Datasheet for Telink

Bluetooth LE SoC

TLSR8270

DS-TLSR8270-E2

Ver 1.0.1

2022/07/04

Keyword

Bluetooth LE; 2.4 GHz; PWM; Audio; QDEC; MDEC; ADC; Temperature sensor; Low power comparator; AES; PKE; TRNG; PTA

Brief

This datasheet is dedicated for Telink Bluetooth LE SoC TLSR8270.

In this datasheet, key features, working modes, main modules, electrical specifications and application of the TLSR8270 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
1.0.0	Initial release
1.0.1	• Section 1.2.1 General Features: Updated the descriptions of clock sources and AES • Section 1.2.3 Features of Power Management Module: Removed the voltage description of deep sleep • Section 2.1 Memory: Removed E-Fuse definition • Chapter 14 Low Power Comparator: Removed the input channel AVDD3, and updated Figure 14-1

Table of Contents

1 Overview 14
1.1 Block Diagram 14
1.2 Key Features 16
1.2.1 General Features 16
1.2.2 RF Features 17
1.2.3 Features of Power Management Module 17
1.2.4 USB Features 18
1.2.5 Flash Features 18
1.2.6 BLE Features 18
1.2.7 Concurrent Mode Feature 18
1.3 Typical Applications 19
1.4 Ordering Information 19
1.5 Package 20
1.6 Pin Layout 21
1.6.1 Pin Layout for TLSR8270F512ET48 21
2 Memory and MCU 31
2.1 Memory 31
2.1.1 SRAM/Register 31
2.1.2 Flash 33
2.1.3 Unique ID 33
2.2 MCU 33
2.3 Working Modes 33
2.4 Reset 35
2.5 Power Management 37
2.5.1 Power-On-Reset (POR) and Brown-Out Detect 37
2.5.2 Working Mode Switch 39
2.5.3 LDO and DCDC 40
2.5.4 VBAT and VANT Power-Supply Mode 40
2.6 Wakeup Sources 41
2.6.1 Wakeup Source - USB 41
2.6.2 Wakeup Source - 32 kHz Timer 41
2.6.3 Wakeup Source - Low Power Comparator 41
2.6.4 Wakeup Source - IO 41
2.6.5 Wakeup Source - MDEC 42
2.6.6 Register Table 42
3 BLE/2.4 GHz RF Transceiver 44

3.1 Block Diagram 44
3.2 Air Interface Data Rate and RF Channel Frequency 45
3.3 Baseband 45
3.3.1 Packet Format 45
3.3.2 BLE Location Function 46
3.3.3 RSSI and Frequency Offset 46
4 Clock 47
4.1 Clock Sources 47
4.2 System Clock 48
4.3 Module Clock 48
4.3.1 System Timer Clock 48
4.3.2 USB Clock 48
4.3.3 I2S Clock 48
4.3.4 CODEC Clock 48
4.4 Register Table 49
5 Timers 51
5.1 Timer0 ~ Timer2 51
5.1.1 Register Table 51
5.1.2 Mode 0 (System Clock Mode) 52
5.1.3 Mode 1 (GPIO Trigger Mode) 53
5.1.4 Mode 2 (GPIO Pulse Width Mode) 54
5.1.5 Mode 3 (Tick Mode) 54
5.1.6 Watchdog Timer 55
5.2 32K LTIMER 55
5.3 System Timer 55
6 Interrupt System 59
6.1 Interrupt Structure 59
6.2 Register Configuration 59
6.2.1 Enable/Mask Interrupt Sources 60
6.2.2 Interrupt Mode and Priority 61
6.2.3 Interrupt Source Flag 61
7 Interface 63
7.1 GPIO 63
7.1.1 Basic Configuration 63
7.1.1.1 GPIO Lookup Table 63
7.1.1.2 Multiplexed Functions 66
7.1.1.3 Drive Strength 67
7.1.2 Connection Relationship Between GPIO and Related Modules 68

7.1.3 Pull-Up/Pull-Down Resistor 71
7.2 SWM and SWS 74
7.2.1 Swire Through USB 74
7.3 I2C 75
7.3.1 Communication Protocol 75
7.3.2 Register Table 76
7.3.3 I2C Slave Mode 77
7.3.3.1 DMA Mode 77
7.3.3.2 Mapping Mode 78
7.3.4 I2C Master Mode 79
7.3.4.1 I2C Master Write Transfer 79
7.3.4.2 I2C Master Read Transfer 79
7.3.5 I2C and SPI Usage 79
7.4 SPI 80
7.4.1 Register Table 80
7.4.2 SPI Master Mode 80
7.4.3 SPI Slave Mode 81
7.4.4 I2C and SPI Usage 82
7.5 UART 82
7.6 USB 85
8 PWM 87
8.1 Register Table 87
8.2 Enable PWM 91
8.3 Set PWM Clock 91
8.4 PWM Waveform, Polarity and Output Inversion 91
8.4.1 Waveform of Signal Frame 91
8.4.2 Invert PWM Output 92
8.4.3 Polarity for Signal Frame 92
8.5 PWM Modes 92
8.5.1 Select PWM Modes 92
8.5.2 Continuous Mode 92
8.5.3 Counting Mode 93
8.5.4 IR Mode 93
8.5.5 IR FIFO Mode 94
8.5.6 IR DMA FIFO Mode 95
8.6 PWM Interrupt 98
9 Audio 100
9.1 Audio Input Path 100

