

7.1.1.1 GPIO Lookup Table

Table 7-1 GPIO PAD Function Mux

Pad	Default	Register = [35:3]	Register = 1	Register = 0	register
PA[0]	GPIO	All functions[a]	PA_KS0_IO	UART_CTS_I	0x548[5:0]
PA[1]	DM	-	UART_RX_I	DM_IO	0x549[5:0]
PA[2]	DP(SWS)	-	UART_TX	DP_IO	0x54a[5:0]
PA[3]	SWS	-	-	SWS_IO	0x54b[5:0]
PA[4]	GPIO	All functions	PA_KS4_IO	SWM_IO	0x54c[5:0]
PA[5]	GPIO	All functions	PA_KS5_IO	UART_RTS	0x54d[5:0]
PA[6]	GPIO	All functions	PA_KS6_IO	UART_TX	0x54e[5:0]
PA[7]	GPIO	All functions	PA_KS7_IO	UART_RX_I	0x54f[5:0]
PB[0]	SPI_CN	-	PB_KS0_IO	SPI_CN_IO	0x550[5:0]
PB[1]	SPI_CK	-	PB_KS1_IO	SPI_CK_IO	0x551[5:0]
PB[2]	GPIO	All functions	PB_KS2_IO	PWM0	0x552[5:0]
PB[3]	SPI_IO2	-	PB_KS3_IO	SPI_IO2_IO	0x553[5:0]
PB[4]	GPIO	All functions	PB_KS4_IO	PWM0_N	0x554[5:0]
PB[5]	GPIO	All functions	PB_KS5_IO	PWM1	0x555[5:0]
PB[6]	GPIO	All functions	PB_KS6_IO	PWM2	0x556[5:0]
PB[7]	GPIO	All functions	PB_KS7_IO	PWM3	0x557[5:0]
PC[0]	GPIO	All functions	PC_KS0_IO	ANT_SEL0	0x558[5:0]
PC[1]	GPIO	All functions	PC_KS1_IO	ANT_SEL1	0x559[5:0]

Pad	Default	Register = [35:3]	Register = 1	Register = 0	register
PC[2]	GPIO	All functions	PC_KS2_IO	ANT_SEL2	0x55a[5:0]
PC[3]	GPIO	All functions	PC_KS3_IO	BLE_ACTIVITY	0x55b[5:0]
PC[4]	GPIO	All functions	PC_KS4_IO	BLE_STATUS	0x55c[5:0]
PC[5]	GPIO	All functions	PC_KS5_IO	WIFI_DENY_I	0x55d[5:0]
PC[6]	GPIO	All functions	PC_KS6_IO	RX_CYC2LNA	0x55e[5:0]
PC[7]	GPIO	All functions	PC_KS7_IO	TX_CYC2PA	0x55f[5:0]
PD[0]	GPIO	All functions	PD_KS0_IO	I2C_SCL_IO	0x560[5:0]
PD[1]	GPIO	All functions	PD_KS1_IO	I2C_SDA_IO	0x561[5:0]
PD[2]	GPIO	All functions	PD_KS2_IO	PWM4	0x562[5:0]
PD[3]	GPIO	All functions	PD_KS3_IO	PWM5	0x563[5:0]
PD[4]	SPI_IO3	-	PD_KS4_IO	SPI_IO3_IO	0x564[5:0]
PD[5]	GPIO	All functions	PD_KS5_IO	PWM1_N	0x565[5:0]
PD[6]	GPIO	All functions	PD_KS6_IO	CLK_7816	0x566[5:0]
PD[7]	GPIO	All functions	PD_KS7_IO	UART_RTX_IO	0x567[5:0]
PE[0]	MSDO	-	-	MOSI_IO	0x568[5:0]
PE[1]	MCLK	-	-	MCLK	0x569[5:0]
PE[2]	MSCN	-	-	MSCN	0x56a[5:0]
PE[3]	MSDI	-	-	MISO_IO	0x56b[5:0]
PF[0]	SPI_MOSI	-	PA_KS1_IO	SPI_MOSI_IO	0x56c[5:0]
PF[1]	SPI_MISO	-	PA_KS2_IO	SPI_MISO_IO	0x56d[5:0]

a. "All functions" include 32 functions: WIFI_DENY_I, BLE_STATUS, BLE_ACTIVITY, SPI_CN_IO, SPI_CK_IO, SPI_MOSI_IO, SPI_MISO_IO, SWM_IO, TX_CYC2PA, RX_CYC2LNA, ANT_SEL2, ANT_SEL1, ANT_SEL0, UART_RTX_IO, CLK_7816, I2C_SDA_IO, I2C_SCL_IO, UART_RX_I, UART_TX, UART_RTS, UART_CTS_I, PWM5_N, PWM4_N, PWM3_N, PWM2_N, PWM1_N, PWM0_N, PWM5, PWM4, PWM3, PWM2, PWM1, PWM0

NOTE:

- The initial status of PB[0], PB[1], PB[3], PD[4], PF[0] and PF[1] at power up or after sleep wake-up is SPI function, so these pins are not recommend to use as wakeup source.

Table 7-2 GPIO Setting

Pad	Input	IE	OEN	Output/PE	Polarity	DS	Act as GPIO
PA[0]	0x500[0]	0x501[0]	0x502[0]	0x503[0]	0x504[0]	0x505[0]	0x506[0]
PA[1]	0x500[1]	0x501[1]	0x502[1]	0x503[1]	0x504[1]	0x505[1]	0x506[1]
PA[2]	0x500[2]	0x501[2]	0x502[2]	0x503[2]	0x504[2]	0x505[2]	0x506[2]
PA[3]	0x500[3]	0x501[3]	0x502[3]	0x503[3]	0x504[3]	0x505[3]	0x506[3]
PA[4]	0x500[4]	0x501[4]	0x502[4]	0x503[4]	0x504[4]	0x505[4]	0x506[4]
PA[5]	0x500[5]	0x501[5]	0x502[5]	0x503[5]	0x504[5]	0x505[5]	0x506[5]
PA[6]	0x500[6]	0x501[6]	0x502[6]	0x503[6]	0x504[6]	0x505[6]	0x506[6]
PA[7]	0x500[7]	0x501[7]	0x502[7]	0x503[7]	0x504[7]	0x505[7]	0x506[7]
PB[0]	0x508[0]	0x509[0]	0x50a[0]	0x50b[0]	0x50c[0]	0x50d[0]	0x50e[0]
PB[1]	0x508[1]	0x509[1]	0x50a[1]	0x50b[1]	0x50c[1]	0x50d[1]	0x50e[1]
PB[2]	0x508[2]	0x509[2]	0x50a[2]	0x50b[2]	0x50c[2]	0x50d[2]	0x50e[2]
PB[3]	0x508[3]	0x509[3]	0x50a[3]	0x50b[3]	0x50c[3]	0x50d[3]	0x50e[3]
PB[4]	0x508[4]	0x509[4]	0x50a[4]	0x50b[4]	0x50c[4]	0x50d[4]	0x50e[4]
PB[5]	0x508[5]	0x509[5]	0x50a[5]	0x50b[5]	0x50c[5]	0x50d[5]	0x50e[5]
PB[6]	0x508[6]	0x509[6]	0x50a[6]	0x50b[6]	0x50c[6]	0x50d[6]	0x50e[6]
PB[7]	0x508[7]	0x509[7]	0x50a[7]	0x50b[7]	0x50c[7]	0x50d[7]	0x50e[7]
PC[0]	0x510[0]	0xc0[0]	0x512[0]	0x513[0]/ 0xc1[0]	0x514[0]	0xc2[0]	0x516[0]
PC[1]	0x510[1]	0xc0[1]	0x512[1]	0x513[1]/ 0xc1[1]	0x514[1]	0xc2[1]	0x516[1]
PC[2]	0x510[2]	0xc0[2]	0x512[2]	0x513[2]/ 0xc1[2]	0x514[2]	0xc2[2]	0x516[2]
PC[3]	0x510[3]	0xc0[3]	0x512[3]	0x513[3]/ 0xc1[3]	0x514[3]	0xc2[3]	0x516[3]
PC[4]	0x510[4]	0xc0[4]	0x512[4]	0x513[4]/ 0xc1[4]	0x514[4]	0xc2[4]	0x516[4]
PC[5]	0x510[5]	0xc0[5]	0x512[5]	0x513[5]/ 0xc1[5]	0x514[5]	0xc2[5]	0x516[5]

Pad	Input	IE	OEN	Output/PE	Polarity	DS	Act as GPIO
PC[6]	0x510[6]	0xc0[6]	0x512[6]	0x513[6]/ 0xc1[6]	0x514[6]	0xc2[6]	0x516[6]
PC[7]	0x510[7]	0xc0[7]	0x512[7]	0x513[7]/ 0xc1[7]	0x514[7]	0xc2[7]	0x516[7]
PD[0]	0x518[0]	0x519[0]	0x51a[0]	0x51b[0]	0x51c[0]	0x51d[0]	0x51e[0]
PD[1]	0x518[1]	0x519[1]	0x51a[1]	0x51b[1]	0x51c[1]	0x51d[1]	0x51e[1]
PD[2]	0x518[2]	0x519[2]	0x51a[2]	0x51b[2]	0x51c[2]	0x51d[2]	0x51e[2]
PD[3]	0x518[3]	0x519[3]	0x51a[3]	0x51b[3]	0x51c[3]	0x51d[3]	0x51e[3]
PD[4]	0x518[4]	0x519[4]	0x51a[4]	0x51b[4]	0x51c[4]	0x51d[4]	0x51e[4]
PD[5]	0x518[5]	0x519[5]	0x51a[5]	0x51b[5]	0x51c[5]	0x51d[5]	0x51e[5]
PD[6]	0x518[6]	0x519[6]	0x51a[6]	0x51b[6]	0x51c[6]	0x51d[6]	0x51e[6]
PD[7]	0x518[7]	0x519[7]	0x51a[7]	0x51b[7]	0x51c[7]	0x51d[7]	0x51e[7]
PE[0]	0x520[0]	0x521[0]	0x522[0]	0x523[0]	-	0x525[0]	0x526[0]
PE[1]	0x520[1]	0x521[1]	0x522[1]	0x523[1]	-	0x525[1]	0x526[1]
PE[2]	0x520[2]	0x521[2]	0x522[2]	0x523[2]	-	0x525[2]	0x526[2]
PE[3]	0x520[3]	0x521[3]	0x522[3]	0x523[3]	-	0x525[3]	0x526[3]
PF[0]	0x528[0]	0x529[0]	0x52a[0]	0x52b[0]	0x52c[0]	0x52d[0]	0x52e[0]
PF[1]	0x528[1]	0x529[1]	0x52a[1]	0x52b[1]	0x52c[1]	0x52d[1]	0x52e[1]

NOTE:

- IE: Input enable, high active. 1: enable input, 0: disable input.
- OEN: Output enable, low active. 0: enable output, 1: disable output.
- Register: See Table 7-2 for configuration of multiplexed functions.
- Output: Configure GPO output.
- Input: Read GPI input.
- DS: Drive strength. 1: maximum DS level (default), 0: minimal DS level.
- Act as GPIO: Enable (1) or disable (0) GPIO function.
- Polarity: See Section 7.1.3.
- Priority: "Act as GPIO" has the highest priority. To configure as multiplexed function, disable GPIO function first.
- 0xc0, 0xc1, and 0xc2 are analog registers; others are digital registers.
- For all unused GPIOs, corresponding "IE" must be set as 0.
- To use SAR ADC pin function, please refer to the corresponding module section.

7.1.1.2 Multiplexed Functions

Each pin listed in Table 7-2 acts as the function in the "Default Function" column by default.

- PA[1] acts as DM function by default.
- PA[2] acts as DP(SWS) function by default.
- PA[3] acts as SWS function by default.
- PB[0] acts as SPI_CN function by default.
- PB[1] acts as SPI_CK function by default.
- PB[3] acts as SPI_IO2 function by default.
- PD[4] acts as SPI_IO3 function by default.
- PE[0] acts as MSDO function by default.
- PE[1] acts as MCLK function by default.
- PF[0] acts as SPI_MOSI function by default.
- PF[1] acts as SPI_MISO function by default.
- The other digital IOs act as GPIO function by default.

If a pin with multiplexed functions does not act as GPIO function by default, to use it as GPIO, first set the bit in "Act as GPIO" column in as 1'b1. After GPIO function is enabled, if the pin is used as output, both the bits in "IE" and "OEN" columns should be set as 1'b0, then set the register value in the "Output" column; if the pin is used as input, both the bits in "IE" and "OEN" columns should be set as 1'b1, and the input data can be read from the register in the "Input" column.

To use a pin as certain multiplexed function (neither the default function nor GPIO function), first clear the bit in "Act as GPIO" column to disable GPIO function, and then configure "Register" column to enable multiplexed function correspondingly.

7.1.1.3 Drive Strength

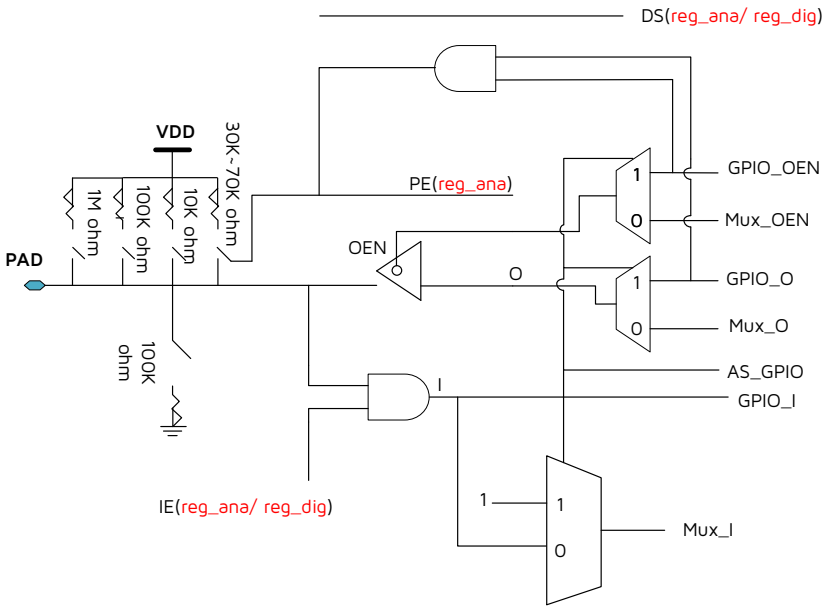
The registers in the "DS" column are used to configure the corresponding pin's driving strength: "1" indicates maximum drive level, while "0" indicates minimal drive level.

The "DS" configuration will take effect when the pin is used as output. It's set as the strongest driving level by default. In actual applications, driving strength can be decreased to lower level if necessary.

- PA[0,4:7], PC[0:7], PD[0:7], PE[0:3], and PF[0:1]: maximum = 4 mA ("DS" = 1), minimum = 2 mA ("DS" = 0)
- PA[1:3], and PB[0:7]: maximum = 8 mA ("DS" = 1), minimum = 4 mA ("DS" = 0)

7.1.2 GPIO Logic Introduction

Figure 7-1 GPIO Logic Diagram



In the figure above,

1. DS: drive strength, 1: high drive strength; 0: low drive strength
2. PE: pull-up enable, 1: pull up; 0: no pull up
3. OEN: output enable, 1: high Z; 0: output
4. O: output value, when OEN is 0, output this value
5. I: input value
6. IE: input enable, if IE is 0, C is always zero
7. 1M, 10K,pull up and 100K pull down resistors are controlled by analog 3.3V register controller

NOTE:

- When PAD is set as functional IO, no need to configure GPIO_OEN as the functional IO will enable Mux-_OEN.
- When PAD is input, IE should be enabled regardless of functional IO or GPIO, and output to I, AS_GPIO is 1, Mux_I is 1.
- There are two methods to configure digital pull-up of 30k~70k ohm:
  - PC group and PD group (may vary for different chips), pad can configure analog register PE and enable digital pull-up.
  - Other group of pad, when GPIO_OEN=1 and GPIO_I=1, it enables digital pull-up.
- Analog pull-up has three options: 1M, 100k, 10k ohm; analog pull-down has only 100k ohm. They can be configured via corresponding analog registers.
- The GPIO configuration sequence should be: configure the MUX function, and then disable GPIO function. If disable GPIO first and then set function, the default function of the pad may be enabled and will cause false output level.

7.1.3 Connection Relationship Between GPIO and Related Modules

GPIO can be used to generate GPIO interrupt signal for interrupt system, counting or control signal for Timer/Counter module, or GPIO2RISC interrupt signal for interrupt system.

For the "Exclusive Or (XOR)" operation result for input signal from any GPIO pin and respective "Polarity" value, on one hand, it takes "And" operation with "irq" and generates GPIO interrupt request signal; on the other hand, it takes "And" operation with "m0/m1/m2", and generates counting signal in Mode 1 or control signal in Mode 2 for Timer0/Timer1/Timer2, or generates GPIO2RISC[0]/GPIO2RISC[1]/GPIO2RISC[2] interrupt request signal.

GPIO interrupt request signal = | ((input ^ polarity) & irq);

Counting (Mode 1) or control (Mode 2) signal for Timer0 = | ((input ^ polarity) & m0);

Counting (Mode 1) or control (Mode 2) signal for Timer1 = | ((input ^ polarity) & m1);

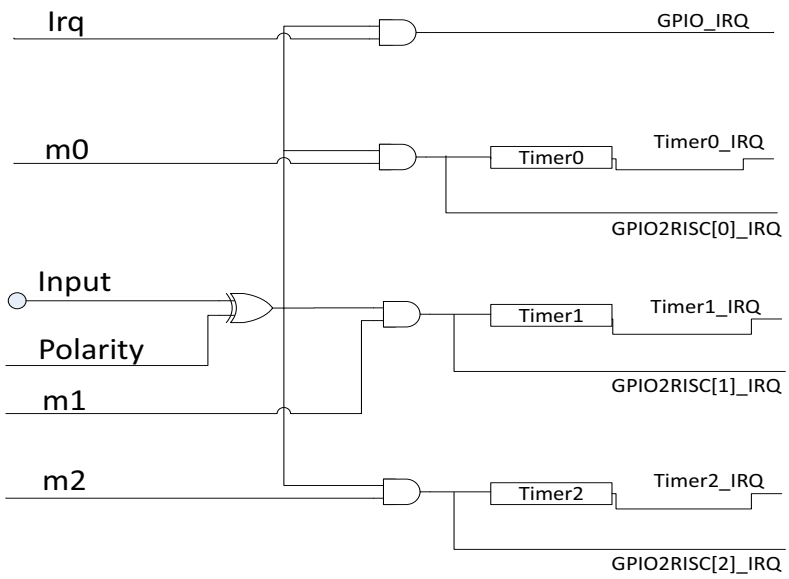
Counting (Mode 1) or control (Mode 2) signal for Timer2 = | ((input ^ polarity) & m2);

GPIO2RISC[0] interrupt request signal = | ((input ^ polarity) & m0);

GPIO2RISC[1] interrupt request signal = | ((input ^ polarity) & m1).

GPIO2RISC[2] interrupt request signal = | ((input ^ polarity) & m2).

Figure 7-2 Logic Relationship Between GPIO and Related Modules



Please refer to Table 7-3 and Table 6-1 to learn how to configure GPIO for interrupt system or Timer/Counter (Mode 1 or Mode 2).

Enable GPIO function

First enable GPIO function, enable IE and disable OEN. Please see Section 7.1.1.

GPIO IRQ signal:

Select GPIO interrupt trigger edge (positive edge or negative edge) via configuring "**Polarity**", and set corresponding GPIO interrupt enabling bit "**Irq**".

Then set address 0x574[2] (irq_enable) to enable GPIO IRQ.

Finally enable GPIO interrupt (irq_gpio) via address 0x642[2].

User can read addresses 0x578 ~ 0x57b to see which GPIO asserts GPIO interrupt request signal. Note: 0x578[7:0] --> PA[7] ~ PA[0], 0x579[7:0] --> PB[7] ~ PB[0], 0x57a[7:0] --> PC[7] ~ PC[0], 0x57b[7:0] --> PD[7] ~ PD[0].

GPIO IRQ GROUP signal:

Select a set of GPIOs from PA, PB, PC, PD, PF as interrupt source.

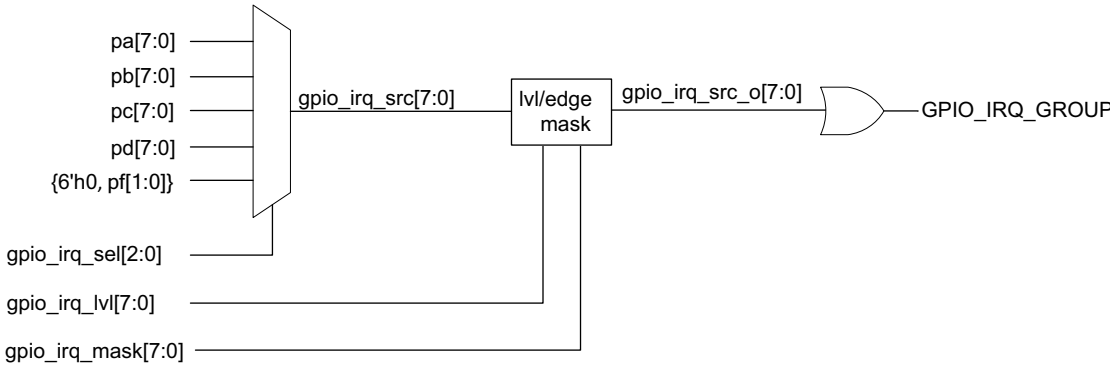
Set address 0x575 (gpio_irq_sel) to select a set of GPIOs.

Then set address 0x576 (gpio_irq_mask) to enable interrupts for the selected GPIO signals.

Finally set address 0x577 (gpio_irq_lvl) to set the type of interrupt to be edge sensitive or level sensitive.

User can read address 0x56f to see which GPIO asserts interrupt request signal and clear the interrupt through asserting the corresponding bit of 0x56f.

Figure 7-3 GPIO IRQ GROUP signal



Timer/Counter counting or control signal:

Configure "**Polarity**". In Timer Mode 1, it determines GPIO edge when Timer Tick counting increases. In Timer Mode 2, it determines GPIO edge when Timer Tick starts counting.

Then set "**m0/m1/m2**" to specify the GPIO which generates counting signal (Mode 1)/control signal (Mode 2) for Timer0/Timer1/Timer2.

User can read addresses 0x580 ~ 0x583/0x588 ~ 0x58b/0x590 ~ 0x593 to see which GPIO asserts counting signal (in Mode 1) or control signal (in Mode 2) for Timer0/Timer1/Timer2. Note: Timer0: 0x580[7:0] --> PA[7] ~ PA[0], 0x581[7:0] --> PB[7] ~ PB[0], 0x582[7:0] --> PC[7] ~ PC[0], 0x583[7:0] --> PD[7] ~ PD[0]; Timer1: 0x588[7:0] --> PA[7] ~ PA[0], 0x589[7:0] --> PB[7] ~ PB[0], 0x58a[7:0] --> PC[7] ~ PC[0], 0x58b[7:0] --> PD[7] ~ PD[0]; Timer2: 0x590[7:0] --> PA[7] ~ PA[0], 0x591[7:0] --> PB[7] ~ PB[0], 0x592[7:0] --> PC[7] ~ PC[0], 0x593[7:0] --> PD[7] ~ PD[0].

GPIO2RISC IRQ signal:

Select GPIO2RISC interrupt trigger edge (positive edge or negative edge) via configuring "**Polarity**", and set corresponding GPIO enabling bit "**m0**"/"**m1**"/"**m2**".

Enable GPIO2RISC[0]/GPIO2RISC[1]/GPIO2RISC[2] interrupt, i.e. "gpio2risc[0]" (address 0x642[5])/ "gpio2risc[1]"(address 0x642[6])/"gpio2risc[2]"(address 0x642[7]).