9.1.1 Audio Input Process 100
9.1.1.1 CODEC 100
9.1.1.2 ALC 101
9.1.2 Audio Input Path 103
9.1.2.1 AMIC Input 104
9.1.2.2 DMIC Input 105
9.1.2.3 I2S Input 105
9.1.2.4 USB Host Input 105
9.1.2.5 DFIFO 105
9.1.3 Register Configuration 106
9.2 Audio Output Path 112
9.2.1 Rate Matching 113
9.2.2 SDM 113
9.2.3 Register Configuration 114
10 Quadrature Decoder 117
10.1 Input Pin Selection 117
10.2 Common Mode and Double Accuracy Mode 117
10.3 Read Real Time Counting Value 119
10.4 QDEC Reset 120
10.5 Other Configuration 120
10.6 Timing Sequence 121
10.7 Register Table 122
11 Manchester Decoder 123
11.1 Frame Format 123
11.2 Function Description 123
11.2.1 Block Diagram 123
11.2.2 Reset MDEC 124
11.2.3 Select Input Channel 124
11.2.4 Read Result Data 124
11.3 Register Table 125
12 SAR ADC 126
12.1 Power On/Down 126
12.2 ADC Clock 126
12.3 ADC Control in Auto Mode 126
12.3.1 Set Max State and Enable Channel 126
12.3.2 "Set" State 127
12.3.3 "Capture" State 127
12.3.4 Usage Case with Detailed Register Setting 128

12.4 Register Table 129
13 Temperature Sensor 133
14 Low Power Comparator 134
14.1 Power On/Down 134
14.2 Select Input Channel 134
14.3 Select Mode and Input Channel for Reference 135
14.4 Select Scaling Coefficient 135
14.5 Low Power Comparator Output 135
14.6 Register Table 135
15 AES 137
15.1 RISC Mode 137
15.2 DMA Mode 137
15.3 AES-CCM 137
15.4 Register Table 138
16 Public Key Engine 139
16.1 Calculation Model Overview 139
16.2 Function Description 139
16.2.1 Module Description 139
16.2.2 Software Interface (Programming Model) 140
16.3 Register Description 142
17 True Random Number Generator (TRNG) 148
17.1 Model Overview 148
17.2 Register Description 148
17.3 Interrupt Description 152
17.3.1 CPU Reads RBG_DR Without Data 152
17.3.2 Data Valid 152
17.4 Usage Procedure 153
17.4.1 Normal Operation 153
17.4.2 Entropy Source 153
18 PTA Interface 154
18.1 Two-Wire Signaling 154
18.2 Three-Wire or Four-Wire Signaling 155
19 Key Electrical Specifications 157
19.1 Absolute Maximum Ratings 157
19.2 Recommended Operating Conditions 157
19.3 DC Characteristics 157
19.4 AC Characteristics 158
19.5 SPI Characteristics 163

19.6 I2C Characteristics 164
19.7 Flash Characteristics 164
19.8 ESD Characteristics 165
19.9 Storage Condition 166
20 Reference Design 167
20.1 Schematic of TLSR8270F512ET48 167
20.2 BOM (Bill of Material) of TLSR8270F512ET48 167

List of Figures

Figure 1-1 Block Diagram of the System 15
Figure 1-2 Package of TLSR8270F512ET48 20
Figure 1-3 Pin Assignments for TLSR8270F512ET48 22
Figure 2-1 Physical Memory Map 31
Figure 2-2 Register Space 32
Figure 2-3 Control Logic for Power Up/Down 37
Figure 2-4 Initial Power-Up Sequence 38
Figure 2-5 Power-Down Sequence 39
Figure 2-6 Wakeup Sources 41
Figure 3-1 Block Diagram of RF Transceiver 44
Figure 4-1 Block Diagram of Clock 47
Figure 7-1 Logic Relationship Between GPIO and Related Modules 68
Figure 7-2 Swire Through USB Diagram 74
Figure 7-3 Timing Sequence of Enabling Swire Through USB 75
Figure 7-4 I2C Timing Chart 75
Figure 7-5 Byte Consisted of Slave Address and R/W Flag Bit 77
Figure 7-6 Read Format in DMA Mode 78
Figure 7-7 Write Format in DMA Mode 78
Figure 7-8 Read Format in Mapping Mode 78
Figure 7-9 Write Format in Mapping Mode 78
Figure 7-10 SPI Write/Read Command Format 82
Figure 7-11 UART Communication 83
Figure 8-1 A Signal Frame 91
Figure 8-2 PWM Output Waveform Chart 92
Figure 8-3 Continuous Mode 93
Figure 8-4 Counting Mode (n=0) 93
Figure 8-5 IR Mode (n=0) 94

Figure 8-6 IR Format Examples 95
Figure 9-1 Audio Input Process 100
Figure 9-2 Average Filter Envelope Detect 102
Figure 9-3 ATK/DCY Processing 102
Figure 9-4 Audio Input Path 104
Figure 9-5 Audio Output Path 112
Figure 9-6 Linear Interpolation 113
Figure 9-7 Delay Interpolation 113
Figure 9-8 Block Diagram of SDM 114
Figure 10-1 Common Mode 118
Figure 10-2 Double Accuracy Mode 119
Figure 10-3 Read Real Time Counting Value 120
Figure 10-4 Shuttle Mode 120
Figure 10-5 Timing Sequence Chart 121
Figure 11-1 Frame Format 123
Figure 11-2 Function Block Diagram 124
Figure 12-1 Block Diagram of ADC 126
Figure 13-1 Block Diagram of Temperature Sensor 133
Figure 14-1 Block Diagram of Low Power Comparator 134
Figure 16-1 Block Diagram of PKE SP Module 140
Figure 17-1 Module Boundary 148
Figure 18-1 Two-Wire Signaling 154
Figure 18-2 Example of Two-Wire PTA Timing Diagram 154
Figure 18-3 Three-Wire or Four-Wire Signaling 155
Figure 18-4 Example of Four-Wire PTA Timing Diagram 156
Figure 20-1 Schematic of TLSR8270F512ET48 167

List of Tables

Table 1-1 Ordering Information of TLSR8270 19
Table 1-2 Mechanical Dimensions of TLSR8270F512ET48 21
Table 1-3 Pin Function of TLSR8270F512ET48 22
Table 1-4 GPIO Pin Mux of TLSR8270F512ET48TLSR8277F512EL40 24
Table 1-5 PWM Signal Description 26
Table 1-6 I2C Signal Description 26
Table 1-7 I2S Signal Description 26
Table 1-8 UART Signal Description 26
Table 1-9 Audio Output Signal Description 27
Table 1-10 SPI Signal Description 27
Table 1-11 7816 Signal Description 27
Table 1-12 DMIC Signal Description 27
Table 1-13 Swire Signal Description 28
Table 1-14 AOA/AOD Signal Description 28
Table 1-15 External Power Amplifier, Low Noise Amplifier Signal Description 28
Table 1-16 USB Signal Description 28
Table 1-17 DECODEC Signal Description 28
Table 1-18 Audio_in Signal Description 28
Table 1-19 Low Current Comparator Signal Description 28
Table 1-20 SAR ADC Signal Description 29
Table 1-21 Strong Pull Up Signal Description 29
Table 1-22 Crystal Signal Description 30
Table 2-1 Working Modes 33
Table 2-2 Retention Analog Registers in Deep Sleep 35
Table 2-3 Register Configuration for Software Reset 36
Table 2-4 Analog Register to Control Delay Counters 37
Table 2-5 Characteristics of Initial Power-Up/Power-Down Sequence 39

Table 2-6 Analog Registers for Wakeup 42
Table 2-7 Digital Register for Wakeup 43
Table 3-1 External RF Transceiver Control Example 45
Table 3-2 Packet Format in Standard 1 Mbps BLE Modea 45
Table 3-3 Packet Format in Standard 2 Mbps BLE Mode 45
Table 3-4 Packet Format in Standard 500 kbps/125 kbps BLE Mode 46
Table 3-5 Packet Format in Proprietary Mode 46
Table 4-1 Clock Register Table 49
Table 5-1 Register Configuration for Timer0 ~ Timer2 51
Table 5-2 Register Table for System Timer 56
Table 6-1 Register Table for Interrupt System 59
Table 7-1 GPIO PAD Function Mux 63
Table 7-2 GPIO Setting 64
Table 7-3 Select Multiplexed SPI/I2C 67
Table 7-4 GPIO IRQ Table 69
Table 7-5 Analog Registers for Pull-Up/Pull-Down Resistor Control 71
Table 7-6 Register Configuration for I2C 76
Table 7-7 Register Configuration for SPI 80
Table 7-8 SPI Master Mode 81
Table 7-9 SPI Slave Mode 81
Table 7-10 Register Configuration for UART 83
Table 8-1 Register Table for PWM 87
Table 9-1 CODEC Frequency Table 100
Table 9-2 Audio Data Flow Direction 103
Table 9-3 Audio Input Registers 106
Table 9-4 Register Configuration Related to Audio Output Path 115
Table 10-1 Input Pin Selection 117
Table 10-2 Timing 121
Table 10-3 Register Table for QDEC 122

Table 11-1 Analog Registers for MDEC 125
Table 12-1 Overall Register Setting 128
Table 12-2 Register Table Related to SAR ADC 129
Table 13-1 Analog Register for Temperature Sensor 133
Table 14-1 Analog Register Table Related to Low Power Comparator 135
Table 15-1 Register Table Related to AES 138
Table 16-1 Dual Port RAM Address Map 141
Table 16-2 Register Map 142
Table 17-1 Register Map 148
Table 18-1 Register Configuration for t1/t2 156
Table 19-1 Absolute Maximum Ratings 157
Table 19-2 Recommended Operating Conditions 157
Table 19-3 DC Characteristics 158
Table 19-4 Digital Inputs/Outputs Characteristics 158
Table 19-5 RF Performance Characteristics 159
Table 19-6 USB Characteristics 162
Table 19-7 RSSI Characteristics 162
Table 19-8 Crystal Characteristics 162
Table 19-9 RC Oscillator Characteristics 162
Table 19-10 ADC Characteristics 163
Table 19-11 SPI Characteristics 163
Table 19-12 I2C Characteristics 164
Table 19-13 Flash Memory Characteristics 164
Table 19-14 HBM/CDM Results 165
Table 19-15 Latch-Up I-Test Result 165
Table 19-16 Latch-Up Vsupply Over Voltage Test Result 166
Table 20-1 BOM Table of TLSR8270F512ET48 167

1 Overview

The TLSR8270 is a Telink-developed Bluetooth LE SoC solution with internal Flash and audio support, which combines the features and functions needed for all 2.4 GHz IoT standards into a single SoC. It's completely RoHS-compliant and 100% lead (Pb)-free.