Table 7-3 GPIO IRQ Table

Pin	Input (R)	Polarity 1: Active Low 0: Active High	IRQ	m0	m1	m2
PA[0]	0x500[0]	0x504[0]	0x507[0]	0x530[0]	0x538[0]	0x540[0]
PA[1]	0x500[1]	0x504[1]	0x507[1]	0x530[1]	0x538[1]	0x540[1]
PA[2]	0x500[2]	0x504[2]	0x507[2]	0x530[2]	0x538[2]	0x540[2]
PA[3]	0x500[3]	0x504[3]	0x507[3]	0x530[3]	0x538[3]	0x540[3]
PA[4]	0x500[4]	0x504[4]	0x507[4]	0x530[4]	0x538[4]	0x540[4]
PA[5]	0x500[5]	0x504[5]	0x507[5]	0x530[5]	0x538[5]	0x540[5]

Pin	Input (R)	Polarity 1: Active Low 0: Active High	IRQ	m0	m1	m2
PA[6]	0x500[6]	0x504[6]	0x507[6]	0x530[6]	0x538[6]	0x540[6]
PA[7]	0x500[7]	0x504[7]	0x507[7]	0x530[7]	0x538[7]	0x540[7]
PB[0]	0x508[0]	0x50c[0]	0x50f[0]	0x531[0]	0x539[0]	0x541[0]
PB[1]	0x508[1]	0x50c[1]	0x50f[1]	0x531[1]	0x539[1]	0x541[1]
PB[2]	0x508[2]	0x50c[2]	0x50f[2]	0x531[2]	0x539[2]	0x541[2]
PB[3]	0x508[3]	0x50c[3]	0x50f[3]	0x531[3]	0x539[3]	0x541[3]
PB[4]	0x508[4]	0x50c[4]	0x50f[4]	0x531[4]	0x539[4]	0x541[4]
PB[5]	0x508[5]	0x50c[5]	0x50f[5]	0x531[5]	0x539[5]	0x541[5]
PB[6]	0x508[6]	0x50c[6]	0x50f[6]	0x531[6]	0x539[6]	0x541[6]
PB[7]	0x508[7]	0x50c[7]	0x50f[7]	0x531[7]	0x539[7]	0x541[7]
PC[0]	0x510[0]	0x514[0]	0x517[0]	0x532[0]	0x53a[0]	0x542[0]
PC[1]	0x510[1]	0x514[1]	0x517[1]	0x532[1]	0x53a[1]	0x542[1]
PC[2]	0x510[2]	0x514[2]	0x517[2]	0x532[2]	0x53a[2]	0x542[2]
PC[3]	0x510[3]	0x514[3]	0x517[3]	0x532[3]	0x53a[3]	0x542[3]
PC[4]	0x510[4]	0x514[4]	0x517[4]	0x532[4]	0x53a[4]	0x542[4]
PC[5]	0x510[5]	0x514[5]	0x517[5]	0x532[5]	0x53a[5]	0x542[5]
PC[6]	0x510[6]	0x514[6]	0x517[6]	0x532[6]	0x53a[6]	0x542[6]
PC[7]	0x510[7]	0x514[7]	0x517[7]	0x532[7]	0x53a[7]	0x542[7]
PD[0]	0x518[0]	0x51c[0]	0x51f[0]	0x533[0]	0x53b[0]	0x543[0]
PD[1]	0x518[1]	0x51c[1]	0x51f[1]	0x533[1]	0x53b[1]	0x543[1]
PD[2]	0x518[2]	0x51c[2]	0x51f[2]	0x533[2]	0x53b[2]	0x543[2]
PD[3]	0x518[3]	0x51c[3]	0x51f[3]	0x533[3]	0x53b[3]	0x543[3]
PD[4]	0x518[4]	0x51c[4]	0x51f[4]	0x533[4]	0x53b[4]	0x543[4]
PD[5]	0x518[5]	0x51c[5]	0x51f[5]	0x533[5]	0x53b[5]	0x543[5]
PD[6]	0x518[6]	0x51c[6]	0x51f[6]	0x533[6]	0x53b[6]	0x543[6]
PD[7]	0x518[7]	0x51c[7]	0x51f[7]	0x533[7]	0x53b[7]	0x543[7]

Pin	Input (R)	Polarity 1: Active Low 0: Active High	IRQ	m0	m1	m2
PE[0]	0x520[0]	-	-	0x534[0]	0x53c[0]	0x544[0]
PE[1]	0x520[1]	-	-	0x534[1]	0x53c[1]	0x544[1]
PE[2]	0x520[2]	-	-	0x534[2]	0x53c[2]	0x544[2]
PE[3]	0x520[3]	-	-	0x534[3]	0x53c[3]	0x544[3]
PF[0]	0x528[0]	0x52c[0]	0x52f[0]	0x535[0]	0x53d[0]	0x545[0]
PF[1]	0x528[1]	0x52c[1]	0x52f[1]	0x535[1]	0x53d[1]	0x545[1]

7.1.4 Pull-Up/Pull-Down Resistor

All GPIOs support configurable pull-up resistor of rank x1 and x100 or pull-down resistor of rank x10 which are all disabled by default. Analog registers afe_0x0e<7:0> ~ afe_0x16<3:0> serve to control the pull-up/pull-down resistor for each GPIO.

The DP pin also supports 1.5 kΩ pull-up resistor for USB use. The 1.5 kΩ pull up resistor is disabled by default and can be enabled by setting analog register afe_0x0b<7> as 1'b1. For the DP/PA[2] pin, user can only enable either 1.5 kΩ pull-up resistor or pull-up resistor of rank x1/x100 / pull-down resistor of rank x10 at the same time. Please refer to Table 7-4 for details.

Take the PA[3] for example: Setting analog register afe_0x0e<7:6> to 2'b01/2'b11/2'b10 is to respectively enable pull-up resistor of rank x100/pull-up resistor of rank x1/pull-down resistor of rank x10 for PA[3]; Clearing the two bits (default value) disables pull-up and pull-down resistor for PA[3].

Table 7-4 Analog Registers for Pull-Up/Pull-Down Resistor Control

Address	Name	Description	Default Value
afe_0x0b<7>	dp_pullup_res_3v	1.5k (typ.) pull-up resistor for USB DP PAD 0: disable 1: enable	0x00
Rank	**Typical value (depend on actual application)**		
x1	10 kOhm		
x10	100 kOhm		
x100	1 MOhm		

Address	Name	Description	Default Value
afe_0x0e<7:0>	a_sel<7:0>	PA[3:0] pull up and down select: <7:6>: PA[3] <5:4>: PA[2] <3:2>: PA[1] <1:0>: PA[0] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00
afe_0x0f<7:0>	a_sel<15:8>	PA[7:4] pull up and down select: <7:6>: PA[7] <5:4>: PA[6] <3:2>: PA[5] <1:0>: PA[4] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00
afe_0x10<7:0>	b_sel<7:0>	PB[3:0] pull up and down select: <7:6>: PB[3] <5:4>: PB[2] <3:2>: PB[1] <1:0>: PB[0] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00

Address	Name	Description	Default Value
afe_0x11<7:0>	b_sel<15:8>	PB[7:4] pull up and down select: <7:6>: PB[7] <5:4>: PB[6] <3:2>: PB[5] <1:0>: PB[4] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00
afe_0x12<7:0>	c_sel<7:0>	PC[3:0] pull up and down select: <7:6>: PC[3] <5:4>: PC[2] <3:2>: PC[1] <1:0>: PC[0] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00
afe_0x13<7:0>	c_sel<15:8>	PC[7:4] pull up and down select: <7:6>: PC[7] <5:4>: PC[6] <3:2>: PC[5] <1:0>: PC[4] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00

Address	Name	Description	Default Value
afe_0x14<7:0>	d_sel<7:0>	PD[3:0] pull up and down select: <7:6>: PD[3] <5:4>: PD[2] <3:2>: PD[1] <1:0>: PD[0] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00
afe_0x15<7:0>	d_sel<15:8>	PD[7:4] pull up and down select: <7:6>: PD[7] <5:4>: PD[6] <3:2>: PD[5] <1:0>: PD[4] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00
afe_0x16<3:0>	f_sel<3:0>	PF[3:0] pull up and down select: <3:2>: PF[1] <1:0>: PF[0] 00: Null 01: x100 pull up 10: x10 pull down 11: x1 pull up	0x00

7.2 SWM and SWS

The TLSR8208 supports Single Wire interface. SWM (Single Wire Master) and SWS (Single Wire Slave) represent the master and slave device of the single wire communication system developed by Telink. The maximum data rate can be up to 2 Mbps.

SWS usage is not supported in power-saving mode (Deep Sleep or Suspend).

7.2.1 Swire Through USB

The default function of PA[2] is DP. If swire_usb_en (swire_base+0x1[7]) = 1, when PA[2] (DP) and PA[1] (DM) receive a specific timing sequence (see Figure 7-5), swire_usb_sel will be set to 1, then the Swire slave data will switch to DP and PA[2] will switch to SWS function.

Figure 7-4 Swire Through USB Diagram

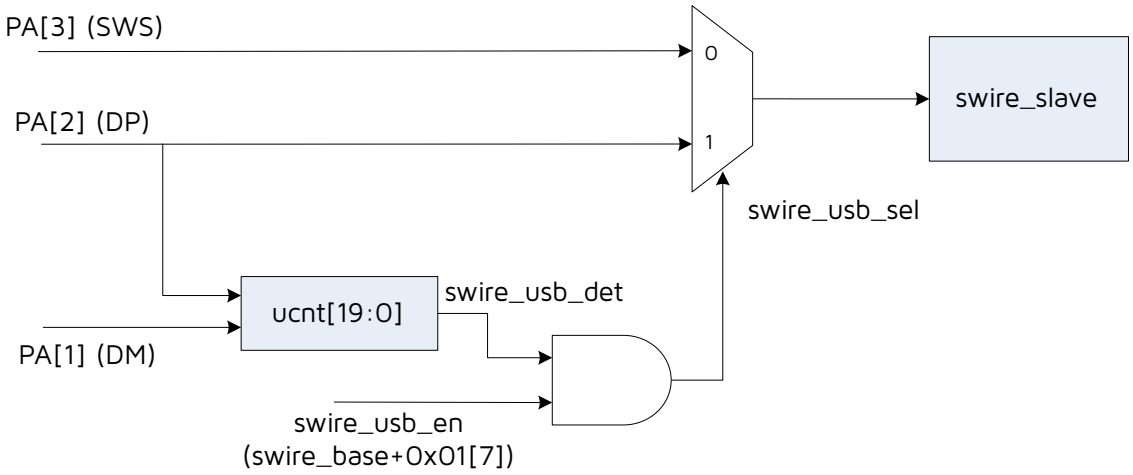
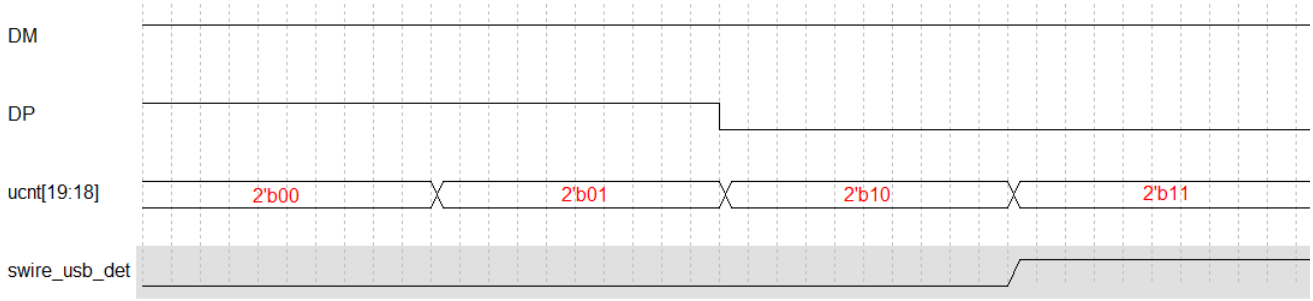


Figure 7-5 shows the timing sequence of enabling Swire through USB. DM should remain high all the time. DP should remain high until ucnt[19:18] = 2'b10, then DP switches to the low level and remains low until ucnt[19:18] = 2'b11, at which point swire_usb_det is set to 1. That is, assuming the system clock is 24M, then the timing sequence should be: DP remains high for about 22 ms and low for about 11 ms.

Figure 7-5 Timing Sequence of Enabling Swire Through USB



7.3 I2C

The TLSR8208 embeds I2C hardware module, which could act as Master mode or Slave mode. I2C is a popular inter-IC interface requiring only 2 bus lines, a serial data line (SDA) and a serial clock line (SCL).

7.3.1 Communication Protocol

Telink I2C module supports standard-mode (100 kbps) and fast-mode (400 kbps) with restriction that system clock must be by at least 10x of data rate.

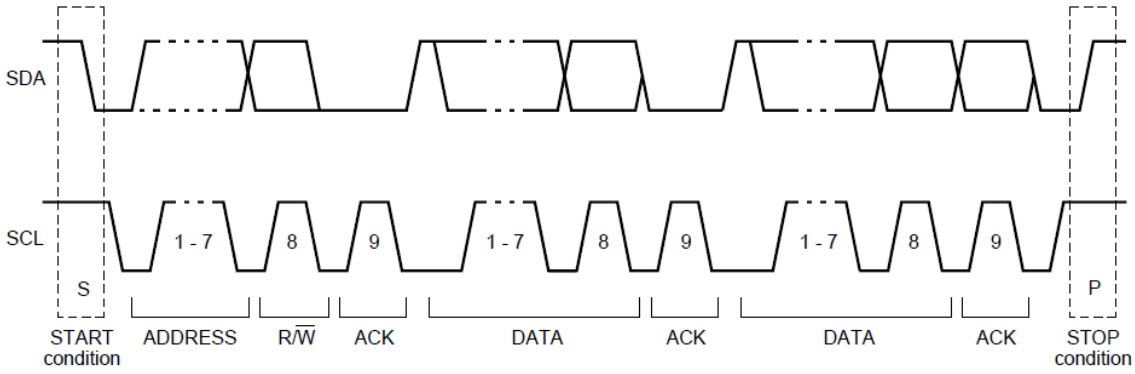
Two wires, SDA and SCL (SCK) carry information between Master device and Slave device connected to the bus. Each device is recognized by unique address (ID). Master device is the device which initiates a data

transfer on the bus and generates the clock signals to permit that transfer. Slave device is the device addressed by a Master.

Both SDA and SCL are bidirectional lines connected to a positive supply voltage via a pull-up resister. It's recommended to use external 3.3 kOhm pull-up resistor. For standard mode, the internal pull-up resistor of rank x1 can be used instead of the external 3.3 kOhm pull-up.

When the bus is free, both lines are HIGH. It's noted that data in SDA line must keep stable when clock signal in SCL line is at high level, and level state in SDA line is only allowed to change when clock signal in SCL line is at low level.

Figure 7-6 I2C Timing Chart



7.3.2 Register Table

Table 7-5 Register Configuration for I2C

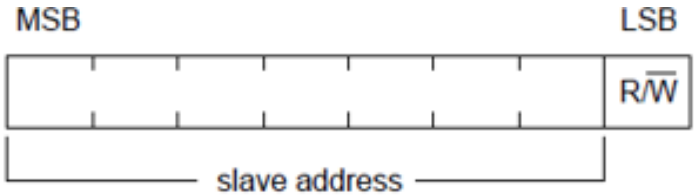
Address	R/W	Description	Default Value
0x00	RW	I2C master clock speed	0x1f
0x01	RW	[7:1]: I2C ID	0x5c
0x02	RW	[0]: master busy [1]: master packet busy [2]: master received status 0 for ACK; 1 for NAK	0x00
0x03	RW	[0]: address auto increase enable [1]: I2C master enable [2]: enable Mapping Mode [3]: r_clk_stretch_en, suspend transmission by pulling SCL down to low level, and continue transmission after SCL is released to high level	0x01
0x04	RW	[7:0]: Data buffer in master mode	0x5a
0x05	RW	[7:0]: Data buffer in master mode	0xf1

Address	R/W	Description	Default Value
0x06	RW	[7:0]: Data buffer for Read or Write in master mode	0x00
0x07	RW	[0]: launch ID cycle [1]: launch address cycle (send I2CAD data) [2]: launch data write cycle [3]: launch data read cycle For Master Write: 0: I2CAD & I2CDW, 1: I2CAD & I2CDW & I2CDR. To write 3 bytes: bit[3] = 1; to write 2 bytes: bit[3] = 0. For Master Read: always 1. [4]: launch start cycle [5]: launch stop cycle [6]: enable read ID [7]: enable ACK in read command	0x00
0xe0	R	[6:0]: I2C read address	0x00
0xe1	RW	Low byte of Mapping mode buffer address	0x80
0xe2	RW	Middle byte of Mapping mode buffer address	0xd7
0xe3	RW	High byte of Mapping mode buffer address	0x00
0xe4	RW	[0]: host_cmd_irq_o, I2C host operation has happened. Write 1 to clear. [1]: host_rd_tag_o, I2C host operation has happened and is read operation. Write 1 to clear.	0x00

7.3.3 I2C Slave Mode

I2C module of the TLSR8208 acts as Slave mode by default. I2C slave address can be configured via register I2C_ID (address 0x01) [7:1].

Figure 7-7 Byte Consisted of Slave Address and R/W Flag Bit



I2C Slave mode supports two sub modes including Direct Memory Access (DMA) mode and Mapping mode, which is selectable via address 0x03[2].

In I2C Slave mode, Master could initiate transaction anytime. I2C slave module will reply with ACK automatically. To monitor the start of I2C transaction, user could set interrupt from GPIO for SDA or SCL.

7.3.3.1 DMA Mode

In DMA mode, other devices (Master) could access (read/write) designated address in Register and/or SRAM of the TLSR8208 according to I2C protocol. I2C module of the TLSR8208 will execute the read/write command from I2C master automatically. But user needs to notice that the system clock shall be at least 10x faster than I2C bit rate.

The access address designated by Master is offset by 0x800000. In the TLSR8208, Register address starts from 0x800000 and SRAM address starts from 0x840000. For example, if Addr High (AddrH) is 0x04, Addr Middle (AddrM) is 0x00, and Addr Low (AddrL) is 0xcc, the real address of accessed data is 0x8400cc.

In DMA mode, Master could read/write data byte by byte. The designated access address is initial address and it supports auto increment by setting address 0x03[0] to 1'b1.

Figure 7-8 Read Format in DMA Mode

Read Format in DMA mode

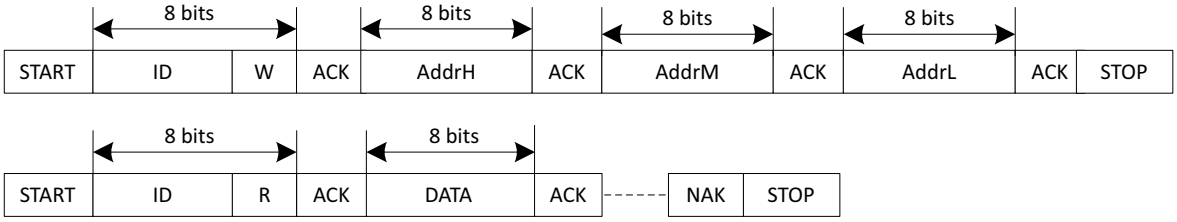


Figure 7-9 Write Format in DMA Mode

Write Format in DMA mode

7.3.3.2 Mapping Mode

Mapping mode could be enabled via setting register I2CSCT0 (address 0x03)[2] to 1'b1.

In mapping mode, data written and read by I2C master will be redirected to specified 128-byte buffer in SRAM. User could specify the initial address of the buffer by configuring registers HOSR_ADR_L (address 0xe1, lower byte), HOSR_ADR_M (address 0xe2, middle byte) and HOSR_ADR_H (address 0xe3, higher byte). The first 64-byte buffer is for written data and following 64-byte buffer is for read data. Every time the data access will start from the beginning of the Write-buffer/Read-buffer after I2C stop condition occurs. The last accessed data address could be checked in register I2CMAP_HADR (address 0xe0) [6:0] which is only updated after I2C STOP occurs.

Figure 7-10 Read Format in Mapping Mode

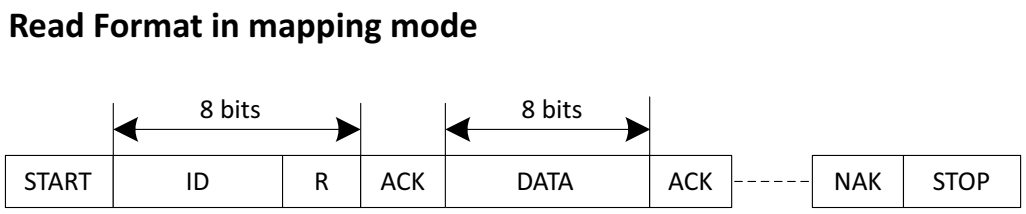
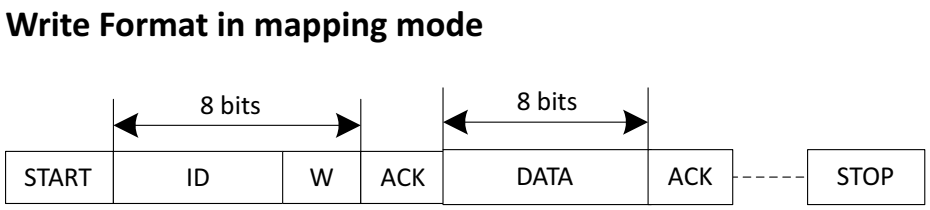


Figure 7-11 Write Format in Mapping Mode



7.3.4 I2C Master Mode

Address 0x03[1] should be set to 1'b1 to enable I2C master mode for the TLSR8208.

Address 0x00 serves to set I2C Master clock: F_{I2C} = (System Clock / (4 *clock speed configured in address 0x00).

A complete I2C protocol contains START, Slave Address, R/W bit, data, ACK and STOP. Slave address could be configured via address 0x01[7:1].

I2C Master (i.e. I2C module of the TLSR8208) could send START, Slave Address, R/W bit, data and STOP cycle by configuring address 0x07. I2C master will send enabled cycles in the correct sequence.

Address 0x02 serves to indicate whether Master/Master packet is busy, as well as Master received status. Bit[0] will be set to 1 when one byte is being sent, and the bit can be automatically cleared after a start signal/ address byte/acknowledge signal/data /stop signal is sent. Bit[1] is set to 1 when the start signal is sent, and the bit will be automatically cleared after the stop signal is sent. Bit[2] indicates whether to succeed in sending acknowledgement signal.

7.3.4.1 I2C Master Write Transfer

I2C Master has 3-byte buffer for write data, which are I2CAD (0x04), I2CDW (0x05) and I2CDR (0x06). Write transfer will be completed by I2C master module.

For example, to implement an I2C write transfer with 3-byte data, which contains START, Slave Address, Write bit, ACK from Slave, 1st byte, ACK from Slave, 2nd byte, ACK from Slave, 3rd byte, ACK from Slave and STOP, user needs to configure I2C Slave Address to I2C_ID (0x01) [7:1], 1st byte data to I2CAD, 2nd byte data to I2CDW and 3rd byte to I2CDR. To start I2C write transfer, I2CSCT1 (0x07) is configured to 0x3f (0011 1111). I2C Master will launch START, Slave address, Write bit, load ACK to I2CMST (0x02) [2], send I2CAD data, load ACK to I2CMST[2], send I2CDW data, load ACK to I2CMST[2], send I2CDR data, load ACK to I2CMST[2] and then STOP sequentially.

For I2C write transfer whose data are more than 3 bytes, user could split the cycles according to I2C protocol.

7.3.4.2 I2C Master Read Transfer

I2C Master has one byte buffer for read data, which is I2CDR (0x06). Read transfer will be completed by I2C Master.

For example, to implement an I2C read transfer with 1 byte data, which contains START, Slave Address, Read bit, ACK from Slave, 1st byte from Slave, ACK by Master and STOP, user needs to configure I2C Slave address to I2C_ID (0x01) [7:1]. To start I2C read transfer, I2CSCT1 (0x07) is configured to 0xf9 (1111 1001). I2C Master will launch START, Slave address, Read bit, load ACK to I2CMST (0x02) [2], load data to I2CDR, reply ACK and then STOP sequentially.

For I2C read transfer whose data are more than 1 byte, user could split the cycles according to I2C protocol.

7.3.5 I2C and SPI Usage

I2C hardware and SPI hardware modules in the chip share part of the hardware, as a result, when both hardware interfaces are used, the restrictions listed within this section need to be taken into consideration.

I2C and SPI hardware cannot be used as Slave at the same time.

The other cases are supported, including:

- I2C Slave and SPI Master can be used at the same time.
- I2C Master and SPI Slave can be used at the same time.
- I2C and SPI can be used as Master at the same time.