The TLSR8270 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 (up to Bluetooth 5.1), and 2.4 GHz proprietary standard. The TLSR8270's embedded FLASH enables dynamic stack and profile configuration, and the final end product functionality is configurable via software, providing ultimate flexibility. The TLSR8270 also has hardware OTA upgrades support and multiple boot switching, allowing convenient product feature roll outs and upgrades.

The TLSR8270 supports concurrent multi-standards. For some use cases, the TLSR8270 can "concurrently" run two standards, for example, stacks such as BLE 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 BLE 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 BLE specification up to Bluetooth 5.1, allows easy connectivity with Bluetooth Smart Ready mobile phones, tablets, laptops, which supports BLE 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 TLSR8270 integrates hardware acceleration to support the complicated security operations required by Bluetooth, up to and including 5.1 standard, without the requirement for an external DSP, thereby significantly reducing the product eBOM.

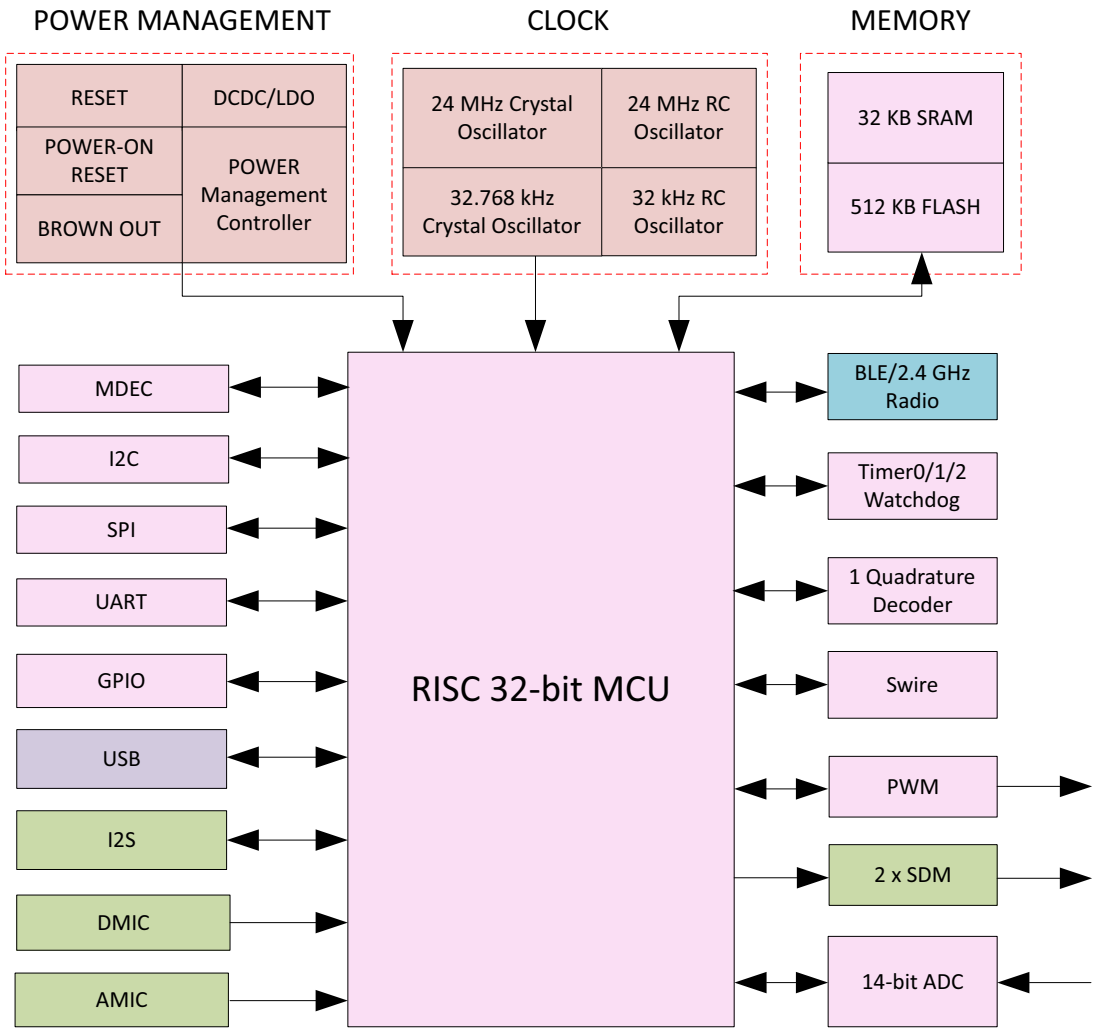
The TLSR8270 supports single-channel analog microphone, dual-channel digital microphone, and stereo audio output with enhanced voice performance for voice search and other such applications. The TLSR8270 also includes a full range of on-chip peripherals for interfacing with external components such as LEDs, sensors, keyboards, and motors. This makes it an ideal single-chip solution for IoT (Internet of Things) and HID (Human Interface Devices) applications such as wearable devices, smart lighting, smart home devices, advanced remote controls, and wireless toys.

The TLSR8270 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 TLSR8270 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



NOTE:

- Modules marked with different colors belong to different power domains. Power state of each power domain can be controlled independent of other power domains, for example, the audio module (including I2S, DMIC, AMIC, SDM) can be independently powered on or powered down irrespective of other modules such as power management module, clock, and etc.
- The 2.4 GHz Radio, USB and Audio (I2S, DMIC, AMIC, SDM) are powered down by default.
- The power management module and clock should be always powered on, even in deep sleep.
- In deep sleep, except for the power management and clock, all other modules should be powered down.