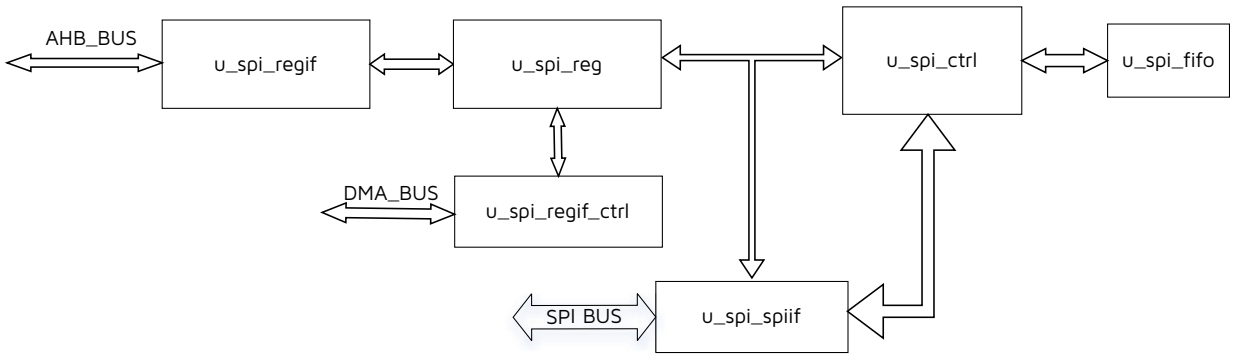
Please refer to corresponding SDK instructions for details.

7.4 SPI

7.4.1 Diagram

The TLSR8208 embeds SPI (Serial Peripheral interface), which could act as Master mode or Slave mode. SPI diagram is shown as following:

Figure 7-12 SPI Diagram



As shown in the diagram, AHB_BUS is used to configure the direct address mapping of registers. DMA_BUS is the bus between the SPI module and the DMA module. SPI_BUS is the SPI interface connected to the pad.

The u_spi_regif is to parse the AHB protocol and send it to the u_spi_reg module for register configuration.

The u_spi_ctrl module selects the state and mode according to the value of the register configuration, and controls the format of the transmitted data.

The u_spi_spiif module adjusts the characteristics of the SPI rate or polarity of transmission and reception according to the configuration.

The u_spi_regif_ctrl module is mainly used to control and analyze signals related to DMA.

The u_spi_fifo serves as a buffer for sending and receiving data.

7.4.2 Features

The features of SPI are listed as following:

- Supports SPI Master/Slave mode
- Supports Dual line, Quad line and 3 line I/O SPI interface
- Supports LCD driving with SPI ports
- Supports DMA transmission

7.4.3 Function Description

7.4.3.1 Master Mode

Users can define transmit data format by configure TransMode registers, the transmitted data will be written in to SPI FIFO via software or DMA.

Master transfer format is shown as following:

Figure 7-13 Master Transfer Mode Format

8bit cmd(default disable)	1~4bytes Addr(default disable)	Transfer mode

Set the cmd_en bit of the SPI_MODE2 register to 1 to indicate that cmd phase is enabled.

See SPI_TRANS0 register for transfer mode configuration.

The SPI output clock of Master mode can be divided by register, which can be up to ahb_clock.

7.4.3.2 Slave Mode

The format that Slave receives is fixed, so the Master needs to send in the prescribed format.

Slave transfer format is shown as following:

Figure 7-14 Slave Transfer Mode Format

8bit slave command (default disable)	1~4bytes dummy	Slave data

Slave judges the read and write operations of the Master according to the received command.

Slave commands are listed in the table below:

Table 7-1 Slave Commands

Slave Command	OP Code	Slave Data
Read status single io	0x05	8bit state(slave ready:0x5a or not ready: 0x00)
Read status dual io	0x15	8bit state(slave ready:0x5a or not ready:0x00)
Read status quad io	0x25	8bit state(slave ready:0x5a or not ready:0x00)
Read data single io	0x0b	Reply data from txfifo
Read data dual io	0x0c	Reply data from txfifo
Read data quad io	0x0e	Reply data from txfifo
Write data single io	0x51	Data saved to rxfifo
Write data dual io	0x52	Data saved to rxfifo
Write data quad io	0x54	Data saved to rxfifo
User-defined	Any 8bit numbers other than the listed OP codes	Depending on the transfer control Register

The SPI input clock should be in the following range for Slave mode: master spi_clk frequency <= 1/4 slave ahb_clk frequency

7.4.3.3 Dual, Quad and 3line I/O

Master's Dual and Quad I/O are configured via the following registers:

spi_dual, spi_quad, cmd_fmt, Addr_fmt.

Spi_dual and spi_quad are used for the Data section. Writing 1 to cmd_fmt means that the I/O mode of the command is the same as that of the Data section, and writing 1 to Addr_fmt means that the I/O mode of the Addr is the same as the Data section.

Master's 3line I/O mode indicates that mosi is a bidirectional I/O. Configured by spi_lsb of register SPIMODE0. SPI_CSN, SPI_CLK, SPI_MOSI form a group of SPI interfaces.

Slave's Dual and Quad I/O are determined based on the command analyzed by Slave. But Slave's command and dummy are fixed only according to single I/O.

Slave also supports 3line mode, and the slave command is only available in single IO mode. The spi_lsb of the SPIMODE0 configuration register is also required. SPI_CSN, SPI_CLK, and SPI_MOSI form a group of SPI interfaces.

7.4.3.4 LCD Display Driving

The SPI can drive LCD display with SPI interface.

There are 3 ways to drive LCD display with SPI: 3-line, 4-line, 2-line.

These 3 methods and all RGB data formats (565, 666, 888) are all supported.

The 4-line format requires an additional GPIO to represent the D/CX signal.

Read/Write Sequences

Read/write sequences of 3-line, 4-line and 2-line serial interfaces are shown in figures below. Please be noted, 2-Line serial interface is designed for writing data in LCD screen, so only writing sequence is shown.

Figure 7-15 3-Line Write Sequence

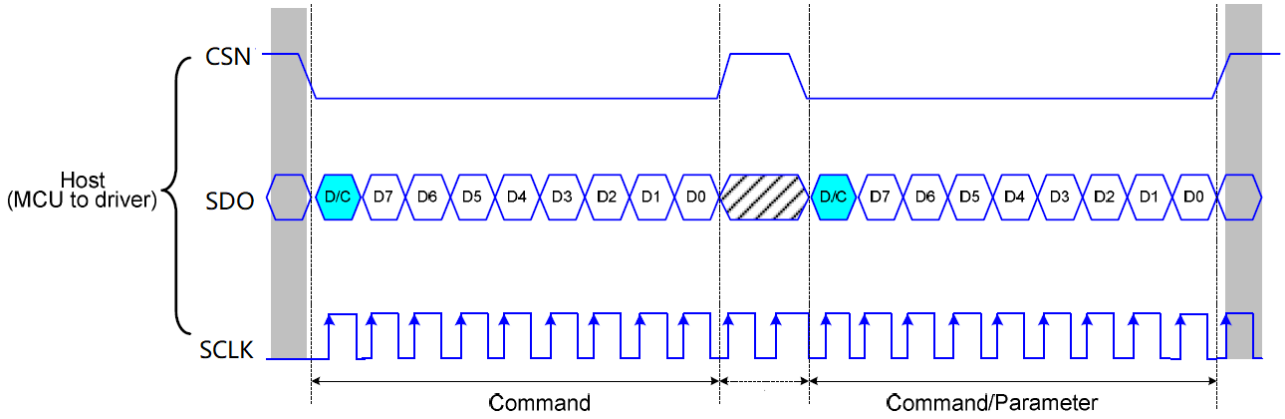


Figure 7-16 3-Line Read Sequence

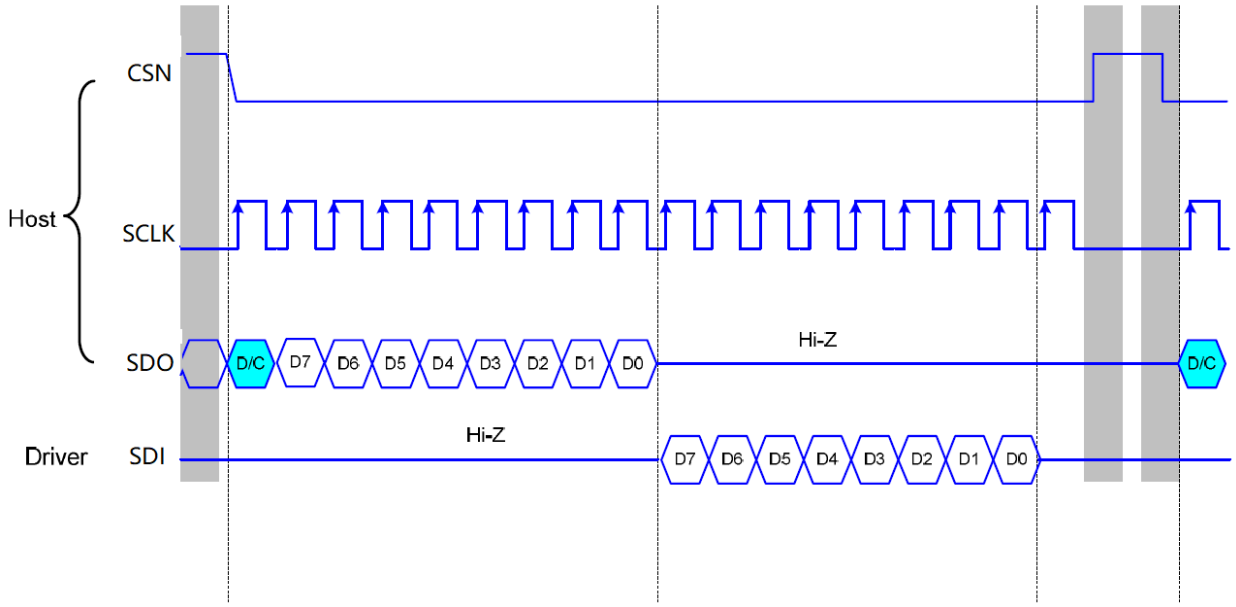


Figure 7-17 4-Line Write Sequence

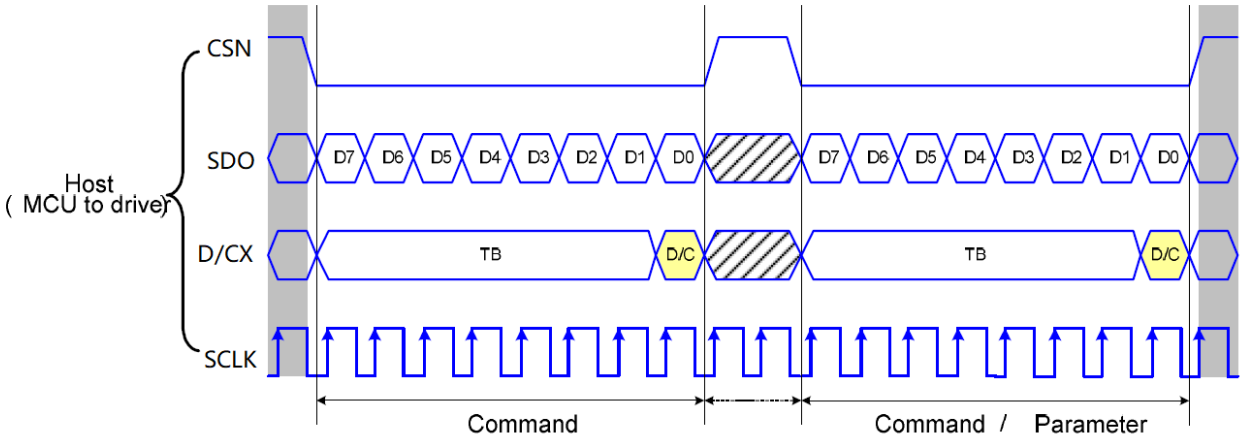


Figure 7-18 4-Line Read Sequence

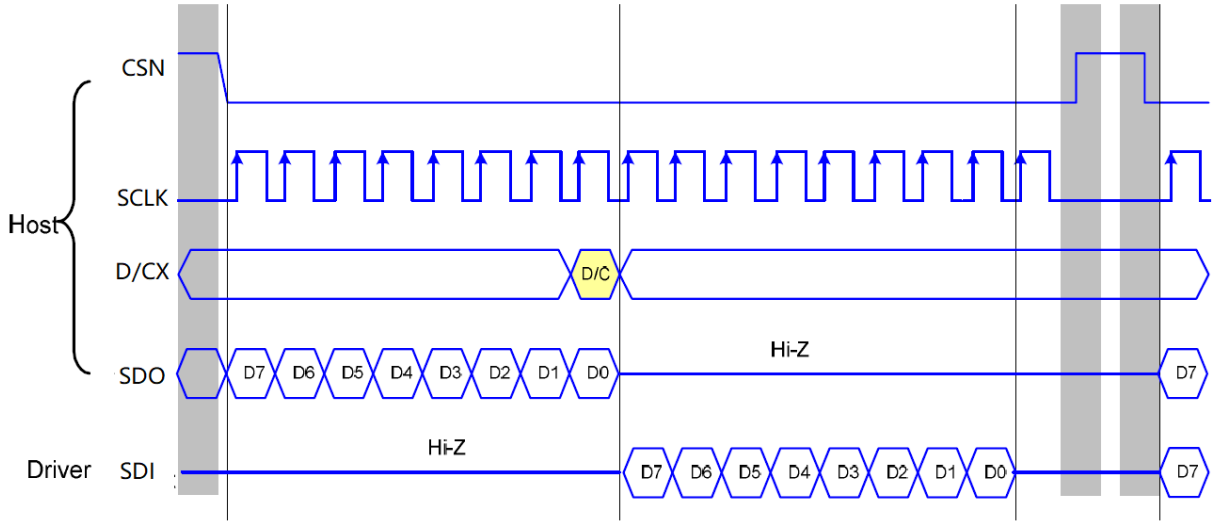
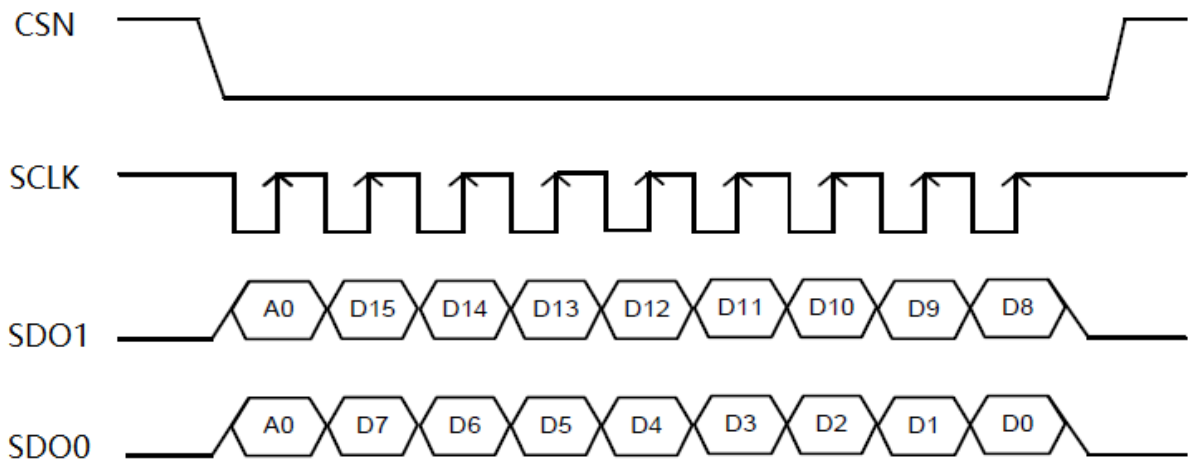


Figure 7-19 2-Line Write Sequence



RGB Formats

RGB565, RGB666 and RGB888 pixel transitions are shown in figures below.

Figure 7-20 RGB565 Pixel Transition

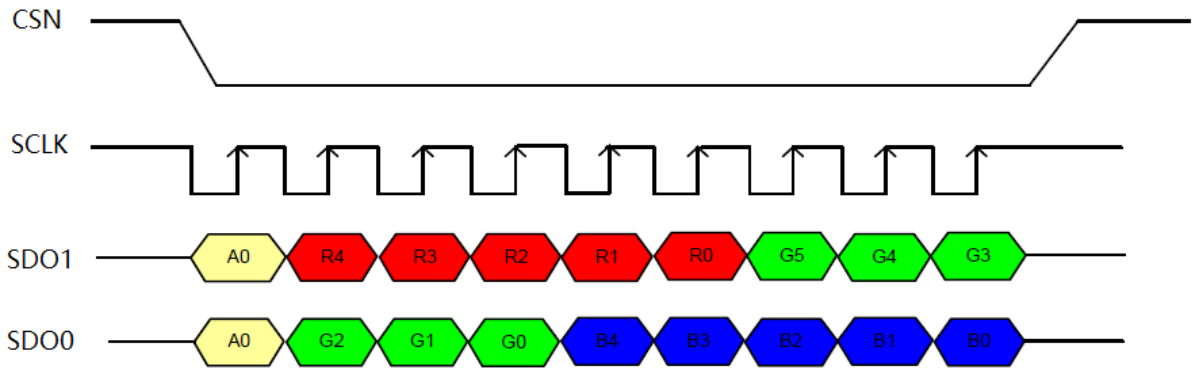


Figure 7-21 RGB666 Pixel Transition

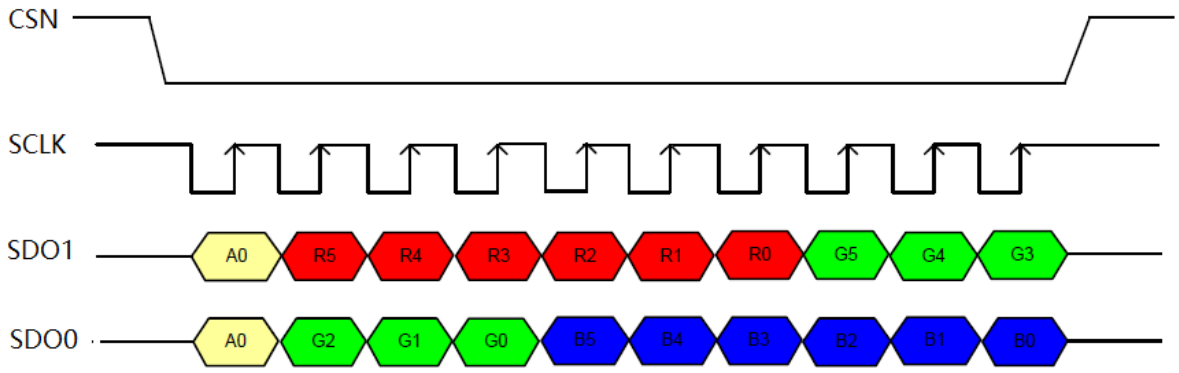
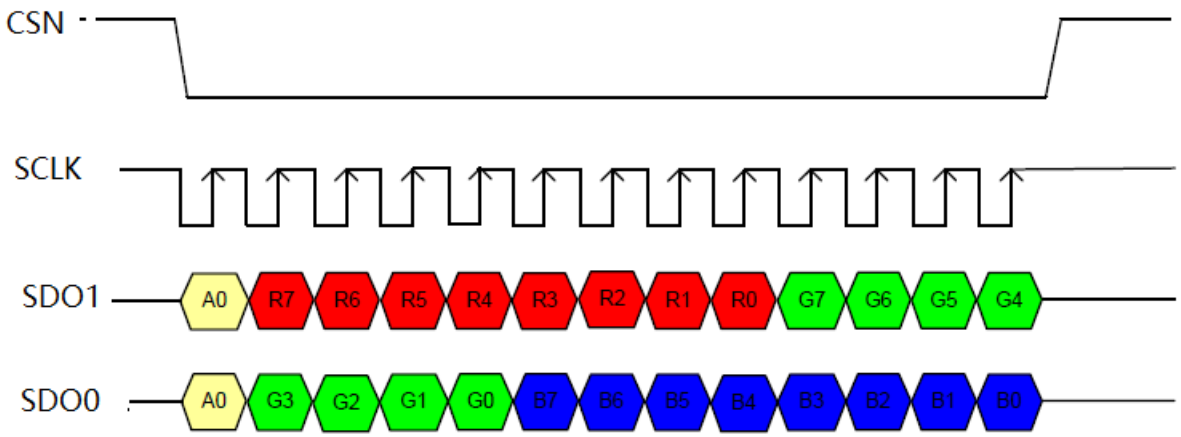


Figure 7-22 RGB888 Pixel Transition



7.4.4 Register Table

Table 7-6 Register Configuration for SPI

Address	R/W	Description	Default Value
0x20	RW	SPI_MODE0 [1:0]: cs2sclk The minimum time between the edge of SPI_CS and the edges of SPI_CLK. The actual duration is (SPI_CLK/2)*(cs2sclk+1) [2]: spi_3line MOSI is bi-directional signal in regular mode [3]: spi_lsb Transfer data with least significant bit first [4]: spi_dual SPI dual data mode [5]: CPHA SPI_CLK Phase [6]: CPOL SPI_CLK Polarity [7]: spi_master SPI master mode	0x80
0x21	RW	SPI_CLK_DIV [7:0]: spi_clk_div The clock freq ratio between the source clock and SPI_CLK. SPI_CLK period = ((spi_clk_div+1)*2)*(period of the source clock)	0x01

Address	R/W	Description	Default Value
0x22	RW	SPI_MODE2 [0]: cmd_fmt 0: single mode 1: the format of the cmd phase is the same as the data phase(Dual/Quad) [1]: spi_quad SPI quad data mode [2]: cmd_en The spi command phase enable [3]: slv_end_int_opt 1: slv_spi_end interrupt using rxfifo_entires < rxfifo_threshold 0: slv_spi_end when csn to high [7:4]: csht The minimum time that spi cs should stay high. The actual duration is (SPI_CLK period / 2)*(csht+1)	0x20
0x23	RW	SPI_TRANS0 [3:0]: dummy_cnt Dummy is always single wire mode. Dummy number = dummy_cnt + 1 [7:4]: transmode The transfer sequence could be: 0x0: write and read at the same time 0x1: write only 0x2: read only 0x3: write, read 0x4: read, write 0x5: write, dummy, read 0x6: read, dummy, write 0x7: none data 0x8: dummy, write 0x9: dummy, read 0xa~0xf: reserved	0x00

Address	R/W	Description	Default Value
0x24	RW	SPI_TRANS1 [7:0]: cmd SPI command	0x00
0x25	RW	SPI_TRANS2 [0]: Enable the spi receive FIFO overrun interrupt [1]: Enable the spi transmit FIFO underrun interrupt [2]: Enable the spi receive FIFO threshold interrupt [3]: Enable the spi transmit FIFO threshold interrupt [4]: Enable the end of spi transfer interrupt [5]: Enable the slave command interrupt [6]: RX DMA enable [7]: TX DMA enable	0x00
0x26	R	SPI_RXFIFO_NUM [3:0]: Number of valid entries in the rxfifo [7:4]: Reserved	-
0x27	R	SPI_TXFIFO_NUM [5:0]: Number of valid entries in the txfifo [7:6]: Reserved	-
0x28	VOLATILE	SPI_WR_RD_DATA0 [7:0]: wr_rd_data0 data[7:0] to transmit or received	0x00
0x29	VOLATILE	SPI_WR_RD_DATA1 [7:0]: wr_rd_data1 data[15:8] to transmit or received	0x00
0x2a	VOLATILE	SPI_WR_RD_DATA2 [7:0]: wr_rd_data2 data[23:16] to transmit or received	0x00
0x2b	VOLATILE	SPI_WR_RD_DATA3 [7:0]: wr_rd_data3 data[31:24] to transmit or received	0x00

Address	R/W	Description	Default Value
0x2c	R	SPI_FIFO_STATE [0]: tx_fifo_sof_clr (RW) [1]: rx_fifo_eof_clr (RW) [2]: rxf_clr (W) rxfifo reset write 1 to reset [3]: txf_clr (W) txfifo reset write 1 to reset [4]: rxf_full rxfifo full flag [5]: rx_empty rxfifo empty flag [6]: txf_full txfifo full flag [7]: txf_empty txfifo empty flag	0xa3
0x2d	W1C	SPI_INTERRUPT_STATUS [1:0]: Reserved [1]: rx_fifo_eof_clr (RW) [2]: rxfifo overrun interrupt [3]: txfifo unerrun interrupt [4]: rxfifo threshold interrupt [5]: txfifo threshold interrupt [6]: End of SPI Transfer interrupt [7]: slave command interruupt	-
0x2e	RW	SPI_STATUS [4:0]: Fifo threshold [5]: Set this bit to indicate the spi as slave is ready for data transaction [6]: SPI soft reset [7]: SPI is transferring	0x04

Address	R/W	Description	Default Value
0x2f	RW	ADDR_CTRL [0]: enable addr phase. 0: disable addr phase 1: enable addr phase [1]: addr_fmt 0: single mode 1: the format of the addr phase is the same as the data phase(Dual/Quad) [3:2]: addr_len 2'b00: 1byte 2'b01: 2bytes 2'b10: 3bytes 2'b11: 4bytes [7:4]: Reserved	0x08
0x30	RW	SPI_ADDR0 [7:0]: addr[7:0]	0x00
0x31	RW	SPI_ADDR1 [7:0]: addr[15:8]	0x00
0x32	RW	SPI_ADDR2 [7:0]: addr[23:16]	0x00
0x33	RW	SPI_ADDR3 [7:0]: addr[31:24]	0x00
0x34	RW	hspi_rx_cnt0 [7:0]: rx_cnt0 Transfer count for read data	0x00
0x35	RW	hspi_rx_cnt1 [7:0]: rx_cnt1 Transfer count for read data	0x00
0x36	RW	hspi_rx_cnt2 [7:0]: rx_cnt2 Transfer count for read data	0x00

Address	R/W	Description	Default Value
0x37	RW	PANEL_REG [0]: line3_dcx_en 1: enable 3 line mode [1]: dcx 1: set dcx field to 1 [4:2]: data_2lane_sel 001: RGB 565 011: RGB 666 111: RGB 888 [5]: endian_mode 0:RGB byte is from low address to high address 1:RGB byte is from high address to low address [6]: fetch_flash Fetch from flash directly	0x00
0x38	RW	hspi_tx_cnt0 [7:0]: tx_cnt0	0x00
0x39	RW	hspi_tx_cnt1 [7:0]: tx_cnt1	0x00
0x3a	RW	hspi_tx_cnt2 [7:0]: tx_cnt2	0x00
0x3b	RW	TX_BURST [1:0]: tx_burst 0: 1 word burst 1: 2 word burst 2: 4 word burst 3: reserved [6:2]: dummy_num_slv [7]: Reserved	0x1c
0x3c	RW	RXFIFO_THRESHOLD [2:0]: rxfifo_threshold	0x4
0x3d	RW	fetch_flash_addr0 [7:0]: fetch_flash_addr0	0x00

Address	R/W	Description	Default Value
0x3e	RW	fetch_flash_addr1 [7:0]: fetch_flash_addr1	0x00
0x3f	RW	fetch_flash_addr2 [7:0]: fetch_flash_addr2	0x00

7.5 UART

7.5.1 Introduction

The TLSR8208 embeds UART (Universal Asynchronous Receiver/Transmitter) to implement full-duplex transmission and reception via UART TX and RX interface. Both TX and RX interface are 4-layer FIFO (First In First Out) interface.

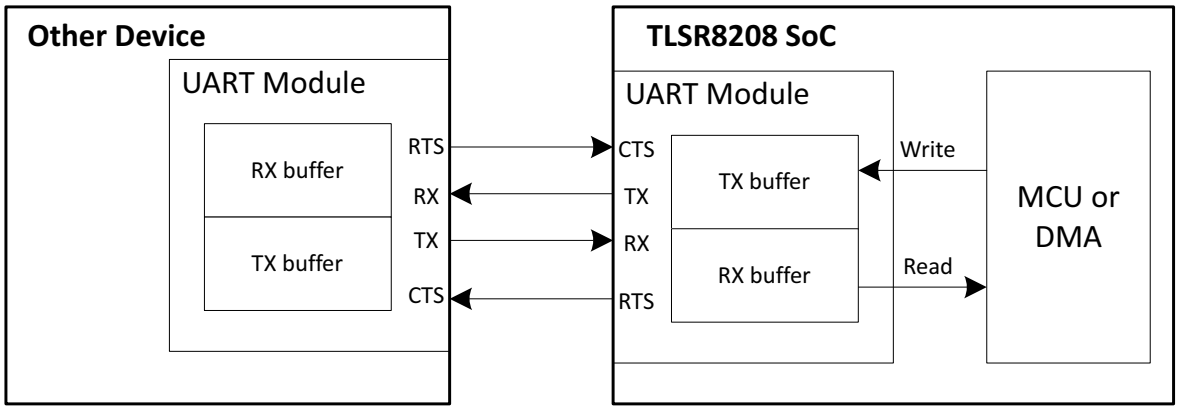
The UART module also supports ISO7816 protocol to enable communication with ISO/IEC 7816 integrated circuit card, especially smart card. In this mode, half-duplex communication (transmission or reception) is supported via the shared 7816_TRX interface.

7.5.2 Function Description

7.5.2.1 Hardware Flow Control

Hardware flow control is supported via RTS and CTS.

Figure 7-23 UART Communication



As shown in the figure above, data to be sent is first written into TX buffer by MCU or DMA, then UART module transmits the data from TX buffer to other device via pin TX. Data to be read from other device is first received via pin RX and sent to RX buffer, then the data is read by MCU or DMA.

The TX FIFO/RX FIFO depth is 8 bytes, and they are controlled by read and write pointers.

For TX FIFO, the write pointer increments by 1 (0x9d[6:4]) for every byte of data written.

For RX FIFO, the read pointer increments by 1 (0x9d[2:0]) for every byte of data read.

The amount of bytes in TX FIFO/RX FIFO can be read from address 0x9c. If the amount of bytes reaches 8, it means the FIFO is full. In this case, if TX FIFO continues to write data or RX FIFO continues to receive data, it will result in data overwriting.

If RX buffer of the TLSR8208 UART is close to full, the TLSR8208 will send a signal (configurable high or low level) via pin RTS to inform other device that it should stop sending data. Similarly, if the TLSR8208 receives a signal from pin CTS, it indicates that RX buffer of other device is close to full and the TLSR8208 should stop sending data.

7.5.2.2 Receiver

When RX, the usage instructions of NDMA (No DMA) and DMA are as follows.

1. NDMA

Since there is no rxdone interrupt under NDMA:

if the length of the received data is random, RX level should be set to 1;

if the length of the received data is known, RX level should be set to less than 8 (The value is recommended to be below the flow control threshold-0x98[3:0]) and an integer multiple of the received length;

rx_irq interrupt processing: The amount of data in the RX FIFO is obtained through register rx_buf_cnt (0x9c[3:0]) and read all data RX FIFO by MCU or DMA;

The depth size of the UART FIFO is 8. If the time before and after entering the rx_irq interrupt exceeds the time of receiving 8 bytes, the FIFO pointer may be disturbed, resulting in abnormal received data. User can determine whether register rx_buf_cnt is greater than 8 as an exception, If this exception occurs, it is recommended to use DMA mode to receive.

2. DMA

Advantage: Automatically received by DMA hardware, does not require MCU polling receive.

Shortcoming: The maximum receive length of DMA is 4075 bytes, if this length is reached, excess data will overwrite the previously received data.

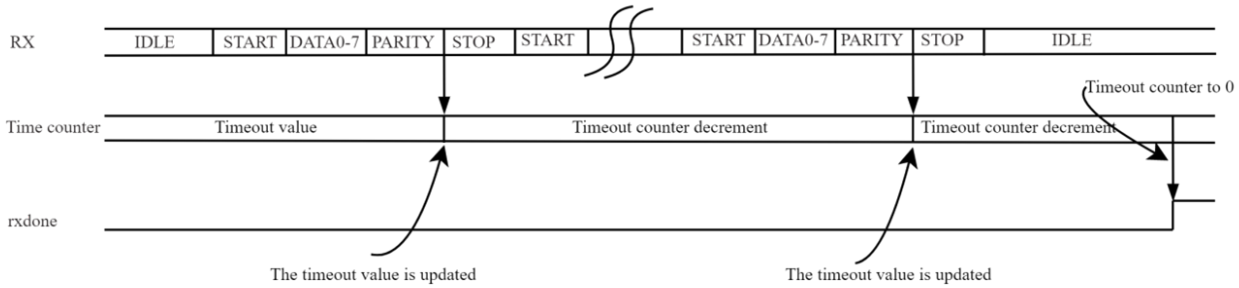
7.5.2.3 Receiver Timeout

Receiver timeout is used to handle when the data received per frame does not reach the threshold. Because data read from the Receiver Buffer Register is a multiple of 4 at a time, The rxdone interrupt is required to process the remaining data below the threshold.

NOTE:

- The DMA Operation threshold is fixed at 4.
- The NDMA Operation threshold can be configured through the register rx_irq_triq_lev (0x99[3:0]).

Figure 7-24 Timeout Flag Used for Data Transmission



The Time out counter inside the UART is updated at the STOP bit, and when receiving data stops, the Timeout counter decreases to 0 and generates a rxdone interrupt.

Note: If the register rxtimeout_rts_en (0x9b[3]) is configured to 1 and the RTS is triggered at the same time, causing the timeout counter to pause.

The configurable total timeout is determined via registers uart_rxtimeout_o_l and uart_rxtimeout_o_h[1:0].

Total timeout = uart_rxtimeout_o_l * (uart_rxtimeout_o_h + 1)

The uart_rxtimeout_o_l register:

The setting is transfer one bytes need cycles base on uart_clk. For example, if transfer one bytes (1 start bit+8 bits data+1 priority bit+2 stop bits) total 12 bits, this register setting should be (register uart_ctrl0[3:0]+1)*12.

The uart_rxtimeout_o_h[1:0] register:

2'b00: rx timeout time is r_rxtimeout[7:0]

2'b01: rx timeout time is r_rxtimeout[7:0]*2

2'b10: rx timeout time is r_rxtimeout[7:0]*3

3'b11: rx timeout time is r_rxtimeout[7:0]*4

The register r_rxtimeout (uart_rxtimeout_o_l and uart_rxtimeout_o_h) is for rx dma to decide the end of each transaction. Supposed the interval between each byte in one transaction is very short.

The minimum time supported via function timeout the time required for a single transmission of 1 byte data, The maximum time is the maximum value supported via register r_rxtimeout. But registers uart_rxtimeout_o_l and uart_rxtimeout_o_h[1:0] still expect to follow our recommended approach.

7.5.3 Register Description

UART related registers are listed in tables below.

Figure 7-25 Register Configuration for UART

Address	R/W	Description	Default Value
0x90	RW	UART_DATA_BUF0 Bit7-0 of Transmitter/Receiver Buffer Register (TX/RX FIFO)	0x00
0x91	RW	UART_DATA_BUF1 Bit15-8 of Transmitter/Receiver Buffer Register (TX/RX FIFO)	0x00

Address	R/W	Description	Default Value
0x92	RW	UART_DATA_BUF2 Bit23-16 of Transmitter/Receiver Buffer Register (TX/RX FIFO)	0x00
0x93	RW	UART_DATA_BUF3 Bit31-24 of Transmitter/Receiver Buffer Register (TX/RX FIFO)	0x00
0x94	RW	UART_CLK_DIV_L [7:0]: Least significant byte of the uart_clk_div register	0xff
0x95	RW	UART_CLK_DIV_H [6:0]: Most significant byte of the uart_clk_div register uart_sclk = pclk/(uart_clk_div[14:0]+1) [7]:enable clock divider 1:enable 0:disable	0x0f
0x96	RW	UART_CTRL0 [3:0]: bpwc_o bwpc, bit width, should be larger than 2 Baud rate = uart_sclk/(bwpc+1) [4]: rx_dma_en [5]: tx_dma_en [6]: mask_rx_irq rx interrupt enable [7]: mask_tx_irq tx interrupt enable	0x0f

Address	R/W	Description	Default Value
0x97	RW	UART_CTRL1 [0]: tx_cts_polarity Polarity of CTS 0: Active low (0 - End of transmission) 1: Active high (1 - End of transmission) [1]: tx_cts_enable cts enable, 1: enable, 0: disable [2]: parity_enable When this bit is set, a parity bit is generated in transmitted data before the first STOP bit and the parity bit would be checked for the received data. [3]: parity_polarity Even parity select, 1: old parity; 0: even parity (an even number of logic-1 is in the data and parity bits). [5:4]: stop_sel stop bit 00: 1 bit, 01: 1.5 bits, 1x: 2 bits [6]: ttl_enable TX and RX polarity selection, 0: Non-inverting; 1: Inverting [7]: loopback_o Enable loopback mode, 1: enable; 0: disable	0x0e
0x98	RW	UART_CTRL2 [3:0]: rts_triq_lev RTS trig level. Trigger RTS when the RX FIFO reaches the threshold. [4]: rts_polarity Polarity of RTS 0: Active high (0-ready for receiving) 1: Active low (1-ready for receiving) [5]: rts_manul_v rts manual value [6]: rts_manul_m rts manual enable [7]: rts_en RTS enable, 1: enable; 0: disable	0xa5

Address	R/W	Description	Default Value
0x99	RW	UART_CTRL3 [3:0]: rx_irq_triq_lev rx_irq_trig level. Trigger rx_buf_irq interrupt when the RX FIFO reaches the threshold. [7:4]: tx_irq_trig_lev tx_irq_trig level. Trigger tx_buf_irq interrupt when the TX FIFO under the threshold.	0x44
0x9a	RW	UART_RXTIMEOUT_O_L [7:0]: r_rxtimeout_o[7:0] Least significant byte of the r_rxtimeout_o register: The setting is transfer one bytes need cycles base on uart_clk. For example, if transfer one bytes (1 start bit+8bits data+1 priority bit+2 stop bits) total 12 bits, this register setting should be (bwpc+1)*12.	0xc0
0x9b	RW	UART_RXTIMEOUT_O_H [1:0]: r_rxtimeout_o[9:8] Most significant byte of the r_rxtimeout register: 2'b00:rx timeout time is r_rxtimeout[7:0] 2'b01:rx timeout time is r_rxtimeout[7:0]*2 2'b10:rx timeout time is r_rxtimeout[7:0]*3 3'b11: rx timeout time is r_rxtimeout[7:0]*4 r_rxtimeout is for rx dma to decide the end of each transaction. Supposed the interval between each byte in one transaction is very short. [2]: rxdone_rts_en 1'b1:rxdone work on rts; 1'b0:rxdone doesn't work on rts [3]: rxtimeout_rts_en RTS controls timeout stop enabling signal [4]: mask_rxdone_irq Enable rxdone_irq interrupt [5]: p7816_en_o 7816 enable [6]: mask_txdone_irq Enable txdone interrupt [7]: mask_err_irq Enable rx_err interrupt	0x0d

Address	R/W	Description	Default Value
0x9c	R	UART_BUFCNT [3:0]: rx_bufcnt This register is increased when there are incoming received data in the Receiver Buffer Register. When there is read data in the Receiver Buffer Register, this register is decremented. [7:4]: tx_bufcnt This register is decremented when there are outgoing sent data in the Transmitter Buffer Register. When there is write data in the Transmitter Buffer Register, this register is increased.	0x00
0x9d	VOLATILE	UART_STATUS [2:0]: rbcnt When there is read data in the Receiver Buffer Register, this register is decremented. [3]: irq_o Total interruption of UART. [6:4]: wbcnt (R) W:[4] write 1 to clear rxdone_irq;W: [6] write 1 to clear rx [7]: rx_err R: rx_err, W: write 1 to clear tx	0x00

Address	R/W	Description	Default Value
0x9e	VOLATILE	UART_TXRX_STATUS [0]: txdone When the transmitter ends, the UART controller will assert txdone interrupt. W: write 1 to clear txdone [1]: tx_buf_irq When the TX FIFO under the threshold set by the tx_irq_trig register, the UART controller will assert tx_buf_irq interrupt. W: write 1 to clear TX FIFO pointer, and so on. Note: When TX FIFO is greater than the threshold set by the tx_irq_trig register, the tx_buf_irq interrupt clears automatically. [2]: rxdone (R) [3]: rx_buf_irq When the RX FIFO reaches the threshold set by the rx_irq_trig Register, the UART controller will assert rx_buf_irq interrupt. W: write 1 to clear RX FIFO pointer, rx err, and so on. Note: When RX FIFO is below the threshold set by the rx_irq_trig Register, the rx_buf_irq interrupt clears automatically. [4]: rxdone_irq When the receiver ends (the timeout counter decays to 0), the UART controller will assert rxdone_irq interrupt. W: write 1 to clear rxdone_irq [5]: hold1 (RW) [6]: auto_rxclr_en (RW) DMA and NDMA mode: auto clr function switch; 1:enable,0:disable [7]: ndma_rxdone_en (RW) NDMA mode: rxdone(timeout) function switch; 1:enable,0:disable;dma mode must disable	0xc0
0x9f	VOLATILE	UART_STATE [2:0]: tstate_i [3]: timeout (R) [7:4]: rstate_i rx state machine,W:[6] write 1 to clear sclk_cnt; W:[7] write 1 to clear txdone	0x00

Addresses 0x90 ~ 0x93 serve to write data into TX buffer or read data from RX buffer.

Addresses 0x94 ~ 0x95 serve to configure UART clock.

Address 0x96 serves to set baud rate (bit[3:0]), enable RX/TX DMA mode (bit[4:5]), and enable RX/TX interrupt (bit[6:7]).

Address 0x97 mainly serves to configure CTS. Bit[1] should be set to 1'b1 to enable CTS. Bit[0] serves to configure CTS signal level. Bit[2:3] serve to enable parity bit and select even/odd parity. Bit[5:4] serve to select 1/1.5/2 bits for stop bit. Bit[6] serves to configure whether RX/TX level should be inverted.

Address 0x98 serves to configure RTS. Bit[7] and Bit[3:0] serve to enable RTS and configure RTS signal level.

Address 0x99 serves to configure the number of bytes in RX/TX buffer to trigger interrupt.

The number of bytes in RX/TX buffer can be read from address 0x9c.

7.6 USB

The TLSR8208 has a full-speed (12 Mbps) USB interface for communicating with other compatible digital devices. The USB interface acts as a USB peripheral, responding to requests from a master host controller. The chip contains internal 1.5 kOhm pull up resistor for the DP pin, which can be enabled via analog register afe_0x0b<7>.

Telink USB interface supports the Universal Serial Bus Specification, Revision v2.0 (USB v2.0 Specification).

The chip supports 9 endpoints, including control endpoint 0 and 8 configurable data endpoints. Endpoint 1, 2, 3, 4, 7 and 8 can be configured as input endpoint, while endpoint 5 and 6 can be configured as output endpoint. In audio class application, only endpoint 6 supports iso out mode, while endpoint 7 supports iso in mode. In other applications, each endpoint can be configured as bulk, interrupt and iso mode. For control endpoint 0, the chip's hardware vendor command is configurable.

Optional suspend mode:

- Selectable as USB suspend mode or chip suspend mode, support remote wakeup.
- Current draw in suspend mode complies with USB v2.0 Specification.
- USB pins (DM, DP) can be used as GPIO function in suspend mode.
- Resume and detach detect: Recognize USB device by detecting the voltage on the DP pin with configurable 1.5k pull-up resistor.
- USB pins configurable as wakeup GPIOs.

The USB interface belongs to an independent power domain, and it can be configured to power down independently.

8 PWM

The TLSR8208 supports up to 6-channel PWM (Pulse-Width-Modulation) output. Each PWM#n (n = 0 ~ 5) has its corresponding inverted output at PWM#n_N pin.

8.1 Register Table

Table 8-1 Register Table for PWM

Address	R/W	Description	Default Value
0x780	W	[1]: 0 - disable PWM1, 1 - enable PWM1 [2]: 0 - disable PWM2, 1 - enable PWM2[3]: 0 - disable PWM3, 1 - enable PWM3 [4]: 0 - disable PWM4, 1 - enable PWM4 [5]: 0 - disable PWM5, 1 - enable PWM5	0x00
0x781	W	[0]: 0 - disable PWM0, 1 - enable PWM0	0x00
0x782	RW	Set PWM_clk: (PWM_CLKDIV+1)*sys_clk	0x00
0x783	RW	[3:0]: PWM0 mode select 0000 - PWM0 normal mode 0001 - PWM0 count mode 0011 - PWM0 IR mode 0111 - PWM0 IR FIFO mode 1111 - PWM0 IR DMA FIFO mode	0x00
0x784	RW	[5:0]: 1'b1 invert PWM output	0x00
0x785	RW	[5:0]: 1'b1 invert PWM_INV output	0x00
0x786	RW	[5:0]: Signal frame polarity of PWM5 ~ PWM0 1'b0 - high level first 1'b1 - low level first	0x00
0x788 ~ 0x793	-	Reserved	-
0x794	RW	[7:0] bits 7-0 of PWM0's high time or low time (if pola[0] = 1)	0x00
0x795	RW	[15:8] bits 15-8 of PWM0's high time or low time	0x00
0x796	RW	[7:0] bits 7-0 of PWM0's cycle time	0x00
0x797	RW	[15:8] bits 15-8 of PWM0's cycle time	0x00

Address	R/W	Description	Default Value
0x798	RW	[7:0] bits 7-0 of PWM1's high time or low time (if pola[1] = 1)	0x00
0x799	RW	[15:8] bits 15-8 of PWM1's high time or low time	0x00
0x79a	RW	[7:0] bits 7-0 of PWM1's cycle time	0x00
0x79b	RW	[15:8] bits 15-8 of PWM1's cycle time	0x00
0x79c	RW	[7:0] bits 7-0 of PWM2's high time or low time (if pola[2] = 1)	0x00
0x79d	RW	[15:8] bits 15-8 of PWM2's high time or low time	0x00
0x79e	RW	[7:0] bits 7-0 of PWM2's cycle time	0x00
0x79f	RW	[15:8] bits 15-8 of PWM2's cycle time	0x00
0x7a0	RW	[7:0] bits 7-0 of PWM3's high time or low time	0x00
0x7a1	RW	[15:8] bits 15-8 of PWM3's high time or low time	0x00
0x7a2	RW	[7:0] bits 7-0 of PWM3's cycle time	0x00
0x7a3	RW	[15:8] bits 15-8 of PWM3's cycle time	0x00
0x7a4	RW	[7:0] bits 7-0 of PWM4's high time or low time (if pola[4] = 1)	0x00
0x7a5	RW	[15:8] bits 15-8 of PWM4's high time or low time	0x00
0x7a6	RW	[7:0] bits 7-0 of PWM4's cycle time	0x00
0x7a7	RW	[15:8] bits 15-8 of PWM4's cycle time	0x00
0x7a8	RW	[7:0] bits 7-0 of PWM5's high time or low time (if pola[5] = 1)	0x00
0x7a9	RW	[15:8] bits 15-8 of PWM5's high time or low time	0x00
0x7aa	RW	[7:0] bits 7-0 of PWM5's cycle time	0x00
0x7ab	RW	[15:8] bits 15-8 of PWM5's cycle time	0x00
0x7ac	RW	[7:0] bits 7-0 of PWM0 Pulse number in count mode and IR mode	0x00
0x7ad	RW	[13:8] bits 13-8 of PWM0 Pulse number in count mode and IR mode	0x00
0x7ae ~ 0x7af	-	Reserved	-

Address	R/W	Description	Default Value
0x7b0	RW	INT mask [0]: PWM0 Pnum int 0 - disable, 1 - enable [1]: PWM0 ir dma fifo mode int 0 - disable, 1 - enable [2]: PWM0 frame int 0 - disable, 1 - enable [3]: PWM1 frame int 0 - disable, 1 - enable [4]: PWM2 frame int 0 - disable, 1 - enable[5]: PWM3 frame int 0 - disable, 1 - enable [6]: PWM4 frame int 0 - disable, 1 - enable [7]: PWM5 frame int 0 - disable, 1 - enable	0x00
0x7b1	RW	INT status, write 1 to clear [0]: PWM0 pnum int (have sent PNUM pulses, PWM_NCNT==PWM_PNUM) [1]: PWM0 ir dma fifo mode int (pnum int & fifo empty in ir dma fifo mode) [2]: PWM0 cycle done int (PWM_CNT==PWM_TMAX) [3]: PWM1 cycle done int (PWM_CNT==PWM_TMAX) [4]: PWM2 cycle done int (PWM_CNT==PWM_TMAX)[5]: PWM3 cycle done int (PWM_CNT==PWM_TMAX) [6]: PWM4 cycle done int (PWM_CNT==PWM_TMAX) [7]: PWM5 cycle done int (PWM_CNT==PWM_TMAX)	0x00
0x7b2	RW	[0]: PWM0 fifo mode fifo cnt int mask 0 - disable, 1 - enable	0x00
0x7b3	RW	INT status, write 1 to clear [0]: fifo mode cnt int, when FIFO_NUM (0x7cd[3:0]) is less than FIFO_NUM_LVL (0x7cc[3:0])	0x00
0x7b4	R	[7:0] PWM0 cnt value	0x00
0x7b5	R	[15:8] PWM0 cnt value	0x00

Address	R/W	Description	Default Value
0x7b6	R	[7:0] PWM1 cnt value	0x00
0x7b7	R	[15:8] PWM1 cnt value	0x00
0x7b8	R	[7:0] PWM2 cnt value	0x00
0x7b9	R	[15:8] PWM2 cnt value	0x00
0x7ba	R	[7:0] PWM3 cnt value	0x00
0x7bb	-	[15:8] PWM3 cnt value	0x00
0x7bc	R	[7:0] PWM4 cnt value	0x00
0x7bd	-	[15:8] PWM4 cnt value	0x00
0x7be	R	[7:0] PWM5 cnt value	0x00
0x7bf	-	[15:8] PWM5 cnt value	0x00
0x7c0	R	[7:0] PWM0 pluse_cnt value	0x00
0x7c1	R	[15:8] PWM0 pluse_cnt value	0x00
0x7c2 ~ 0x7c3	-	Reserved	-
0x7c4	RW	[7:0] bits 7-0 of PWM0's high time or low time (if pola[0]=1), if shadow bit (fifo data[14]) is 1'b1 in ir fifo mode or dma fifo mode	0x00
0x7c5	RW	[15:8] bits 15-8 of PWM0's high time or low time, if shadow bit (fifo data[14]) is 1'b1 in ir fifo mode or dma fifo mode	0x00
0x7c6	RW	[7:0] bits 7-0 of PWM0's cycle time, if shadow bit (fifo data[14]) is 1'b1 in ir fifo mode or dma fifo mode	0x00
0x7c7	RW	[15:8] bits 15-8 of PWM0's cycle time, if shadow bit (fifo frame[14]) is 1'b1 in ir fifo mode or dma fifo mode	0x00
0x7c8	RW	Use in IR FIFO mode	0x00
0x7c9	RW	Use in IR FIFO mode	0x00
0x7ca	RW	Use in IR FIFO mode	0x00
0x7cb	RW	Use in IR FIFO mode	0x00
0x7cc	RW	FIFO num int trigger level	0x00

Address	R/W	Description	Default Value
0x7cd	R	[3:0]: FIFO DATA NUM (byte) [4]: FIFO EMPTY [5]: FIFO FULL	0x10
0x7ce	W1C	[0]: clear: write 1 to clear data in FIFO; normal (default): write 0	0x00

8.2 Enable PWM

Register PWM_EN (address 0x780)[5:1] and PWM_EN0 (address 0x781)[0] serves to enable PWM5 ~ PWM0 respectively via writing "1" for the corresponding bits.