The TLSR8270 integrates a power-balanced 32-bit MCU, BLE/2.4 GHz Radio, 32 KB (16K+16K) SRAM, 512 KB internal Flash, 14-bit ADC, single-channel analog microphone input, dual-channel digital microphone input, stereo audio output, 6-channel PWM (1-channel IR/IR FIFO/IR DMA FIFO), one quadrature decoder (QDEC), abundant and flexible GPIO interfaces, and nearly all the peripherals needed for IoT (Internet of Things) and HID (Human Interface Devices) application development (e.g. Bluetooth Low Energy). The TLSR8270 also includes multi-stage power management design allowing ultra-low power operation and making it the ideal candidate for wearable and power-constraint applications.

With the high integration level of the TLSR8270, 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. Support Unique ID (UID)
2. Embedded 32-bit proprietary microcontroller
   - Better power-balanced performance than ARM M0
   - Instruction cache controller
   - Maximum running speed up to 48 MHz
3. Program memory: Internal 512 KB flash
4. Data memory: 32 KB on-chip SRAM
   - Up to 32 KB SRAM with retention in deep sleep
5. 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
6. A rich set of I/Os:
   - Up to 32 GPIOs. All digital IOs can be used as GPIOS.
   - Dual-channel DMIC (Digital Mic)
   - Single-channel AMIC (Analog Mic)
   - I2S
   - Stereo audio output
   - SPI
   - I2C
   - UART with hardware flow control and 7816 protocol support
   - USB
   - Swire debug interface
   - Manchester decoder interface selectable as wakeup source
7. Up to 6 channels of differential PWM:
   - PWM1 ~ PWM5: 5-channel normal PWM output
   - PWM0: 1 channel with normal mode as well as additional IR/IR FIFO/IR DMA FIFO mode for IR generation
8. Sensor:
   - 14-bit 10-channel (only GPIO input) SAR ADC

- Temperature sensor

9. One quadrature decoder
10. Embedded hardware AES block cipher with 128 bit keys and software AES CCM
11. Embedded hardware acceleration for Elliptical Curve Cryptography (ECC) supports Bluetooth standard up to and including BLE 5.1
12. Embedded low power comparator
13. Embedded TRNG (True Random Number Generator) compliant with NIST SP800-22
14. Operating temperature range: -40°C ~ +85°C
15. Support 2.4 GHz IoT standards into a single SoC, including BLE and 2.4 GHz proprietary technologies

1.2.2 RF Features

RF features include:

1. BLE/2.4 GHz RF transceiver embedded, working in worldwide 2.4 GHz ISM band
2. Bluetooth 5.1 compliant, 1 Mbps, 2 Mbps, Long Range 125 kbps and 500 kbps
3. 2.4 GHz proprietary 1 Mbps/2 Mbps/250 kbps/500 kbps mode
   - Support Adaptive Frequency Hopping feature
   - Support flexible GFSK/FSK modulation index configuration
   - Support 1-N receiver capability
4. RX sensitivity: -96 dBm @ BLE 1 Mbps mode, -93 dBm @ BLE 2 Mbps mode, -100 dBm @ BLE 125 kbps mode, -98 dBm @ BLE 500 kbps mode
5. TX output power: -45 to +10 dBm
6. Single-pin antenna interface
7. RSSI monitoring with +/-1 dB resolution
8. Auto acknowledgement, retransmission and flow control
9. Support single-antenna AOA/TX
10. Integrated load inductor
11. PTA interface with 2/3/4-wire support

1.2.3 Features of Power Management Module

Features of power management module include:

1. Embedded LDO and DCDC
   - DCDC for 1.8 V flash with bypass LDO
   - DCDC for chip with bypass LDO
   - USB LDO with power supply of 4.5 V ~ 5.5 V
2. Battery monitor: Support low battery detection
3. Power supply:
   - VDD: 1.8 V ~ 3.6 V
   - VBUS (USB): 4.5 V ~ 5.5 V
4. Multiple stage power management to minimize power consumption
5. Low power consumption:

- Whole chip RX mode: 4.6 mA with DCDC, 9.1 mA with LDO
- Whole chip TX mode @ 0 dBm: 4.9 mA with DCDC, 9.5 mA with LDO
- Deep sleep with external wakeup (without SRAM retention): 0.4 μA
- Deep sleep with SRAM retention: 0.8 μA (with 16 KB SRAM retention), 1.0 μA (with 32 KB SRAM retention)
- Deep sleep with external wakeup, with 32K RC oscillator on (without SRAM retention): 0.8 μA
- Deep sleep with SRAM retention, with 32K RC oscillator on: 1.3 μA (with 16 KB SRAM retention), 1.5 μA (with 32 KB SRAM retention)

1.2.4 USB Features

USB features include:

1. Compatible with USB 2.0 full speed mode
2. Support 9 endpoints including control endpoint 0 and 8 configurable data endpoints
3. Independent power domain
4. Support ISP (In-System Programming) via USB port

1.2.5 Flash Features

The TLSR8270 embeds flash with features below:

1. Total 512 KB (4 Mbits)
2. Flexible architecture: 4 KB per sector, 64 KB/32 KB per block
3. Up to 256 bytes per programmable page
4. Write protect all or portions of memory
5. Sector erase (4 KB)
6. Block erase (32 KB/64 KB)
7. Cycle endurance: 100,000 program/erases
8. Data retention: Typical 20-year retention

1.2.6 BLE Features

1. Fully compliant with Bluetooth 5.1
2. Telink proprietary Mesh support
3. Telink extended profile with audio support for voice command based searches

1.2.7 Concurrent Mode Feature

In concurrent mode, the chip supports multiple standard working concurrently.