8.3 Set PWM Clock

PWM clock derives from system clock. Register PWM_CLKDIV (address 0x782) serves to set the frequency dividing factor for PWM clock. Formula below applies:

$$F_{PWM} = F_{System\ clock} / (PWM\_CLKDIV+1)$$

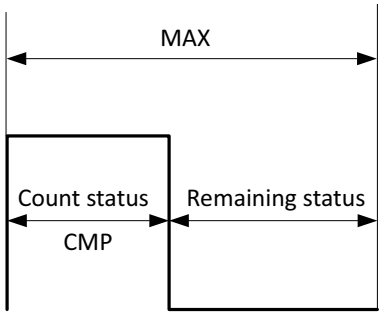
8.4 PWM Waveform, Polarity and Output Inversion

Each PWM channel has independent counter and 2 status including "Count" and "Remaining". Count and Remaining status form a signal frame.

8.4.1 Waveform of Signal Frame

When PWM#n is enabled, first PWM#n enters Count status and outputs High level signal by default. When PWM#n counter reaches cycles set in register PWM_TCMP#n (address 0x794 ~ 0x795, 0x798 ~ 0x799, 0x79c ~ 0x79d, 0x7a0 ~ 0x7a1, 0x7a4 ~ 0x7a5, 0x7a8 ~ 0x7a9) / PWM_TCMP0_SHADOW (0x7c4 ~ 0x7c5), PWM#n enters Remaining status and outputs Low level till PWM#n cycle time configured in register PWM_TMAX#n (address 0x796 ~ 0x797, 0x79a ~ 0x79b, 0x79e ~ 0x79f, 0x7a2 ~ 0x7a3, 0x7a6 ~ 0x7a7, 0x7aa ~ 0x7ab) / PWM_TMAX0_SHADOW (0x7c6 ~ 0x7c7) expires.

Figure 8-1 A Signal Frame



An interruption will be generated at the end of each signal frame if enabled via register PWM_MASK (address 0x7b0[2:7]).

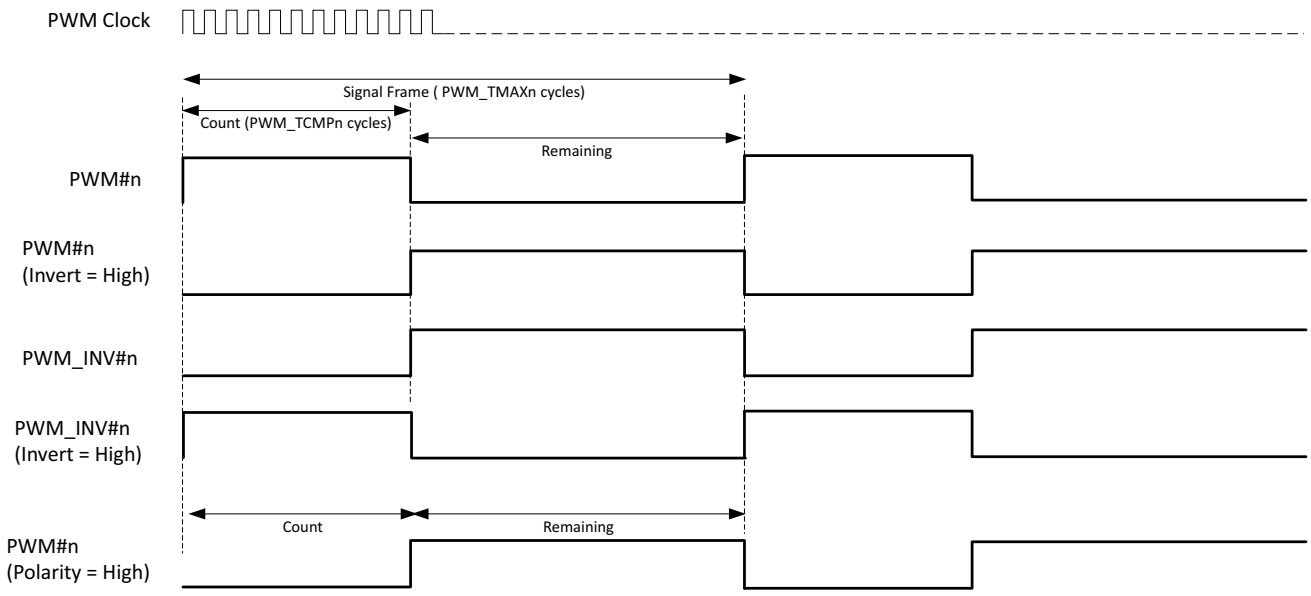
8.4.2 Invert PWM Output

PWM#n and PWM#n_N output could be inverted independently via register PWM_CC0 (address 0x784) and PWM_CC1 (address 0x785). When the inversion bit is enabled, waveform of the corresponding PWM channel will be inverted completely.

8.4.3 Polarity for Signal Frame

By default, PWM#n outputs High level at Count status and Low level at Remaining status. When the corresponding polarity bit is enabled via register PWM_CC2 (address 0x786[5:0]), PWM#n will output Low level at Count status and High level at Remaining status.

Figure 8-2 PWM Output Waveform Chart



8.5 PWM Modes

8.5.1 Select PWM Modes

PWM0 supports five modes, including Continuous mode (normal mode, default), Counting mode, IR mode, IR FIFO mode, IR DMA FIFO mode.

PWM1 ~ PWM5 only support Continuous mode.

Register PWM_MODE (address 0x783) serves to select PWM0 mode.

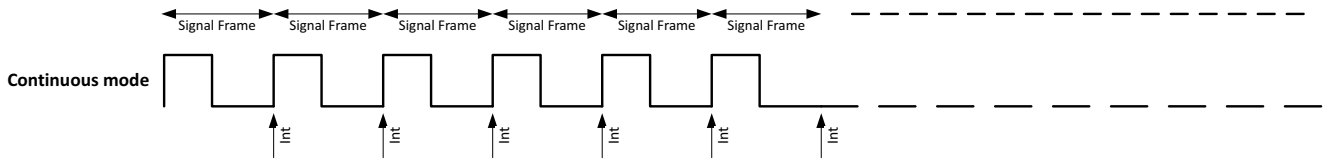
8.5.2 Continuous Mode

PWM0 ~ PWM5 all support Continuous mode. In this mode, PWM#n continuously sends out signal frames. PWM#n should be disabled via address 0x780/0x781 to stop it; when stopped, the PWM output will turn low immediately.

During Continuous mode, waveform could be changed freely via PWM_TCMP#n and PWM_TMAX#n. New configuration for PWM_TCMP#n and PWM_TMAX#n will take effect in the next signal frame.

After each signal frame is finished, corresponding PWM cycle done interrupt flag bit (0x7b1[2:7]) will be automatically set to 1'b1. If the interrupt is enabled by setting PWM_MASK0 (address 0x7b0[2:7]) as 1'b1, a frame interruption will be generated. User needs to write 1'b1 to the flag bit to manually clear it.

Figure 8-3 Continuous Mode



8.5.3 Counting Mode

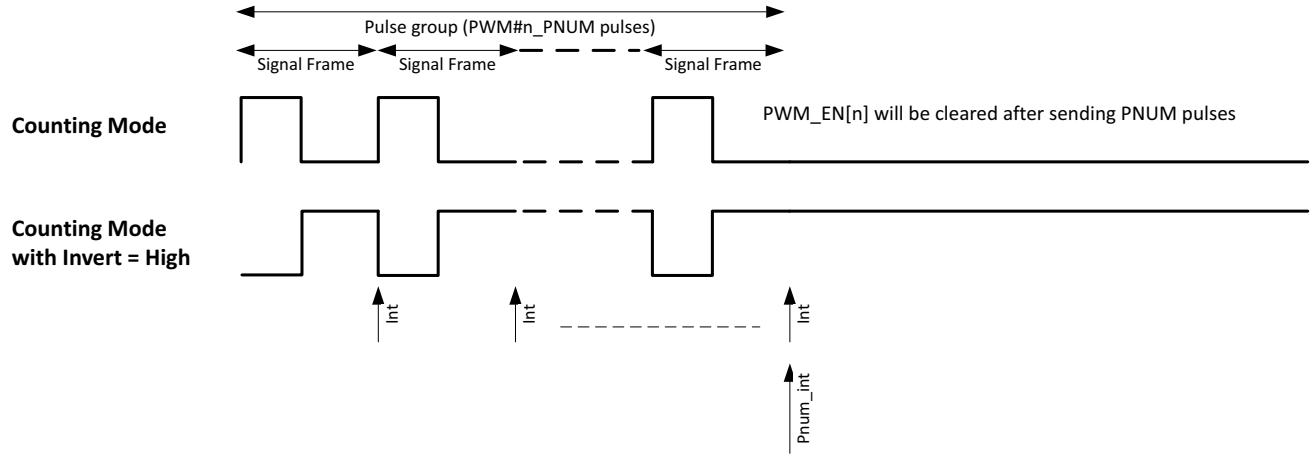
Only PWM0 supports Counting mode. Address 0x783[3:0] should be set as 4'b0001 to select PWM0 counting mode.

In this mode, PWM0 sends out specified number of signal frames which is defined as a pulse group. The number is configured via register PWM_PNUM0 (address 0x7ac ~ 0x7ad).

After each signal frame is finished, PWM0 cycle done interrupt flag bit (0x7b1[2]) will be automatically set to 1'b1. If the interrupt is enabled by setting PWM_MASK0 (address 0x7b0[2]) as 1'b1, a frame interruption will be generated. User needs to write 1'b1 to the flag bit to manually clear it.

After a pulse group is finished, PWM0 will be disabled automatically, and PWM0 Pnum interrupt flag bit (0x7b1[0]) will be automatically set to 1'b1. If the interrupt is enabled by setting PWM_MASK0 (address 0x7b0[0]) as 1'b1, a Pnum interruption will be generated. User needs to write 1'b1 to the flag bit to manually clear it.

Figure 8-4 Counting Mode (n=0)



Counting mode also serves to stop IR mode gracefully. Refer to Section 8.5.4 for details.

8.5.4 IR Mode

Only PWM0 supports IR mode. Address 0x783[3:0] should be set as 4'b0011 to select PWM0 IR mode.

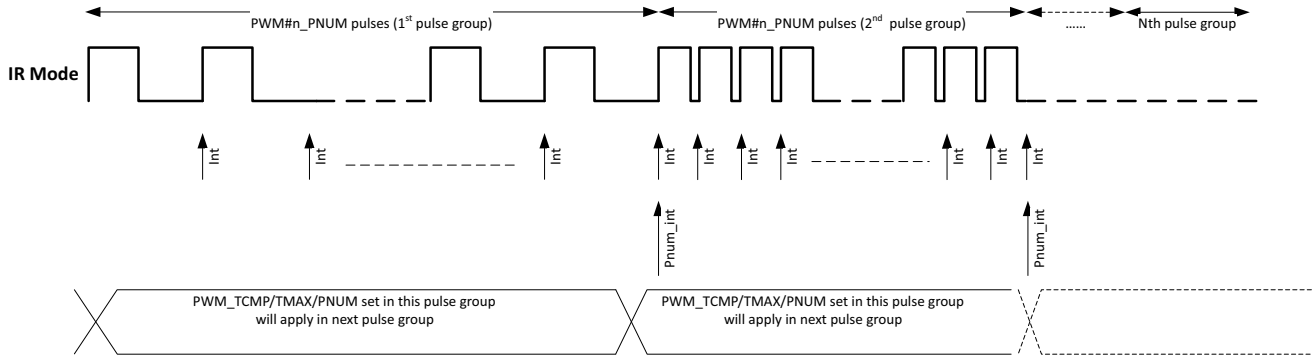
In this mode, specified number of frames is defined as one pulse group. In contrast to Counting mode where PWM0 stops after first pulse group is finished, PWM0 will constantly send pulse groups in IR mode.

During IR mode, PWM0 output waveform could also be changed freely via WM_TCMP0, PWM_TMAX0 and PWM_PNUM0. New configuration for PWM_TCMP0, PWM_TMAX0 and PWM_PNUM0 will take effect in the next pulse group.

To stop IR mode and complete current pulse group, user can switch PWM0 from IR mode to Counting mode so that PWM0 will stop after current pulse group is finished. If PWM0 is disabled directly via PWM_EN0 (0x781[0]), PWM0 output will turn Low immediately despite of current pulse group.

After each signal frame/pulse group is finished, PWM0 cycle done interrupt flag bit (0x7b1[2])/PWM0 Pnum interrupt flag bit (0x7b1[0]) will be automatically set to 1'b1. A frame interruption/Pnum interruption will be generated (if enabled by setting address 0x7b0[2]/0x7b0[0] as 1'b1).

Figure 8-5 IR Mode (n=0)



8.5.5 IR FIFO Mode

IR FIFO mode is designed to allow IR transmission of long code patterns without the continued intervention of MCU, and it is designed as a selectable working mode on PWM0. The IR carrier frequency is divided down from the system clock and can be configured as any normal IR frequencies, e.g. 36 kHz, 38 kHz, 40 kHz, or 56 kHz.

Only PWM0 supports IR FIFO mode. Address 0x783[3:0] should be set as 4'b0111 to select PWM0 IR FIFO mode.

An element ("FIFO CFG Data") is defined as basic unit of IR waveform, and written into FIFO. This element consists of 16 bits, including:

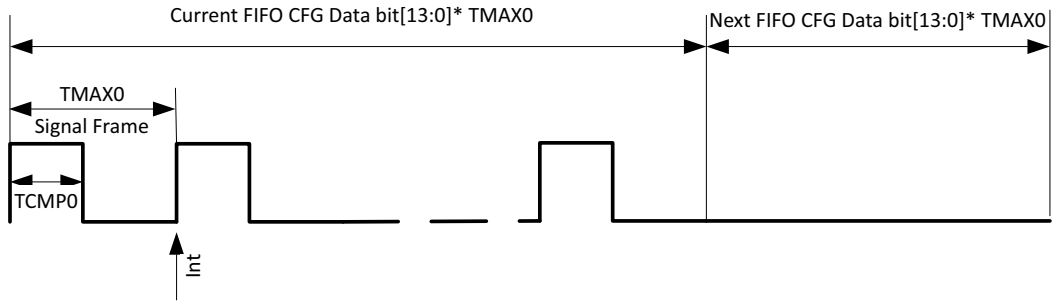
- bit[13:0] defines PWM pulse number of current group.
- bit[14] determines duty cycle and period for current PWM pulse group.
  - 0: use configuration of TCMP0 and TMAX0 in 0x794 ~ 0x797;
  - 1: use configuration of TCMP0_SHADOW and TMAX0_SHADOW in 0x7c4 ~ 0x7c7.
- bit[15] determines whether current PWM pulse group is used as carrier, i.e. whether PWM will output pulse (1) or low level (0).

User should use FIFO_DATA_ENTRY in 0x7c8 ~ 0x7cb to write the 16-bit "FIFO CFG Data" into FIFO by byte or half word or word.

- To write by byte, user should successively write 0x7c8, 0x7c9, 0x7ca and 0x7cb.
- To write by half word, user should successively write 0x7c8 and 0x7ca.
- To write by word, user should write 0x7c8.

FIFO depth is 8 bytes. User can read the register FIFO_SR in 0x7cd to view FIFO empty/full status and check FIFO data number.

Figure 8-6 IR Format Examples

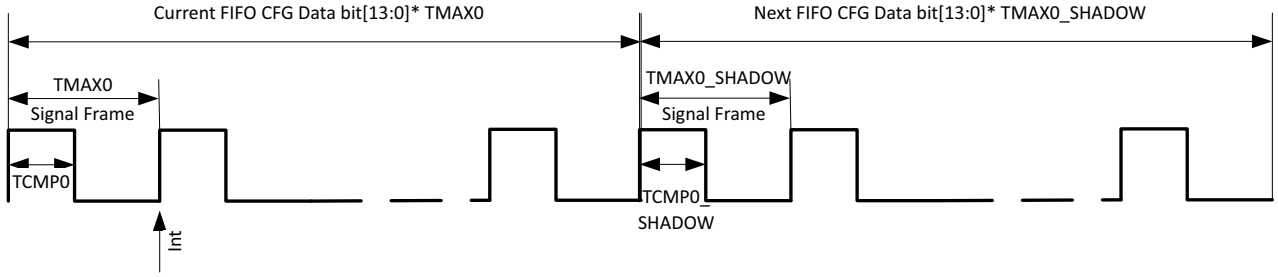


Current FIFO CFG Data bit[15] = 1

Next FIFO CFG Data bit[15] = 0

Current FIFO CFG Data bit[14] = 0

Next FIFO CFG Data bit[14] = 0



Current FIFO CFG Data bit[15] = 1

Next FIFO CFG Data bit[15] = 1

Current FIFO CFG Data bit[14] = 0

Next FIFO CFG Data bit[14] = 1

When "FIFO CFG Data" is configured in FIFO and PWM0 is enabled via PWM_EN0 (address 0x781[0]), the configured waveforms will be output from PWM0 in sequence. As long as FIFO doesn't overflow, user can continue to add waveforms during IR waveforms sending process, and long IR code that exceeds the FIFO depth can be implemented this way. After all waveforms are sent, FIFO becomes empty, PWM0 will be disabled automatically.

The FIFO_CLR register (address 0x7ce[0]) serves to clear data in FIFO. Writing 1'b1 to this register will clear all data in the FIFO. Note that the FIFO can only be cleared when not in active transmission.

8.5.6 IR DMA FIFO Mode

IR DMA FIFO mode is designed to allow IR transmission of long code patterns without occupation of MCU, and it is designed as a selectable working mode on PWM0. The IR carrier frequency is divided down from the system clock and can be configured as any normal IR frequencies, e.g. 36 kHz, 38 kHz, 40 kHz, or 56 kHz.

Only PWM0 supports IR DMA FIFO mode. Address 0x783[3:0] should be set as 4'b1111 to select PWM0 IR DMA FIFO mode.

This mode is similar to IR FIFO mode, except that "FIFO CFG Data" is written into FIFO by DMA instead of MCU. User should write the configuration of "FIFO CFG Data" into RAM, and then enable DMA channel 5. DMA will automatically write the configuration into FIFO.

NOTE: *In this mode, when DMA channel 5 is enabled, PWM will automatically output configured waveform, without the need to manually enable PWM0 via 0x781[0] (i.e. 0x781[0] will be set as 1'b1 automatically).*

Example 1:

Suppose Mark carrier (pulse) frequency1 (F1) = 40 kHz, duty cycle 1/3

Mark carrier (pulse) frequency2 (F2) = 50 kHz, duty cycle 1/2

Space carrier (low level) frequency (F3) = 40 kHz

If user wants to make PWM send waveforms in following format (PWM CLK = 24 MHz):

- Burst(20[F1]), i.e. 20 F1 pulses
- Burst(30[F2]),
- Burst(50[F1]) ,
- Burst(50[F2]),
- Burst(20[F1],10[F3]),
- Burst(30[F2],10[F3])

Step 1 Set carrier F1 frequency as 40 kHz, set duty cycle as 1/3.

- Set **PWM_TMAX0** as 0x258 (i.e. 24 MHz/40 kHz = 600 = 0x258).
- Since duty cycle is 1/3, set **PWM_TCMP0** as 0xc8 (i.e. 600/3 = 200 = 0xc8).
- Set carrier F2 frequency as 50 kHz, set duty cycle as 1/2.
- Set **PWM_TMAX0_SHADOW** as 0x1e0 (i.e. 24 MHz/50 kHz = 480 = 0x1e0).
- Since duty cycle is 1/2, set **PWM_TCMP0_SHADOW** as 0xf0 (i.e. 480/2 = 240 = 0xf0).

Step 2 Generate "FIFO CFG Data" sequence.

- Burst(20[F1]): {[15]: 1'b1, [14]: 1'b0, [13:0]: 'd20} = 0x8014.
- Burst(30[F2]): {[15]: 1'b1, [14]: 1'b1, [13:0]: 'd30} = 0xc01e.
- Burst(50[F1]) : {[15]: 1'b1, [14]: 1'b0, [13:0]: 'd50} = 0x8032.
- Burst(50[F2]): {[15]: 1'b1, [14]: 1'b1, [13:0]:'d50} = 0xc032.
- Burst(20[F1],10[F3]): {[15]: 1'b1, [14]: 1'b0, [13:0]: 'd20} = 0x8014,
  {[15]: 1'b0, [14]: 1'b0, [13:0]: 'd10} = 0x000a.
- Burst(30[F2],10[F3]): {[15]: 1'b1, [14]: 1'b1, [13:0]: 'd30} = 0xc01e,
  {[15]:1'b0, [14]: 1'b0, [13:0]: 'd10} = 0x000a.

Step 3 Write "FIFO CFG Data" into SRAM in DMA format.

- DMA SOURCE ADDRESS+0x00: 0x0000_0010 (DMA transfer-length: 16 bytes)
- DMA SOURCE ADDRESS+0x04: 0xc01e_8014 (little endian)
- DMA SOURCE ADDRESS+0x08: 0xc032_8032
- DMA SOURCE ADDRESS+0x0c: 0x000a_8014
- DMA SOURCE ADDRESS+0x10: 0x000a_c01e

Step 4 Enable DMA channel 5 to send PWM waveforms.

- Write 1'b1 to address 0x524[5] to enable DMA channel 5.

After all waveforms are sent, FIFO becomes empty, PWM0 will be disabled automatically (address 0x781[0] is automatically cleared). The FIFO mode stop interrupt flag bit (address 0x7b3[0]) will be automatically set as

1'b1. If the interrupt is enabled by setting PWM_MASK1 (address 0x7b2[0]) as 1'b1, a FIFO mode stop interrupt will be generated. User needs to write 1'b1 to the flag bit to manually clear it.

Example 2:

Suppose carrier frequency is 38 kHz, system clock frequency is 24 MHz, duty cycle is 1/3, and the format of IR code to be sent is shown as below:

- Preamble waveform: 9 ms carrier + 4.5 ms low level.
- Data 1 waveform: 0.56 ms carrier + 0.56 ms low level.
- Data 0 waveform: 0.56 ms carrier + 1.69 ms low level.
- Repeat waveform: 9 ms carrier + 2.25 ms low level + 0.56 ms carrier. Repeat waveform duration is 11.81 ms, interval between two adjacent repeat waveforms is 108 ms.
- End waveform: 0.56 ms carrier.

User can follow the steps below to configure related registers:

Step 1 Set carrier frequency as 38 kHz, set duty cycle as 1/3.

- Set **PWM_TMAX0** as 0x277 (i.e. 24 MHz/38 kHz = 631 = 0x277).
- Since duty cycle is 1/3, set **PWM_TCMP0** as 0xd2 (i.e. 631/3 = 210 = 0xd2).

Step 2 Generate "FIFO CFG Data" sequence.

- **Preamble waveform:**
  9 ms carrier: {[15]:1'b1, [14]:1'b0, [13:0]: 9*38='d 342=14'h 156} = 0x8156
  4.5 ms low level: {[15]:1'b0, [14]:1'b0, [13:0]: 4.5*38='d 171=14'h ab} = 0x00ab
- **Data 1 waveform:**
  0.56 ms carrier: {[15]:1'b1, [14]:1'b0, [13:0]: 0.56*38='d 21=14'h 15} = 0x8015
  0.56 ms low level: {[15]:1'b0, [14]:1'b0, [13:0]: 0.56*38='d 21=14'h 15} = 0x0015
- **Data 0 waveform:**
  0.56 ms carrier: {[15]:1'b1, [14]:1'b0, [13:0]: 0.56*38='d 21=14'h 15} = 0x8015
  1.69 ms low level: {[15]:1'b0, [14]:1'b0, [13:0]: 1.69*38='d 64=14'h 40} = 0x0040
- **Repeat waveform:**
  9 ms carrier: {[15]:1'b1, [14]:1'b0, [13:0]: 9*38='d 342=14'h 156} = 0x8156
  2.25 ms low level: {[15]:1'b0, [14]:1'b0, [13:0]: 2.25*38='d 86=14'h 56} = 0x0056
  0.56 ms carrier: {[15]:1'b1, [14]:1'b0, [13:0]: 0.56*38='d 21=14'h 15} = 0x8015
  108 ms - 11.81 ms = 96.19 ms low level:
  {[15]:1'b0, [14]:1'b0, [13:0]: 96.19*38='d 3655=14'h e47} = 0x0e47
- **End waveform:**
  0.56 ms carrier: {[15]:1'b1, [14]:1'b0, [13:0]: 0.56*38='d 21=14'h 15} = 0x8015

Step 3 Write "IR CFG Data" into SRAM in DMA format.

If user want PWM0 to send IR waveform in following format:

- Preamble+0x5a+Repeat+End
- Preamble: 0x8156, 0x00ab
- 0x5a = 8'b01011010
- Data 0: 0x8015, 0x0040
- Data 1: 0x8015, 0x0015
- Data 0: 0x8015, 0x0040

- Data 1: 0x8015, 0x0015
- Data 1: 0x8015, 0x0015
- Data 0: 0x8015, 0x0040
- Data 1: 0x8015, 0x0015
- Data 0: 0x8015, 0x0040
- Repeat: 0x8156, 0x0056, 0x8015, 0x0e47
- End: 0x8015.

User needs to write the configuration information above into source address of DMA channel 5, as shown below:

- DMA SOURCE ADDRESS+0x00: 0x0000_002e (DMA transfer-length: 46 bytes)
- DMA SOURCE ADDRESS+0x04: 0x00ab_8156 (Preamble) (little endian)
- DMA SOURCE ADDRESS+0x08: 0x0040_8015 (Data 0)
- DMA SOURCE ADDRESS+0x0c: 0x0015_8015 (Data 1)
- DMA SOURCE ADDRESS+0x10: 0x0040_8015 (Data 0)
- DMA SOURCE ADDRESS+0x14: 0x0015_8015 (Data 1)
- DMA SOURCE ADDRESS+0x18: 0x0015_8015 (Data 1)
- DMA SOURCE ADDRESS+0x1c: 0x0040_8015 (Data 0)
- DMA SOURCE ADDRESS+0x20: 0x0015_8015 (Data 1)
- DMA SOURCE ADDRESS+0x24: 0x0040_8015 (Data 0)
- DMA SOURCE ADDRESS+0x28: 0x0056_8156 (Repeat)
- DMA SOURCE ADDRESS+0x2c: 0x0e47_8015 (Repeat)
- DMA SOURCE ADDRESS+0x30: 0x8015 (End)

Step 4 Enable DMA channel 5 to send PWM waveforms.

  - Write 1'b1 to address 0x524[5] to enable DMA channel 5.

After all waveforms are sent, FIFO becomes empty, PWM0 will be disabled automatically (address 0x781[0] is automatically cleared). The FIFO mode stop interrupt flag bit (address 0x7b3[0]) will be automatically set as 1'b1. If the interrupt is enabled by setting PWM_MASK1 (address 0x7b2[0]) as 1'b1, a FIFO mode stop interrupt will be generated. User needs to write 1'b1 to the flag bit to manually clear it.

8.6 PWM Interrupt

There are 9 interrupt sources from PWM function.

After each signal frame, PWM#n (n = 0 ~ 5) will generate a frame-done IRQ (Interrupt Request) signal.

In Counting mode and IR mode, PWM0 will generate a Pnum IRQ signal after completing a pulse group.

In IR FIFO mode, PWM0 will generate a FIFO mode count IRQ signal when the FIFO_NUM value is less than the FIFO_NUM_LVL, and will generate a FIFO mode stop IRQ signal after FIFO becomes empty.

In IR DMA FIFO mode, PWM0 will generate an IR waveform send done IRQ signal, after DMA has sent all configuration data, FIFO becomes empty and final waveform is sent.

To enable PWM interrupt, the total enabling bit "irq_pwm" (address 0x641[6], see Chapter 6) should be set as 1'b1. To enable various PWM interrupt sources, PWM_MASK0 (address 0x7b0[7:0]) and PWM_MASK1 (address 0x7b2[0]) should be set as 1'b1 correspondingly.

Interrupt status can be cleared via register PWM_INT0 (address 0x7b1[7:0]) and PWM_INT1 (address 0x7b3[0]).

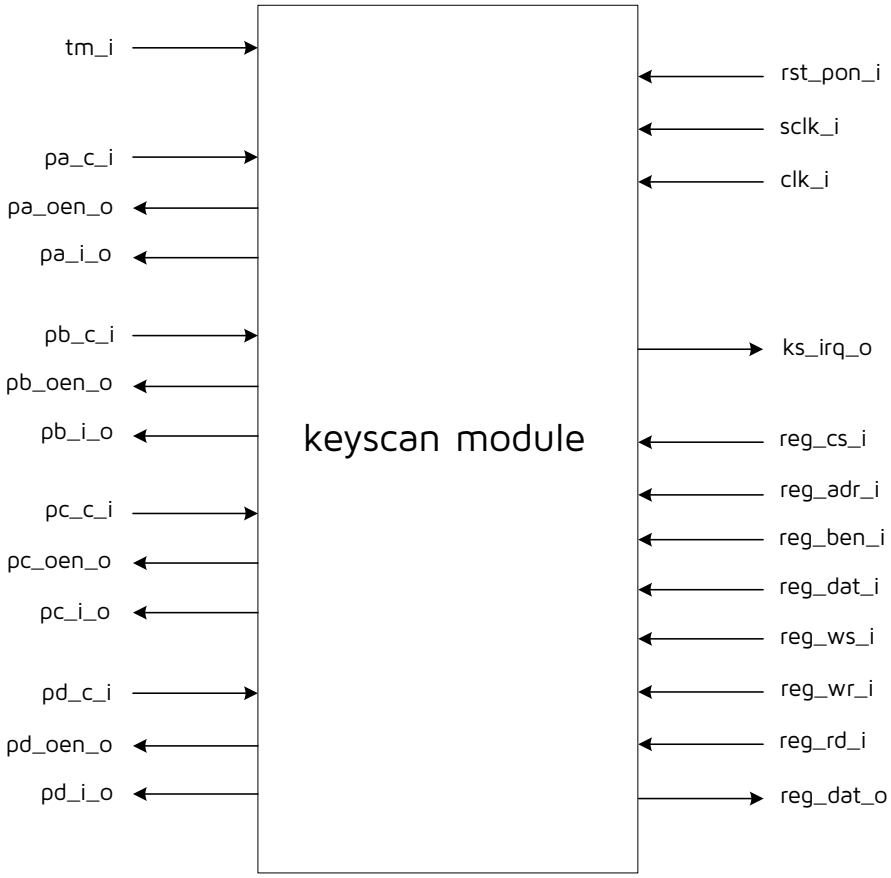
9 Keyscan

The TLSR8208 supports telink keyscan for detecting 8 rows x 16 columns matrix keyboards, which is essentially a row-column scan. Since the column scan is increased 3 at a time, it supports 8 x 18 keyboards. Telink keyscan supports hardware debounce function.

9.1 Signal Description

The keyscan interface signals are shown in Figure 9-1.

Figure 9-1 Keyscan Interface Signal



Descriptions of each signal are listed in Table 9-1.

Table 9-1 Keyscan controller Signal Definition

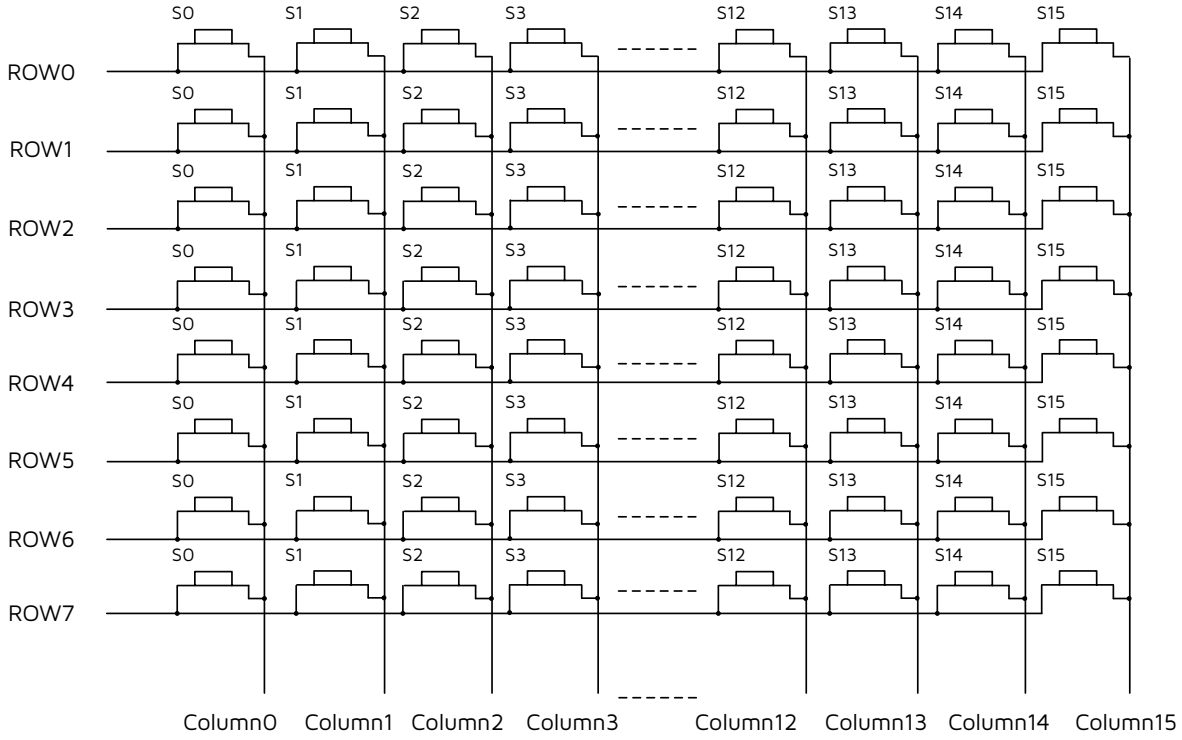
Signal Name	I/O Type	Description
tm_i	I	Test mode
rst_pon_i	I	Power on reset
clk_i	I	32kHz clock input
sclk_i	I	System clock input

Signal Name	I/O Type	Description
uart_irq_o	O	Interrupt
pa_c_i	I	Pa input
pa_oen_o	O	Pa oen
pa_i_o	O	Pa output
pb_c_i	I	Pb input
pb_oen_o	O	Pb oen
pb_i_o	O	Pb output
pc_c_i	I	Pc input
pc_oen_o	O	Pc oen
pc_i_o	O	Pc output
pd_c_i	I	Pd input
pd_oen_o	O	Pd oen
pd_i_o	O	Pd output
reg_cs_i	I	Bus cs
reg_adr_i	I	Bus address
reg_ben_i	I	Bus ben
reg_dat_i	I	Bus write data
reg_dat_o	O	Bus read data
reg_ws_i	I	Bus clock
reg_wr_i	I	Bus wrire
reg_rd_i	I	Bus read

9.2 Keyscan Principle

Telink keyscan uses the IO ports of PA, PB, PC, PD, PF (PA[1]/PA[2]/PA[3] are not available) to control the reading of keys. 8 I/O lines are used as row lines and 16 I/O lines are used as column lines to form the keyboard, and one key is set at each intersection of the row and column lines, the number of keys is 8 x 16. The keyscan performs a row-column scan to determine the row and column in which the key is located and to determine the key value.

Figure 9-2 Keyscan Schematic



Set address 0x804~0x808 (KS_ROW_SEL) to select 8 pins as rows, then set address 0x800~0x803 (KS_COL_MSK) to select 16 pins from the remaining pins as columns.

The keyscan scans keyboards via row-column selection. After scanning, the row-column number is recorded in the KS_KEY buffer, the corresponding row-column number is read in the end_flag interrupt to determine the key.

NOTE:

- PD[0] is used for rows by default, cannot be used for columns.
- For column, use PB[4:0], PC[7:0], PD[7:1] as much as possible.
- Row number is fixed to 8, if less than 8 rows are to be used, make the unused pins floating.

9.3 Register Table

Table 9-2 Register Table for Keyscan

Address	R/W	Description	Default Value
0x800	RW	KS_COL_MSK0 Keyscan column mask for pd[7:0]	0x00
0x801	RW	KS_COL_MSK1 Keyscan column mask for pc[7:0]	0x00

Address	R/W	Description	Default Value
0x802	RW	KS_COL_MSK2 Keyscan column mask for pb[7:0]	0x00
0x803	RW	KS_COL_MSK3 Keyscan column mask for pa[7:0]	0x00
0x804	RW	KS_ROW_SEL0 [4:0]: keyscan row select for row0 [7:5]: keyscan row select for row1[2:0]	0x00
0x805	RW	KS_ROW_SEL1 [1:0]: keyscan row select for row1[4:3] [6:2]: keyscan row select for row2 [7]: keyscan row select for row3[0]	0x00
0x806	RW	KS_ROW_SEL2 [3:0]: keyscan row select for row3[4:1] [7:4]: keyscan row select for row4[3:0]	0x00
0x807	RW	KS_ROW_SEL3 [0]: keyscan row select for row4[4] [5:1]: keyscan row select for row5 [7:6]: keyscan row select for row6[1:0]	-
0x808	RW	KS_ROW_SEL4 [2:0]: keyscan row select for row6[4:2] [7:3]: keyscan row select for row7	0x00
0x809	RW	KS_END_FLG Keyscan frame end flag	0xff
0x80a	RW	KS_EN [0]: Keyscan enable [1]: Keyscan 32k Hz clock enable [2]: Keyscan interrupt enable [3]: Keyscan input invert [4]: Keyscan output invert [5]: Keyscan scan mode select, 1'b0 for mode 0, 1'b1 for mode 1 [6]: Keyscan manually reset [7]: Keyscan tripple check disable	0x07

Address	R/W	Description	Default Value
0x80b	RW	KS_FRM_NUM [4:0]: Keyscan empty frame counter number [7:5]: debounce_period: debounce period 2->8ms, 3->12ms, 4->16ms, 5->20ms	0x41
0x80c	R	KS_IRQ [4:0]: Keyscan read pointer for key buffer [7]: Keyscan interrupt	0x00
0x80d	R	KS_RPTR [4:0]: Keyscan latched write pointer when frame end [6]: Keyscan cap key detect when in any state [7]: Keyscan state, 1'b0 for IDLE, 1'b1 for SCAN	0x00
0x80e	R	KS_WPTR [4:0]: Keyscan write pointer for key buffer [5]: Keyscan no key detect when in SCAN state [6]: Keyscan key detect when in IDLE state [7]: Keyscan internal counter128 count enable	0x00
0x80f	R	KS_GATED [2:0]: Keyscan counter128[6:4] [3]: Reserved [4]: Keyscan 32k Hz clock gated clear [5]: Keyscan 32k Hz clock gated [6]: Keyscan internal counter16 count enable [7]: Reserved	0x00
0x810	-	KS_KEY Keyscan key value	0xff
0x811	R	KS_LPTR [4:0]: Keyscan loop pointer [7:5]: Reserved	0x00
0x812	R	KS_CNT128 [6:0]: Keyscan counter128 count value [7]: Reserved	0x00

Address	R/W	Description	Default Value
0x813	R	KS_CNT16 [3:0]: Keyscan counter16 count value [6:4]: Keyscan latched row number [7]: Reserved	0x00

9.4 Hardware Debounce

Telink keyscan implements the hardware debounce function. The dynamic range of debounce supports 8ms/ 12ms/16ms/20ms/24ms multi-block debounce cycle. The keyscan compares the data of adjacent debounce cycles, and only deposit it into the FIFO for reporting if they are consistent.

9.5 Susepend/Wakup

The keyscan is able to wake up the system following the debounce cycle. The Keyscan module works on a crystal clock or 32 kHz RC oscillator, so it can work when it is asleep. The interrupts are reported following the debounce cycle (8ms/12ms/16ms), or application layer readouts at regular intervals. The purpose of this is to reduce the time spent on the temporary UI and to reduce power consumption.

9.6 FIFO Depth

FIFO is expected to hold 31 keys. If less than 8 keys are expected per debounce cycle, it is necessary to remove all keys in four debounce cycles.

10 Quadrature Decoder

The TLSR8208 embeds one quadrature decoder (QDEC) which is designed mainly for applications such as wheel. The QDEC implements debounce function to filter out jitter on the two phase inputs, and generates smooth square waves for the two phase.

10.1 Input Pin Selection

The QDEC supports two phase input; each input is selectable from the 8 pins of PortA, PortB, PortC and PortD via setting address 0xd2[2:0] (for channel a)/0xd3[2:0] (for channel b).

Table 10-1 Input Pin Selection

Address 0xd2[2:0]/0xd3[2:0]	Pin
0	PA[2]
1	PA[3]
2	PB[6]
3	PB[7]
4	PC[2]
5	PC[3]
6	PD[6]
7	PD[7]

NOTE: *To use corresponding IO as QDEC input pin, it's needed first to enable GPIO function, enable "IE" (1) and disable "OEN" (1) for this IO.*

10.2 Common Mode and Double Accuracy Mode

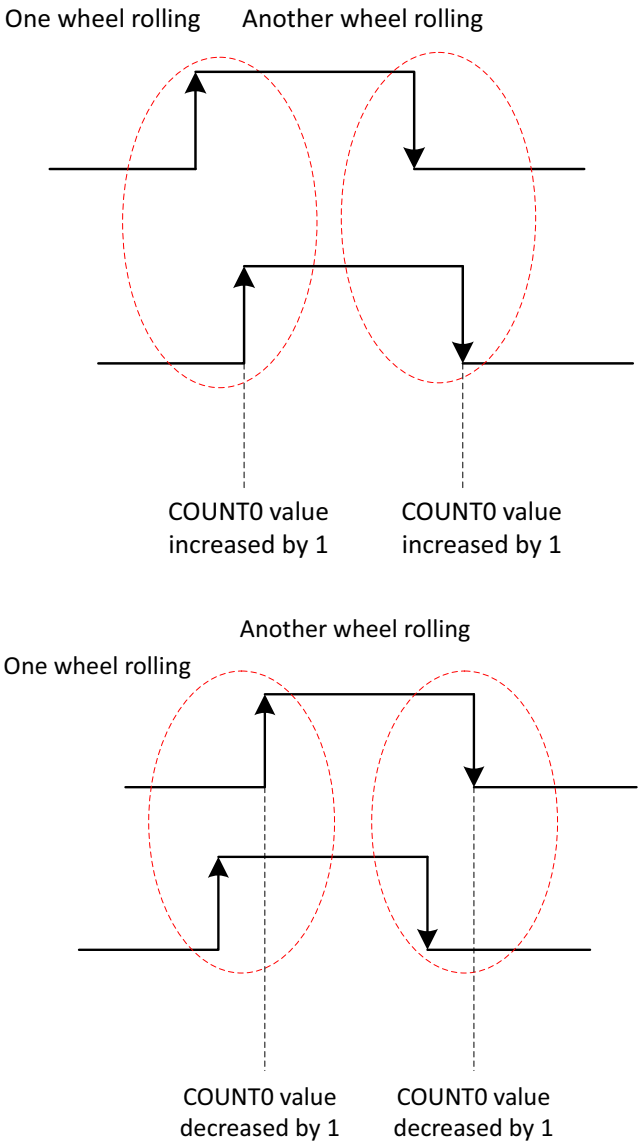
The QDEC embeds an internal hardware counter, which is not connected with bus.

Address 0xd7[0] serves to select common mode or double accuracy mode.

For each wheel rolling step, two pulse edges (rising edge or falling edge) are generated.

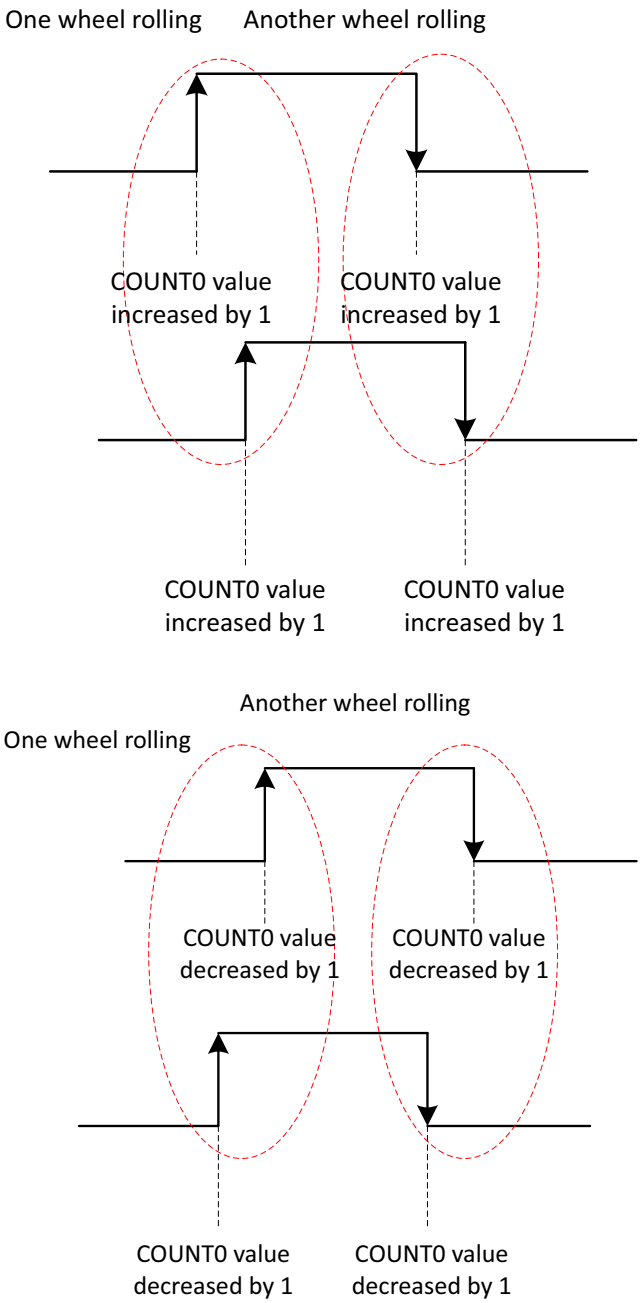
If address 0xd7[0] is cleared to select common mode, the QDEC Counter value (real time counting value) is increased/decreased by 1 only when the same rising/falling edges are detected from the two phase signals.

Figure 10-1 Common Mode



If address 0xd7[0] is set to 1'b1 to select double accuracy mode, the QDEC Counter value (real time counting value) is increased/decreased by 1 on each rising/falling edge of the two phase signals; the COUNT0 will be increased/decreased by 2 for one wheel rolling.