Typical combination is BLE and 2.4G based stacks can run concurrently with one application state but dual radio communication channels for interacting with different devices.

1.3 Typical Applications

The TLSR8270 can be applied to IoT (Internet of Things) and HID (Human Interface Devices) applications, such as BLE smart devices, home automation devices. Its typical applications include, but are not limited to the following:

- Smartphone and tablet accessories
- RF and IR remote control
- Sports and fitness tracking
- Wearable devices
- Wireless toys
- Building automation
- Intelligent logistics/transportation/city
- Industrial control
- Health care

1.4 Ordering Information

Table 1-1 Ordering Information of TLSR8270

Product Series	Package Type	Temperature Range	Ordering No.	Packing Method	Minimum Order Quantity
TLSR8270F512	48-pin TQFN 7x7x0.75 mm	-40°C ~ +85°C	TLSR8270F512ET48	TR[a]	3000

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

1.5 Package

Package dimensions of TLSR8270F512ET48 are shown below.

Figure 1-2 Package of TLSR8270F512ET48

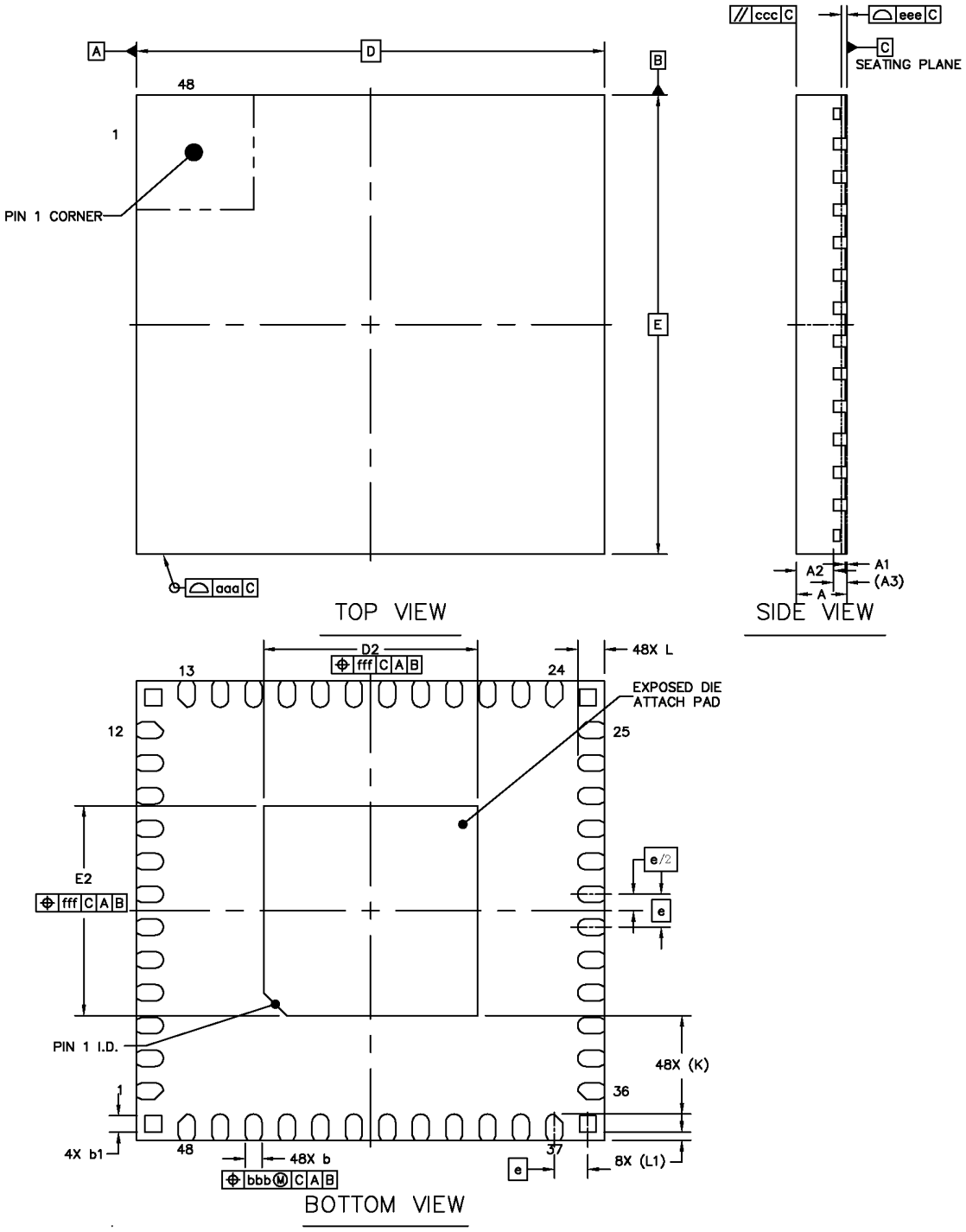


Table 1-2 Mechanical Dimensions of TLSR8270F512ET48

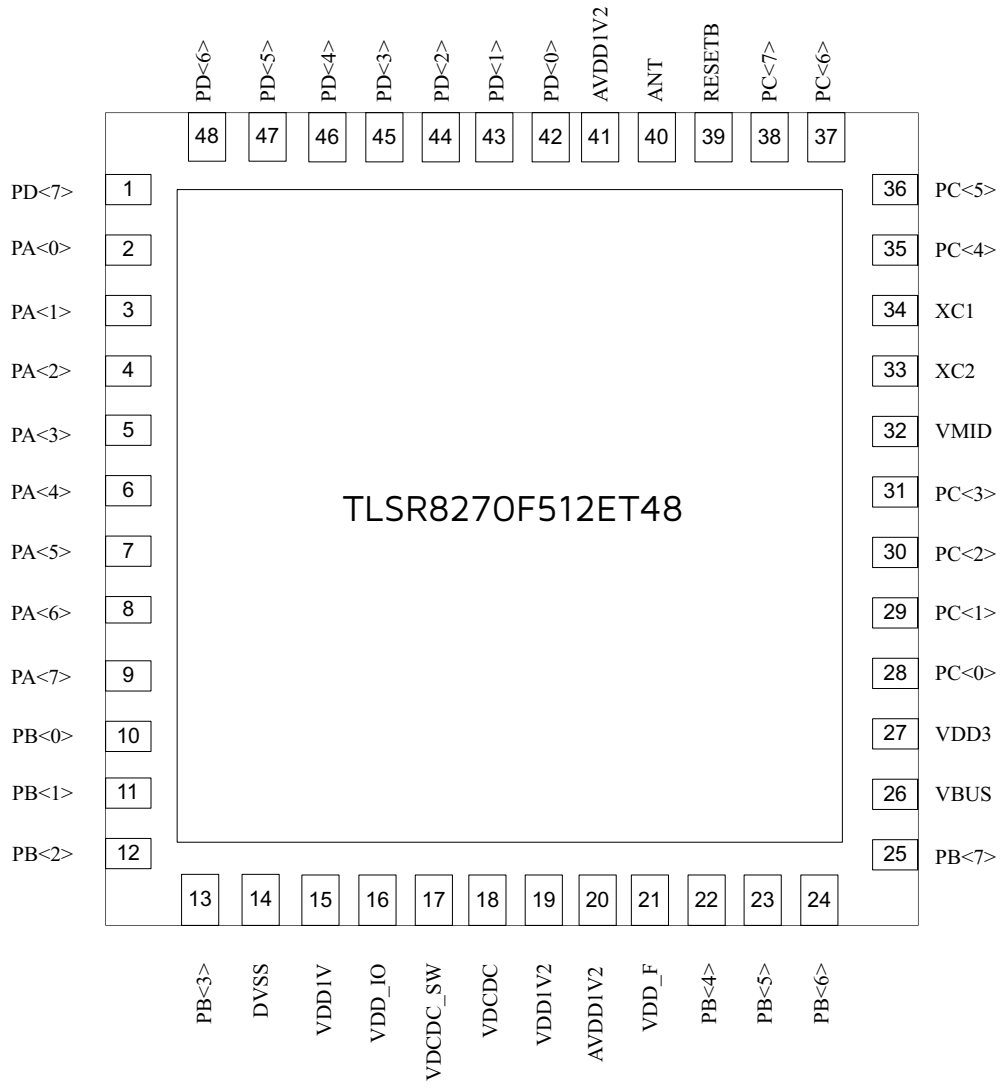
		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.02	0.05
MOLD THICKNESS		A2	---	0.55	---
L/F THICKNESS		A3		0.203 REF	
LEAD WIDTH		b	0.2	0.25	0.3
		b1	0.2	0.25	0.3
BODY SIZE	X	D		7 BSC	
	Y	E		7 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	D2	3.1	3.2	3.3
	Y	E2	3.1	3.2	3.3
LEAD LENGTH		L	0.3	0.4	0.5
LEAD TO PKG LINE		L1		0.125 REF	
LEAD TIP TO EXPOSED PAD EDGE		K		1.5 REF	
PACKAGE EDGE TOLERANCE		aaa		0.1	
MOLD FLATNESS		ccc		0.1	
COPLANARITY		eee		0.08	
LEAD OFFSET		bbb		0.1	
EXPOSED PAD OFFSET		fff		0.1	

1.6 Pin Layout

1.6.1 Pin Layout for TLSR8270F512ET48

Figure 1-3 shows pin assignments for TLSR8270F512ET48.

Figure 1-3 Pin Assignments for TLSR8270F512ET48



Functions of 48 pins for TLSR8270F512ET48 are described in Table 1-3.