Figure 10-2 Double Accuracy Mode

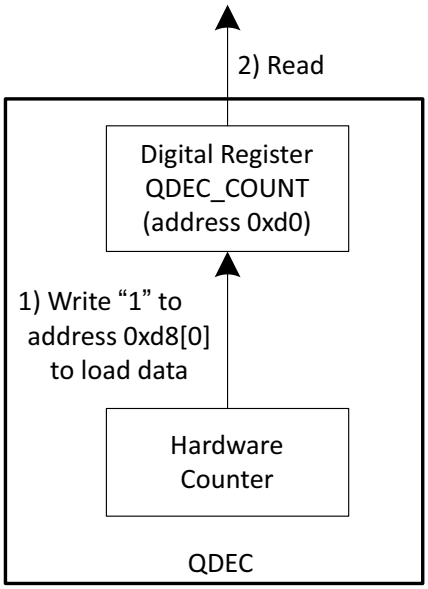


10.3 Read Real Time Counting Value

Neither can Hardware Counter value be read directly via software, nor can the counting value in address 0xd0 be updated automatically.

To read real time counting value, first write address 0xd8[0] with 1'b1 to load Hardware Counter data into the QDEC_COUNT register, then read address 0xd0.

Figure 10-3 Read Real Time Counting Value



10.4 QDEC Reset

Address 0x60[5] serves to reset the QDEC. The QDEC Counter value is cleared to zero.

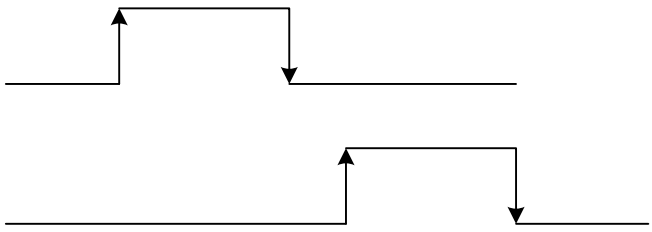
10.5 Other Configuration

The QDEC supports hardware debouncing. Address 0xd1[2:0] serves to set filtering window duration. All jitter with period less than the value will be filtered out and thus does not trigger count change.

Address 0xd1[4] serves to set input signal initial polarity.

Address 0xd1[5] serves to enable shuttle mode. Shuttle mode allows non-overlapping two phase signals as shown in the following figure.

Figure 10-4 Shuttle Mode



10.6 Timing Sequence

Figure 10-5 Timing Sequence Chart

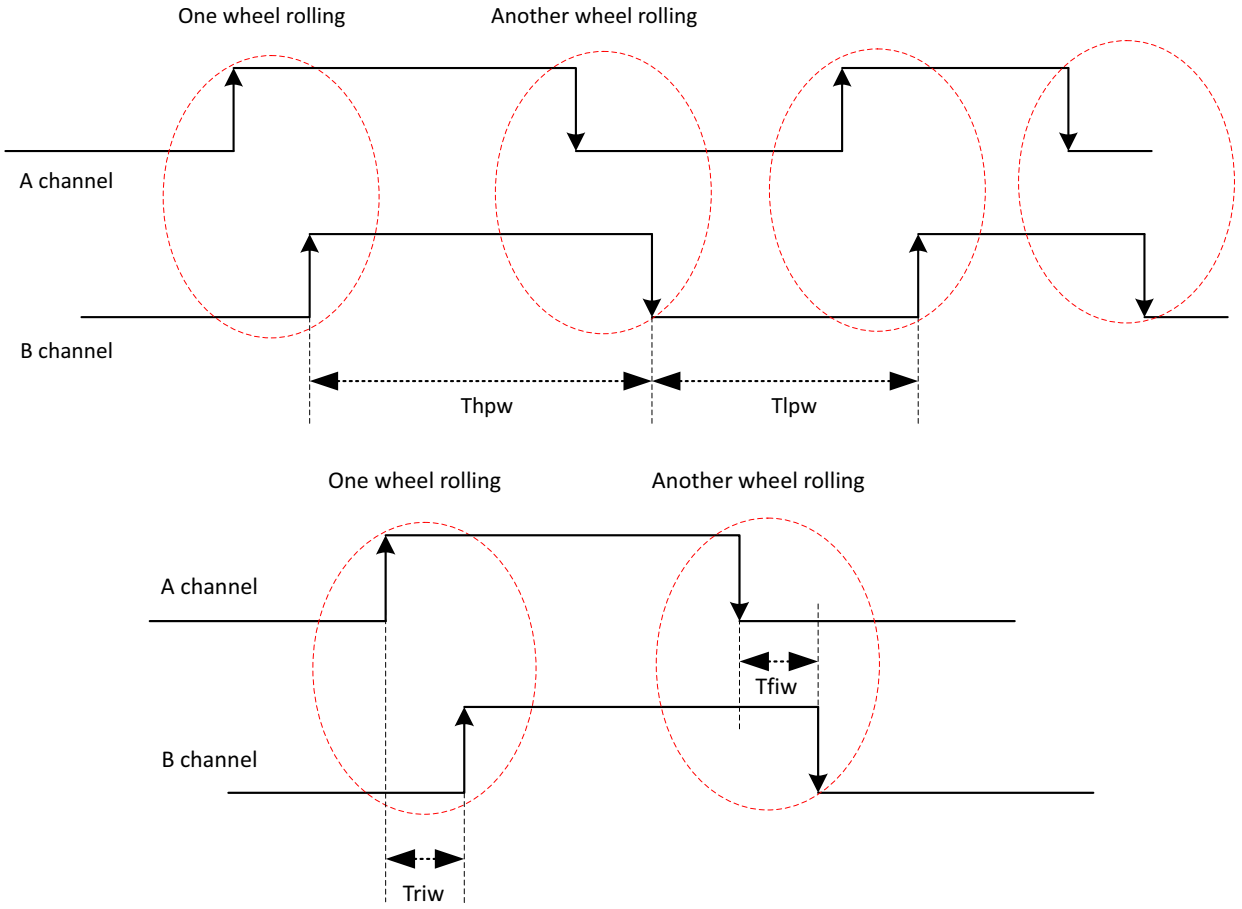


Table 10-2 Timing

Time Interval	Min Value
Thpw (High-level pulse width)	2^(n+1) *clk_32kHz *3 (n=0xd1[2:0])
Tlpw (Low-level pulse width)	2^(n+1) *clk_32kHz *3 (n=0xd1[2:0])
Triw (Interval width between two rising edges)	2^(n+1) *clk_32kHz (n=0xd1[2:0])
Tfiw (Interval width between two falling edges)	2^(n+1) *clk_32kHz (n=0xd1[2:0])

QDEC module works based on 32 kHz clock to ensure it can work in suspend mode. QDEC module supports debouncing function, and any signal with width lower than the threshold (i.e. "2^(n+1) *clk_32kHz *3 (n=0xd1[2:0])) will be regarded as jitter. Therefore, effective signals input from Channel A and B should contain high/low level with width Thpw/Tlpw more than the threshold. The 2^n *clk_32kHz clock is used to synchronize input signal of QDEC module, so the interval between two adjacent rising/falling edges from Channel A and B, which are marked as Triw and Tfiw, should exceed "2^(n+1) *clk_32kHz".

Only when the timing requirements above are met, can QDEC module recognize wheel rolling times correctly.

10.7 Register Table

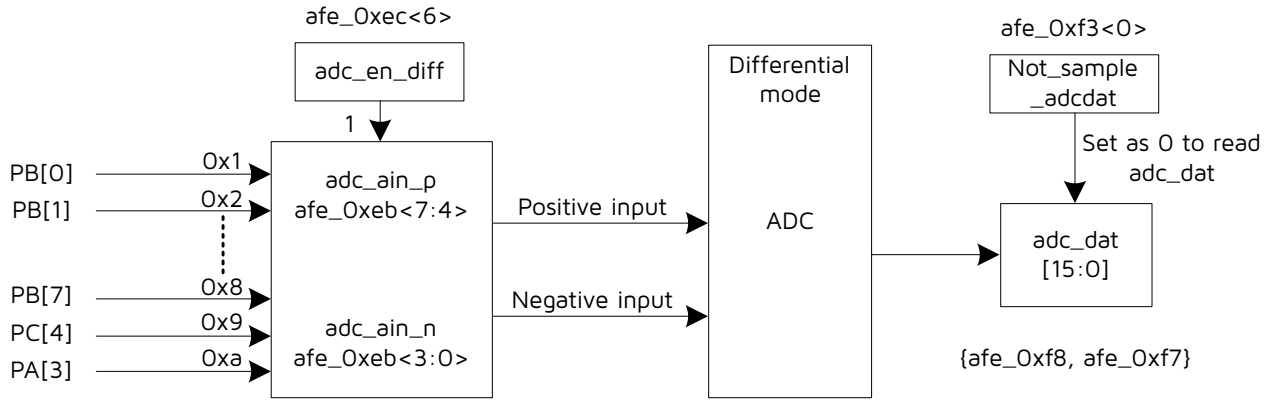
Table 10-3 Register Table for QDEC

Address	R/W	Description	Default Value
0xd0	R	QDEC Counting value (read to clear): Pulse edge number	0x00
0xd1	RW	[5]: shuttle mode 1 - enable shuttle mode [4]: pola, input signal pola 0 - no signal is low, 1 - no signal is high [2:0]: filter time (can filter 2^n *clk_32k*2 width deglitch)	0x00
0xd2	RW	[2:0]: QDEC input pin select for channel A, choose 1 of 8 pins for input channel A 7~0: {PD[7:6], PC[3:2], PB[7:6], PA[3:2]}	0x00
0xd3	RW	[2:0]: QDEC input pin select for channel B, choose 1 of 8 pins for input channel B 7~0: {PD[7:6], PC[3:2], PB[7:6], PA[3:2]}	0x01
0xd6	R	[0]: RSVD	0x00
0xd7	RW	[0]: Enable double accuracy mode	0x01
0xd8	RW	[0]: write 1 to load data When load completes it will be 0.	0x00

11 SAR ADC

The TLSR8208 integrates one SAR ADC module, which can be used to sample analog input signals such as battery voltage and temperature sensor.

Figure 11-1 Block Diagram of ADC



11.1 Power On/Down

The SAR ADC is disabled by default. To power on the ADC, the analog register adc_pd (afe_0xfc<5>) should be set as 1'b0.

11.2 ADC Clock

ADC clock is derived from external 24 MHz crystal source, with frequency dividing factor configurable via the analog register adc_clk_div (afe_0xf4<2:0>).

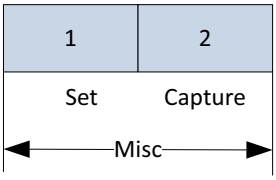
$$ADC\ clock\ frequency\ (marked\ as\ F_{ADC\_clk}) = 24\ MHz/(adc\_clk\_div+1)$$

11.3 ADC Control in Auto Mode

11.3.1 Set Max State and Enable Channel

The SAR ADC supports Misc channel which consists of one "Set" state and one "Capture" state.

- The analog register r_max_scnt (afe_0xf2<5:4>) serves to set the max state index. As shown below, the r_max_scnt should be set as 0x02.



- The Misc channel can be enabled via r_en_misc (afe_0xf2<2>).

11.3.2 "Set" State

The length of "Set" state for the Misc channel is configurable via the analog register r_max_s (afe_0xf1<3:0>).

$$\text{"Set" state duration (marked as } T_{sd}) = r\_max\_s / 24\ MHz$$

Each "Set" state serves to set ADC control signals for the Misc channel via corresponding analog registers, including:

- adc_en_diff: afe_0xec<6>. MUST set as 1'b1 to select differential input mode.
- adc_ain_p: afe_0xeb<7:4>. Select positive input in differential mode.
- adc_ain_n: afe_0xeb<3:0>. Select negative input in differential mode.
- adc_vref: afe_0xea<1:0>. Set reference voltage V_{REF}. ADC maximum input range is determined by the ADC reference voltage.
- adc_sel_ai_scale: afe_0xfa<7:6>. Set scaling factor for ADC analog input as 1 (default), or 1/8.

By setting this scaling factor, ADC maximum input range can be extended based on the V_{REF}.

For example, suppose the V_{REF} is set as 1.2 V:

Since the scaling factor is 1 by default, the ADC maximum input range should be 0 ~ 1.2 V (negative input is GND) / -1.2 V ~ +1.2 V (negative input is ADC GPIO pin).

If the scaling factor is set as 1/8, in theory ADC maximum input range should change to 0 ~ 9.6 V (negative input is GND) / -9.6 V ~ +9.6 V (negative input is ADC GPIO pin). But limited by input voltage of the chip's PAD, the actual range is narrower.

- adc_res: afe_0xec<1:0>. Set resolution as 8/10/12/14 bits.

ADC data is always 16-bit format no matter what the resolution is set. For example, 14 bits resolution indicates ADC data consists of 14-bit valid data and 2-bit sign extension bit.

- adc_tsamp: afe_0xee<3:0>. Set sampling time which determines the speed to stabilize input signals.

$$\text{Sampling time (marked as } T_{samp}) = adc\_tsamp / F_{ADC\_clk}$$

The lower sampling cycle, the shorter ADC convert time.

11.3.3 "Capture" State

For the Misc channel, at the beginning of its "Capture" state, a "run" signal is issued automatically to start an ADC sampling and conversion process; at the end of "Capture" state, ADC output data is captured.

- The length of "Capture" state is configurable via the analog register r_max_mc[9:0] (afe_0xf1<7:6>, afe_0xef<7:0>).

$$\text{"Capture" state duration for Misc channel (marked as } T_{cd}) = r\_max\_mc / 24\ MHz$$

- The "VLD" bit (afe_0xf6<0>) will be set as 1'b1 at the end of "Capture" state to indicate the ADC data is valid, and this flag bit will be cleared automatically.
- The 16-bit ADC output data can be read from the analog register adc_dat[15:0] (afe_0xf8<7:0>, afe_0xf7<7:0>) while the afe_0xf3<0> is set as 1'b0 (default). If the afe_0xf3<0> is set as 1'b1, the data in the afe_0xf8 and afe_0xf7 won't be updated.

NOTE: *The total duration "T_{td}", which is the sum of the length of "Set" state and "Capture" state, determines the sampling rate.*

$$\text{Sampling frequency (marked as } F_s) = 1 / T_{td}$$

11.3.4 Usage Case with Detailed Register Setting

This case introduces the register setting details for Misc channel sampling.

In this case, afe_0xf2<2> should be set as 1'b1, so as to enable the Misc channel, while the max state index should be set as "2" by setting afe_0xf2<5:4> as 0x2.

*The total duration (marked as T_{td}) = (1*r_max_s + 1*r_max_mc) / 24 MHz*

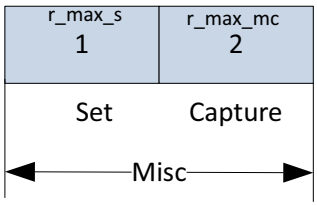


Table 11-1 Overall Register Setting

Function	Register Setting
Power on the ADC	afe_0xfc<5> = 1'b0
Set F_{ADC_clk} (ADC clock frequency) as 4 MHz	afe_0xf4<2:0> = 5 F_{ADC_clk} = 24 MHz/(5+1) = 4 MHz
Enable the Misc channel	afe_0xf2<2> = 1'b1
Set the max state index as "2"	afe_0xf2<5:4> = 2'b10
Set T_{sd} ("Set" state duration)	afe_0xf1<3:0> = 10 T_{sd} = r_max_s/24 MHz = 10/24 MHz = 0.417 μs
Set T_{cd} ("Capture" state duration)	afe_0xf1<7:6> = 1, afe_0xef<7:0> = 0xea T_{cd} = r_max_mc[9:0]/24 MHz = 490/24 MHz = 20.417 μs
T_{td} (total duration)	T_{td} = (1*r_max_s+1*r_max_mc) / 24 MHz = 500/24 MHz = 20.83 μs
F_s (Sampling frequency)	F_s = 1 / T_{td} = 24 MHz/500 = 48 kHz
Set differential input	afe_0xec<6> = 1
Set input channel	afe_0xeb = 0x12 Select PB[0] as positive input and PB[1] as negative input
Set reference voltage V_{REF}	afe_0xea<1:0> = 2 V_{REF} = 1.2 V
Set scaling factor for ADC analog input	afe_0xfa<7:6> = 0 scaling factor: 1
	ADC maximum input range: -1.2 V ~ +1.2 V

Function	Register Setting
Set resolution	afe_0xec<1:0> = 3 resolution: 14 bits
Set T_{samp} (determines the speed to stabilize input before sampling)	afe_0xee<3:0> = 3 T_{samp} = adc_tsamp / F_{ADC_clk} = 12/4 MHz = 3 μs

11.4 Register Table

Table 11-2 Register Table Related to SAR ADC

Address	Description	Default Value
afe_0xea<1:0>	Select V_{REF} for Misc channel 0x0: RSVD 0x1: 0.9 V 0x2: 1.2 V 0x3: RSVD	00
afe_0xea<7:2>	Reserved	-
afe_0xeb<3:0>	Select negative input for Misc channel: 0x0: No input 0x1: B[0] 0x2: B[1] 0x3: B[2] 0x4: B[3] 0x5: B[4] 0x6: B[5] 0x7: B[6] 0x8: B[7] 0x9: C[4] 0xa: A[3] 0xb: RSVD 0xc: RSVD 0xd: RSVD 0xe: new tempsensor_n (Temperature sensor negative output) 0xf: Ground	0000

Address	Description	Default Value
afe_0xeb<7:4>	Select positive input for Misc channel: 0x0: No input 0x1: B[0] 0x2: B[1] 0x3: B[2] 0x4: B[3] 0x5: B[4] 0x6: B[5] 0x7: B[6] 0x8: B[7] 0x9: C[4] 0xa: A[3] 0xb: RSVD 0xc: RSVD 0xd: RSVD 0xe: new tempsensor_n (Temperature sensor negative output) 0xf: vbatdiv	0000
afe_0xec<1:0>	Set resolution for Misc channel 0x0: 8 bits 0x1: 10 bits 0x2: 12 bits 0x3: 14 bits	11
afe_0xec<5:2>	Reserved	-
afe_0xec<6>	Select input mode for Misc channel. 0: RSVD 1: differential mode	0
afe_0xec<7>	Reserved	-

Address	Description	Default Value
afe_0xee<3:0>	Number of ADC clock cycles in sampling phase for Misc channel to stabilize the input before sampling: 0x0: 3 cycles 0x1: 6 cycles 0x2: 9 cycles 0x3: 12 cycles ... 0xf: 48 cycles	0000
afe_0xef<7:0>	r_max_mc[9:0] serves to set length of "capture" state for Misc channel. r_max_s serves to set length of "set" state for Misc channel. Note: State length indicates number of 24M clock cycles occupied by the state.	-
afe_0xf0<7:0>		-
afe_0xf1<3:0>		-
afe_0xf1<5:4>		-
afe_0xf1<7:6>		-
afe_0xf2<0>	Reserved	-
afe_0xf2<1>	Reserved	-
afe_0xf2<2>	Enable Misc channel sampling. 1: enable	-?
afe_0xf2<3>	0: enable write to core 1: disable write to core	0
afe_0xf2<5:4>	Set total length for sampling state machine (i.e. max state index)	00
afe_0xf2<7>	Reserved	-
afe_0xf3<0>	0: sample ADC data to afe_0xf8 and afe_0xf7 1: not sample ADC data to afe_0xf8 and afe_0xf7	0
afe_0xf3<7:2>	Reserved	-
afe_0xf4<2:0>	ADC clock (derive from external 24M crystal) ADC clock frequency = 24M/(adc_clk_div+1)	011
afe_0xf4<7:3>	Reserved	-
afe_0xf5<7:0>	Reserved	-
afe_0xf6<0>	[0]: vld, ADC data valid status bit (This bit will be set as 1 at the end of capture state to indicate the ADC data is valid, and will be cleared when set state starts.)	-
afe_0xf6<7:1>	Reserved	-

Address	Description	Default Value
afe_0xf7<7:0>	Read only [7:0]: Misc adc dat[7:0]	-
afe_0xf8<7:0>	Read only [7:0]: Misc adc_dat[15:8]	-
afe_0xf9<3:2>	Reserved	00
afe_0xfa<7:6>	Analog input pre-scaling select sel_ai_scale[1:0]: scaling factor 0x0: 1 0x1: RSVD 0x2: RSVD 0x3: 1/8	0
afe_0xfc<4>	Reserved	0
afe_0xfc<5>	Power down ADC 1: Power down 0: Power up	1

12 AES

The TLSR8208 embeds AES module with encryption and decryption function. The input 128-bit plaintext in combination of key is converted into the final output ciphertext via encryption; the 128-bit ciphertext in combination of key can also be converted into 128-bit plaintext via decryption.

The AES hardware accelerator provides automatic encryption and decryption. It only takes (1000*system clock cycles) to implement AES encryption/decryption. Suppose system clock is 20 MHz, the time needed for AES encryption/decryption is 50 μs.

Both RISC mode and DMA mode are supported for AES operation.

12.1 RISC Mode

For RISC mode, configuration of related registers is as follows:

- Set the value of key via writing registers AES_KEY0 ~ AES_KEY15 (address 0x710 ~ 0x71f).
- Set operation method of AES module via register AES_CTRL: set address 0x700[0] as 1'b1 for decryption method, while clear this bit for encryption method.
- For encryption method, write registers AES-DAT0 ~ AES-DAT3 (address 0x708 ~ 0x70b) for four times to set the 128-bit plaintext. After encryption, the 128-bit ciphertext can be obtained by reading address 0x708 ~ 0x70b for four times.
- For decryption method, write registers AES-DAT0 ~ AES-DAT3 (address 0x708 ~ 0x70b) for four times to set the 128-bit ciphertext. After decryption, the 128-bit plaintext can be obtained by reading address 0x708 ~ 0x70b for four times.
- Address 0x700 bit[1] and bit[2] are read only bits: bit[1] will be cleared automatically after quartic writing of address 0x708 ~ 0x70b; bit[2] will be set as 1 automatically after encryption/decryption, and then cleared automatically after quartic reading of address 0x708 ~ 0x70b.

12.2 DMA Mode

As for DMA mode, it is only needed to configure the value of key and encryption/decryption method for AES module.

12.3 AES-CCM

The AES-CCM (Counter with the CBC-MAC) mode is disabled by default. AES output is directly determined by current encryption and decryption, irrespective of previous encryption and decryption result.

If 0x700[7] is set as 1'b1 to enable AES-CCM mode, AES output will also take previous encryption and decryption result into consideration.

12.4 Register Table

Table 12-1 Register Table Related to AES

Address	R/W	Description	Default Value
0x700	RW	[0] Select decrypt/encrypt 1: decrypt, 0: encrypt [1] 1: input data needed, 0: input data ready (R) [2] 0: output data not ready, 1: output data ready (R) [7] 1: enable AES-CCM mode	0x02
0x701	R	[1:0] write/read data count	0x00
0x708	RW	Input/Output Data byte 0	0x00
0x709	RW	Input/Output Data byte 1	0x00
0x70a	RW	Input/Output Data byte 2	0x00
0x70b	RW	Input/Output Data byte 3	0x00
0x710	RW	[7:0] KEY0	0x00
0x711	RW	[7:0] KEY1	0x00
0x712	RW	[7:0] KEY2	0x00
0x713	RW	[7:0] KEY3	0x00
0x714	RW	[7:0] KEY4	0x00
0x715	RW	[7:0] KEY5	0x00
0x716	RW	[7:0] KEY6	0x00
0x717	RW	[7:0] KEY7	0x00
0x718	RW	[7:0] KEY8	0x00
0x719	RW	[7:0] KEY9	0x00
0x71a	RW	[7:0] KEY10	0x00
0x71b	RW	[7:0] KEY11	0x00
0x71c	RW	[7:0] KEY12	0x00
0x71d	RW	[7:0] KEY13	0x00
0x71e	RW	[7:0] KEY14	0x00
0x71f	RW	[7:0] KEY15	0x00

13 Key Electrical Specifications

NOTE: *The electrical characteristics currently listed in this section are target specifications and only supplied for reference. Some data may be updated according to actual test results.*

13.1 Absolute Maximum Ratings

Table 13-1 Absolute Maximum Ratings

Item	Sym.	Min	Max	Unit	Conditions
Supply Voltage	VBAT	-0.3	4.2	V	-
Voltage on a Pin	V_{In}	-0.3	VDD + 0.3	V	VDD from internal LDO output
Storage temperature range	T_{Str}	-65	150	°C	-
Soldering temperature	T_{Sld}	-	260	°C	-

CAUTION: *Stresses above those listed in "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress only rating and operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.*

13.2 Recommended Operating Conditions

Table 13-2 Recommended Operating Conditions

Item	Sym.	Min	Typ	Max	Unit	Conditions
Power-supply voltage	VDD	1.8	3.3	3.6	V	For applications that without OTP write
	VDD	2.4	3.3	3.6	V	For applications that with OTP write
Supply rise time (from 1.2 V to 1.5 V)	t_R	-	-	10	ms	-
Operating temperature range	T_{Opr}	-40	-	85	°C	-

13.3 DC Characteristics

VDD = 3.3 V, T = 25°C unless otherwise stated.

Table 13-3 DC Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
RX current	I_{Rx}	-	9.1	-	mA	Whole Chip, load RX bin file, switch to frequency, disable
TX current	I_{Tx}	-	9.5	-	mA	Whole chip @ 0 dBm with LDO
Deep sleep with 16 KB SRAM retention	I_{Deep1}	-	0.9	-	μA	Without 32K RC[a]
Deep sleep without SRAM retention	I_{Deep2}	-	0.55	-	μA	
Deep sleep with 16 KB SRAM retention	I_{Deep3}	-	1.3	-	μA	With 32K RC[b]
Deep sleep without SRAM retention	I_{Deep4}	-	1.0	-	μA	

a. Without 32K RC: The wakeup source is external signal from GPIO input, the internal 32K RC is disabled.

b. With 32K RC: The wakeup source is 32K RC, it is enabled.

13.4 AC Characteristics

VDD = 3.3 V, T = 25°C unless otherwise stated.

Table 13-4 Digital Inputs/Outputs Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
Input high voltage	VIH	0.7VDD	-	VDD	V	-
Input low voltage	VIL	VSS	-	0.3VDD	V	-
Output high voltage	VOH	0.9VDD	-	VDD	V	-
Output low voltage	VOL	VSS	-	0.1VDD	V	-

Table 13-5 RF Performance Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
RF frequency range	-	2400	-	2483.5	MHz	Programmable in 1 MHz step
Data rate	BLE/2.4G proprietary 1 Mbps, ±250 kHz deviation BLE/2.4G proprietary 2 Mbps, ±500 kHz deviation 2.4G proprietary 1 Mbps, ±250 kHz deviation 2.4G proprietary 2 Mbps, ±500 kHz deviation					

Item		Sym.	Min	Typ	Max	Unit	Conditions
BLE 1 Mbps RF_RX Performance (±250 kHz Deviation)							
Sensitivity	1 Mbps	-	-	-97	-	dBm	-
Frequency offset tolerance		-	-250	-	+300	kHz	-
Co-channel rejection		-	-	8	-	dB	Wanted signal at -67 dBm
In-band blocking rejection (equal modulation interference)	+1/-1 MHz offset	-	-	-4/-2	-	dB	Wanted signal at -67 dBm
	+2/-2 MHz offset	-	-	-41/-32	-	dB	
	≥ 3 MHz offset	-	-	-42	-	dB	
Image rejection		-	-	-32	-	dB	Wanted signal at -67 dBm
BLE 1 Mbps RF_TX Performance							
Output power, maximum setting		-	-	10	-	dBm	-
Output power, minimum setting		-	-	-45	-	dBm	-
Programmable output power range		-	55			dB	-
Modulation 20 dB bandwidth		-	-	1.4	-	MHz	-
BLE 2 Mbps RF_RX Performance (±500 kHz Deviation)							
Sensitivity	2 Mbps	-	-	-93	-	dBm	-
Frequency offset tolerance		-	-300	-	+200	kHz	-
Co-channel rejection		-	-	8	-	dB	Wanted signal at -67 dBm
In-band blocking rejection	+2/-2 MHz offset	-	-	-9/-7	-	dB	Wanted signal at -67 dBm
	+4/-4 MHz offset	-	-	-38/-33	-	dB	
	> 4 MHz offset	-	-	-42	-	dB	

Item	Sym.	Min	Typ	Max	Unit	Conditions
Image rejection	-	-	-26	-	dB	Wanted signal at -67 dBm; image frequency = RF_channel - 3 MHz
BLE 2 Mbps RF_TX Performance						
Output power, maximum setting	-	-	10	-	dBm	-
Output power, minimum setting	-	-	-45	-	dBm	-
Programmable output power range	-		55		dB	-
Modulation 20 dB bandwidth	-	-	2.5	-	MHz	-

Table 13-6 RSSI Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
RSSI range	-	-100	-	10	dBm	-
Resolution	-	-	±1	-	dB	-

Table 13-7 Crystal Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
24 MHz Crystal						
Nominal frequency (parallel resonant)	f_{NOM}	-	24	-	MHz	-
Frequency tolerance	f_{TOL}	-20	-	+20	ppm	-
Load capacitance	C_L	5	12	18	pF	Programmable on chip load cap
Equivalent series resistance	ESR	-	50	100	Ohm	-
32.768 kHz Crystal						
Nominal frequency (parallel resonant)	f_{NOM}	-	32.768	-	kHz	-

Item	Sym.	Min	Typ	Max	Unit	Conditions
Frequency tolerance	f_{TOL}	-100	-	+100	ppm	-
Load capacitance	C_L	6	9	12.5	pF	Programmable on chip load cap
Equivalent series resistance	ESR	-	50	80	kOhm	-

Table 13-8 RC Oscillator Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
24 MHz RC Oscillator						
Nominal frequency	f_{NOM}	-	24	-	MHz	-
Frequency tolerance	f_{TOL}	-	-	1	%	On chip calibration
32 kHz RC Oscillator						
Nominal frequency	f_{NOM}	-	32	-	kHz	-
Frequency tolerance	f_{TOL}	-	-	0.03	%	On chip calibration
Calibration time	-	-	3	-	ms	-

Table 13-9 ADC Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
Differential nonlinearity	DNL	-	-	1	LSB	10-bit resolution mode
Integral nonlinearity	INL	-	-	2	LSB	10-bit resolution mode
Signal-to-noise and distortion ratio	SINAD	-	70	-	dB	f_{IN} = 1 kHz, f_S= 16 kHz
Effective number of bits	ENOB	-	10.5	-	bits	-
Sampling frequency	Fs	-	-	200	ksps	-

13.5 SPI Characteristics

Over process, voltage 1.9 ~ 3.6 V, T = -40 ~ +85°C unless otherwise stated.

Table 13-10 SPI Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
CK frequency	F_{CK}	-	-	4	MHz	Slave
CK duty cycle clock	-	-	50	-	%	Master
DI setup time	-	30	-	-	ns	Slave
	-	90	-	-	ns	Master
DI hold time	-	10	-	-	ns	Slave
	-	90	-	-	ns	Master
CK low to DO valid time	-	-	-	30	ns	Slave
	-	-	-	120	ns	Master
CN setup time	-	60	-	-	ns	Master/Slave
CN high to DI tri-state[a]	-	-	-	-	ns	Master

a. Master actively stops reading during transmission, and Slave releases its driver DO and turns to tri-state.

13.6 I2C Characteristics

Over process, voltage 1.9 ~ 3.6 V, T = -40 ~ +85°C unless otherwise stated.

Table 13-11 I2C Characteristics

Item	Sym.	Standard Mode		Fast Mode		Unit	Conditions
		Min	Max	Min	Max		
SCL frequency	F_{SCL}	-	100	-	400	kHz	-
Rise time of SDA and SCL signals	T_R	-	1000	-	300	ns	-
Fall time of SDA and SCL signals	T_F	-	300	-	300	ns	-
START condition hold time	$T_{HD;STA}$	4	-	0.6	-	μs	-
Data hold time	$T_{HD;DAT}$	0	3.45	-	0.9	μs	-
Data setup time	$T_{SU;DAT}$	250	-	100	-	ns	-
STOP condition setup time	$T_{SU;STO}$	4	-	0.6	-	μs	-

13.7 Storage Condition

The TLSR8208 series is applicable to Moisture Sensitivity Level 3 (based on JEDEC Standard).

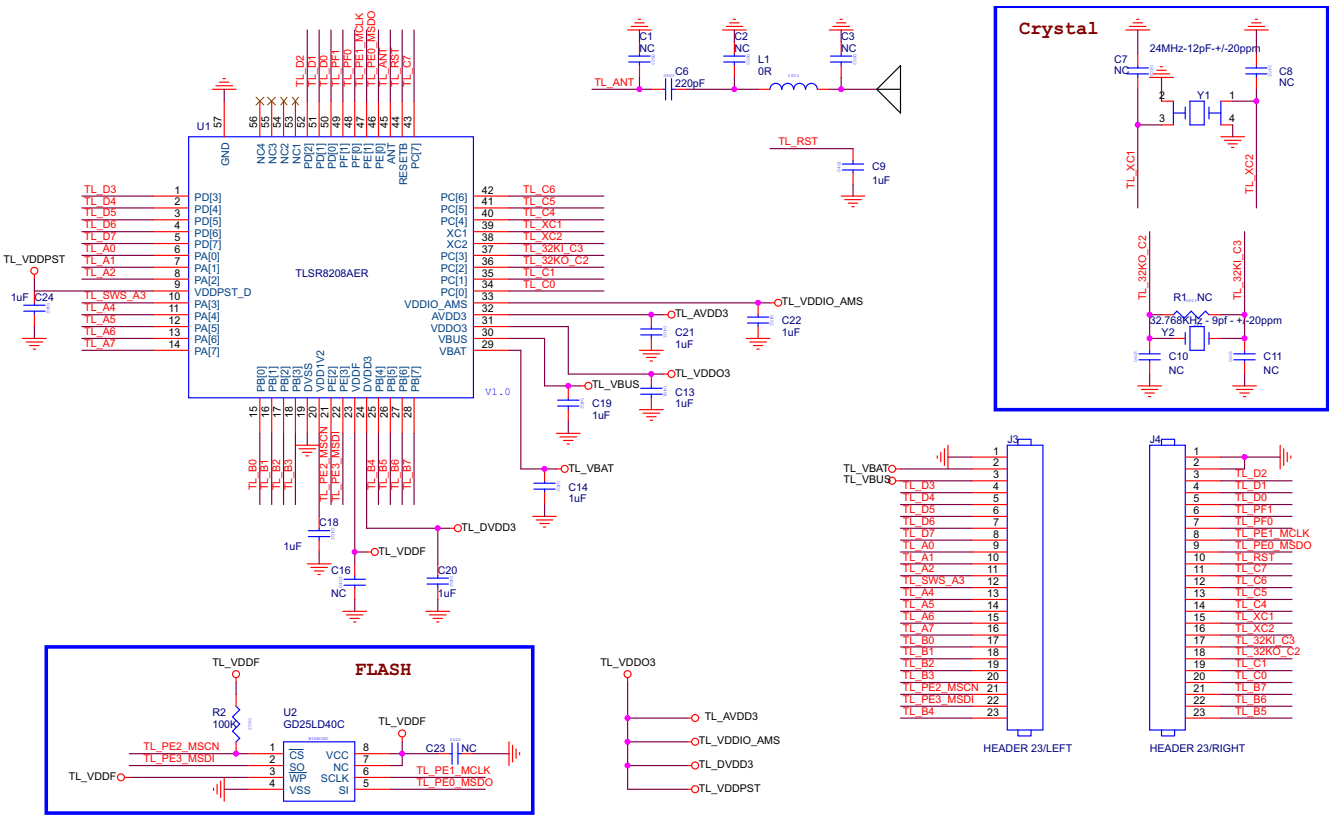
1. Calculated shelf life in sealed moisture barrier bag (MBB): 12 months at <40°C and <90% relative humidity (RH)
2. Peak package body temperature: 260°C
3. After bag is opened, devices that will be subjected to reflow solder or other high temperature process must be
   - Mounted within: 168 hours of factory conditions ≤30°C/60% RH, or
   - Stored at <10% RH
4. Devices require bake, before mounting, if:
   - Humidity Indicator Card reads >10% when read at 23 ± 5°C
   - Both of the conditions in item 3 are not met
5. If baking is required, devices may be baked for 24 hours at 125 ± 5°C

Note: If device containers cannot be subjected to high temperature or shorter bake times are desired, please refer to IPC/JEDEC J-STD-033 for bake condition.

14 Reference Design

14.1 Schematic of TLSR8208A

Figure 14-1 Schematic of TLSR8208A



14.2 BOM (Bill of Material) of TLSR8208A

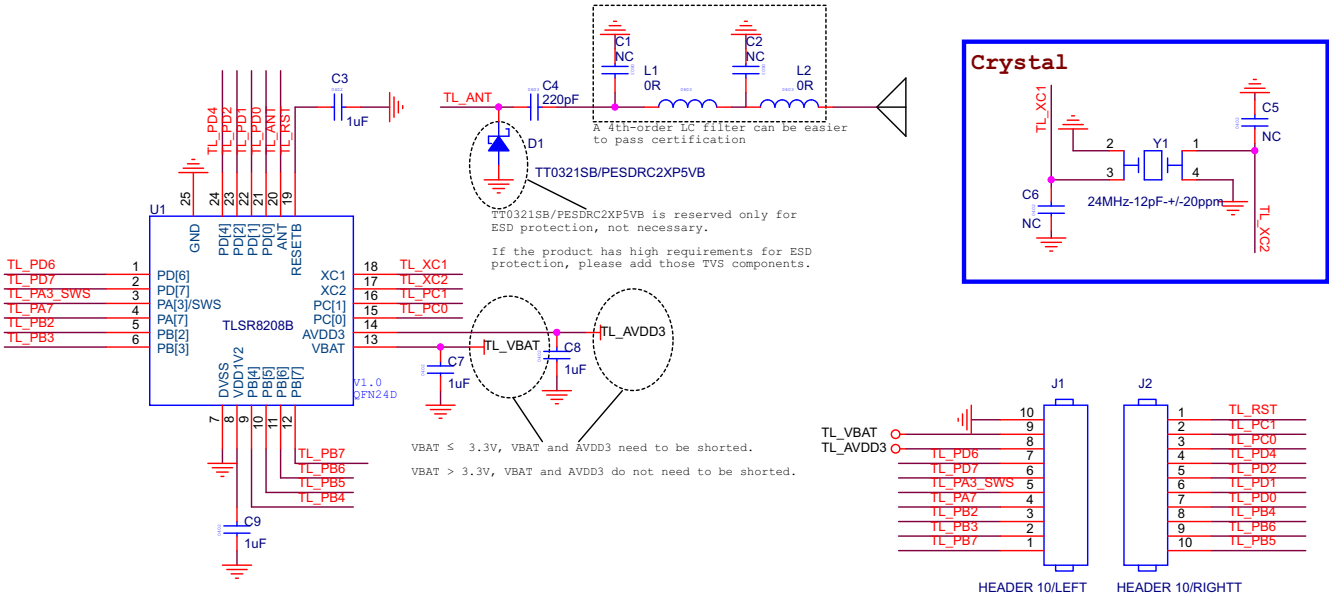
Table 14-1 BOM Table of TLSR8208A

Quantity	Reference	Value	Description	PCB Footprint
1	C6	220pF	Capacitance, X7R, ±10%	0402
9	C1, C7, C8, C9, C12, C13, C14, C15, C17	1uF	Capacitance, X5R, ±10%	0402
4	D1, D2, D3, D4	TT0321SB/ PESDRC2XP5VB	TVS	DFN1006-2L
1	J1	HEADER 22/LEFT	Pin headers	hdr254f-1x22x850
1	J2	HEADER 22/RIGHT	Pin headers	hdr254f-1x22x850
2	L1, L2	0R	Resistance, 5%	0402
1	R2	100K	Resistance, ±5%	0603R

Quantity	Reference	Value	Description	PCB Footprint
1	U1	TLSR8208A	BLE + 2.4G	qfn_7x7_56pin_0p4_4p10x4p10
1	U2	GD25LD40C	Flash	W25X10_20_40BL
1	Y1	24MHz-12pF-+/-20ppm	XTAL SMD 3225, 24 MHz, CI=12pF, total tol.±20ppm	OSCCC250X320X110
1	Y2	32.768KHz - 9pf - +/-20ppm	XTAL RADIAL 2x6mm, 32.768KHz, CI=9pF, total tol.±20ppm	OSC_2x6

14.3 Schematic of TLSR8208B

Figure 14-2 Schematic of TLSR8208B



14.4 BOM (Bill of Material) of TLSR8208B

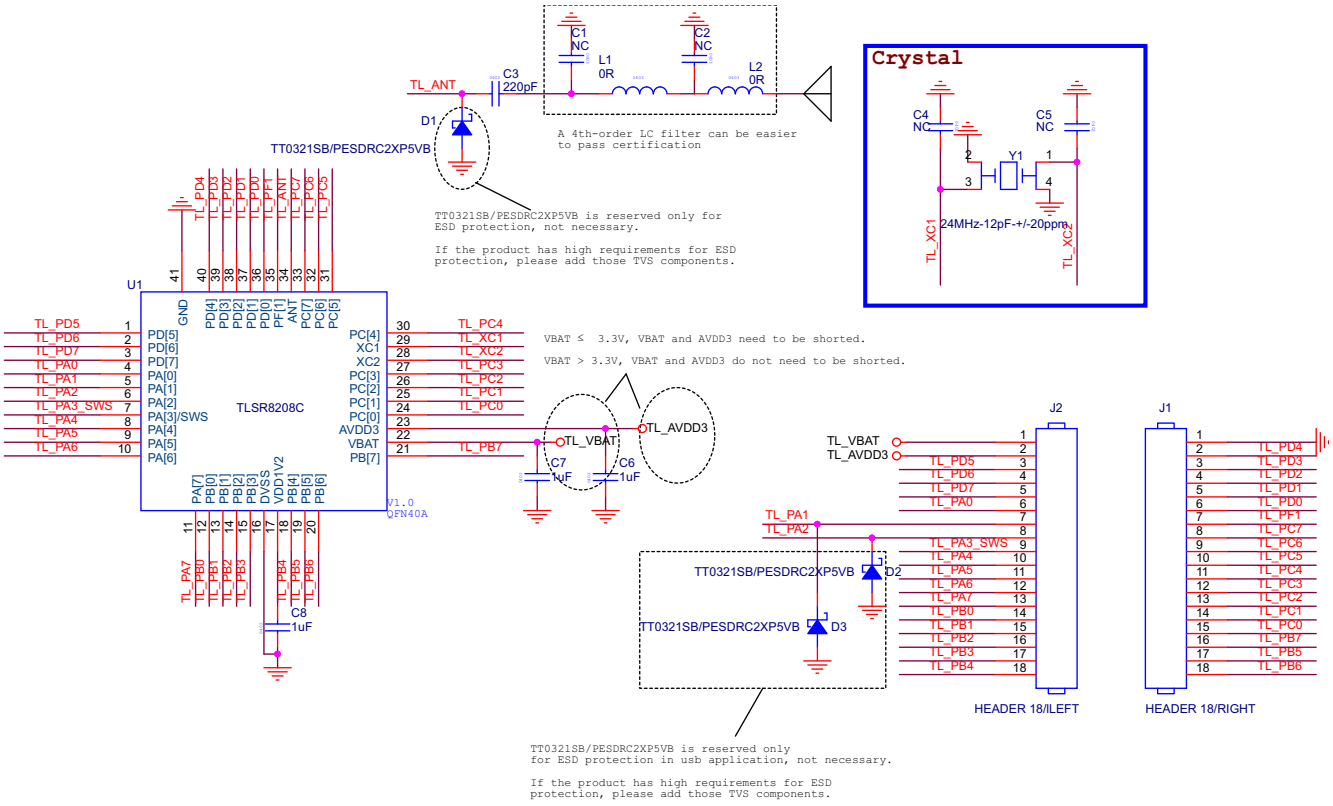
Table 14-2 BOM Table of TLSR8208B

Quantity	Reference	Value	Description	PCB Footprint
4	C3, C7, C8, C9	1uF	Capacitance, X5R, ±10%	0402
1	C4	220pF	Capacitance, X7R, ±10%	0402
1	D1	TT0321SB/ PESDRC2XP5VB	TVS	DFN1006-2L
1	J1	HEADER 10/LEFT	Pin headers	hdr254f-1x10x850

Quantity	Reference	Value	Description	PCB Footprint
1	J2	HEADER 10/RIGHTT	Pin headers	hdr254f-1x10x850
2	L1, L2	0R	Resistance, 5%	0402
1	U1	TLSR8208B	BLE + 2.4G	qfn_4x4_24pin_0p5 _2p00x2p00
1	Y1	24MHz-12pF-+/-20ppm	XTAL SMD 3225, 24 MHz, CI=12pF, total tol.±20ppm	osccc200x250x080

14.5 Schematic of TLSR8208C

Figure 14-3 Schematic of TLSR8208C



14.6 BOM (Bill of Material) of TLSR8208C

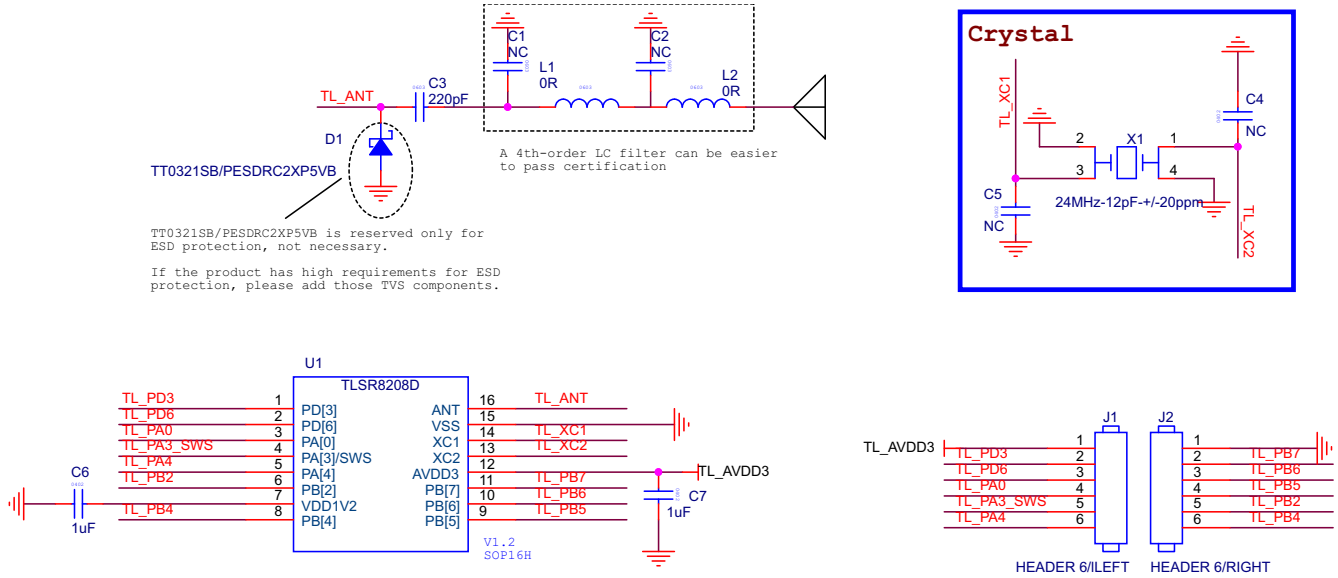
Table 14-3 BOM Table of TLSR8208C

Quantity	Reference	Value	Description	PCB Footprint
1	C3	220pF	Capacitance, X7R, ±10%	0402
3	C6, C7, C8	1uF	Capacitance, X5R, ±10%	0402

Quantity	Reference	Value	Description	PCB Footprint
3	D1, D2, D3	TT0321SB/ PESDRC2XP5VB	TVS	DFN1006-2L
1	J1	HEADER 18/LEFT	Pin headers	hdr254f-1x18x850
1	J2	HEADER 18/RIGHT	Pin headers	hdr254f-1x18x850
2	L1, L2	0R	Resistance, 5%	0402
1	U1	TLSR8208C	BLE + 2.4G	qfn_5x5_40pin_0p4 _2p60x2p60
1	Y1	24MHz-12pF-+/- 20ppm	XTAL SMD 3225, 24 MHz, CI=12pF, total tol.±20ppm	OSCCC250X320X110

14.7 Schematic of TLSR8208D

Figure 14-4 Schematic of TLSR8208D



14.8 BOM (Bill of Material) of TLSR8208D

Table 14-4 BOM Table of TLSR8208D

Quantity	Reference	Value	Description	PCB Footprint
1	C3	220pF	Capacitance, X7R, ±10%	0402
2	C6, C7	1uF	Capacitance, X5R, ±10%	0402
1	D1	TT0321SB/ PESDRC2XP5VB	TVS	DFN1006-2L

Quantity	Reference	Value	Description	PCB Footprint
1	J1	HEADER 6/LEFT	Pin headers	hdr254f-1x6x850
1	J2	HEADER 6/RIGHT	Pin headers	hdr254f-1x6x850
2	L1, L2	0R	Resistance, 5%	0402
1	U1	TLSR8208D	BLE + 2.4G	sop_16pin_4x10_1p27
1	X1	24MHz-12pF-+/-20ppm	XTAL SMD 3225, 24 MHz, CI=12pF, total tol.±20ppm	osccc200x250x080