Table 1-3 Pin Function of TLSR8270F512ET48

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

No.	Pin Name	Type	Description
7	PA[5]	GPIO	GPIO PA[5], refer to Table 1-4 for pin mux function.
8	PA[6]	GPIO	GPIO PA[6], refer to Table 1-4 for pin mux function.
9	PA[7]	GPIO	GPIO PA[7], refer to Table 1-4 for pin mux function.
10	PB[0]	GPIO	GPIO PB[0], refer to Table 1-4 for pin mux function.
11	PB[1]	GPIO	GPIO PB[1], refer to Table 1-4 for pin mux function.
12	PB[2]	GPIO	GPIO PB[2], refer to Table 1-4 for pin mux function.
13	PB[3]	GPIO	GPIO PB[3], refer to Table 1-4 for pin mux function.
14	DVSS	GND	Digital LDO ground
15	VDD1V	PWR	Internal LDO generated power supply input for digital core
16	VDD_IO	PWR	External 3.3 V power supply input for IO
17	VDCDC_SW	Analog	Connected with VDCDC via external inductor
18	VDCDC	Analog	Connected with VDCDC_SW via external inductor
19	VDD1V2	PWR	Internal DCDC generated power supply. Connect to GND via external capacitor. Route this 1.2 V voltage power supply to AVDD1V2.
20	AVDD1V2	PWR	Power supply input for internal RF Modules. Route from VDD1V2. Connect to GND via external capacitor.
21	VDD_F	PWR	Internally generated power supply to flash. Connect to GND via external capacitor.
22	PB[4]	GPIO	GPIO PB[4], refer to Table 1-4 for pin mux function.
23	PB[5]	GPIO	GPIO PB[5], refer to Table 1-4 for pin mux function.
24	PB[6]	GPIO	GPIO PB[6], refer to Table 1-4 for pin mux function.
25	PB[7]	GPIO	GPIO PB[7], refer to Table 1-4 for pin mux function.
26	VBUS	PWR	USB 5 V supply
27	VDD3	PWR	Connect to an external 3.3 V power supply
28	PC[0]	GPIO	GPIO PC[0], refer to Table 1-4 for pin mux function.
29	PC[1]	GPIO	GPIO PC[1], refer to Table 1-4 for pin mux function.
30	PC[2]	GPIO	GPIO PC[2], refer to Table 1-4 for pin mux function.
31	PC[3]	GPIO	GPIO PC[3], refer to Table 1-4 for pin mux function.
32	VMID	Analog	Audio pin connecting to external decap
33	XC2	Analog	Crystal oscillator pin

No.	Pin Name	Type	Description
34	XC1	Analog	Crystal oscillator pin
35	PC[4]	GPIO	GPIO PC[4], refer to Table 1-4 for pin mux function.
36	PC[5]	GPIO	GPIO PC[5], refer to Table 1-4 for pin mux function.
37	PC[6]	GPIO	GPIO PC[6], refer to Table 1-4 for pin mux function.
38	PC[7]	GPIO	GPIO PC[7], refer to Table 1-4 for pin mux function.
39	RESETB	Reset	Power on reset, active low
40	ANT	Analog	Pin to connect to the Antenna through the matching network
41	AVDD1V2	PWR	Supply for the radio IP
42	PD[0]	GPIO	GPIO PD[0], refer to Table 1-4 for pin mux function.
43	PD[1]	GPIO	GPIO PD[1], refer to Table 1-4 for pin mux function.
44	PD[2]	GPIO	GPIO PD[2], refer to Table 1-4 for pin mux function.
45	PD[3]	GPIO	GPIO PD[3], refer to Table 1-4 for pin mux function.
46	PD[4]	GPIO	GPIO PD[4], refer to Table 1-4 for pin mux function.
47	PD[5]	GPIO	GPIO PD[5], refer to Table 1-4 for pin mux function.
48	PD[6]	GPIO	GPIO PD[6], refer to Table 1-4 for pin mux function.

GPIO pin mux functions of TLSR8270F512ET48 are shown in Table 1-4.

Table 1-4 GPIO Pin Mux of TLSR8270F512ET48TLSR8277F512EL40

Pad	Default	Func1	Func2	Func3	Func4
PA[0]	GPIO	UART_RX	PWM0_N	DMIC_DI	PS_PE<0>/MDEC
PA[1]	GPIO	I2S_CLK	7816_CLK	DMIC_CLK	-
PA[2]	GPIO	PWM0	UART_TX	DO	-
PA[3]	GPIO	PWM1	UART_CTS	DI/SDA	-
PA[4]	GPIO	PWM2	UART_RTS	CK/SCL	-
PA[5]	DM	-	-	DM	-
PA[6]	DP (SWS)	-	-	DP (SWS)	-
PA[7]	SWS	-	UART_RTS	SWS	-
PB[0]	GPIO	ATSEL1	UART_RX	PWM3	lc_comp_ain<0>/sar_aio<0>
PB[1]	GPIO	ATSEL2	UART_TX	PWM4	lc_comp_ain<1>/sar_aio<1>

Pad	Default	Func1	Func2	Func3	Func4
PB[2]	GPIO	RX_CYC2LNA	UART_CTS	PWM5	lc_comp_ain<2>/sar_aio<2>
PB[3]	GPIO	TX_CYC2PA	UART_RTS	PWM0_N	lc_comp_ain<3>/sar_aio<3>
PB[4]	GPIO	-	PWM4	SDM_P0	lc_comp_ain<4>/sar_aio<4>
PB[5]	GPIO	-	PWM5	SDM_N0	lc_comp_ain<5>/sar_aio<5>
PB[6]	SPI_DI	UART_RTS	SPI_DI/SDA	SDM_P1	lc_comp_ain<6>/sar_aio<6>
PB[7]	SPI_DO	UART_RX	SPI_DO	SDM_N1	lc_comp_ain<7>/sar_aio<7>/ MDEC
PC[0]	GPIO	UART_RTS	PWM4_N	I2C_SDA	-
PC[1]	GPIO	PWM0	PWM1_N	I2C_SCK	audio_in
PC[2]	GPIO	I2C_SDA	7816_TRX/ UART_TX	PWM0	xtl_32k_out
PC[3]	GPIO	I2C_SCK	UART_RX	PWM1	xtl_32k_in
PC[4]	GPIO	PWM0	UART_CTS	PWM2	sar_aio<8>/MDEC
PC[5]	GPIO	ATSEL0	UART_RX	PWM3_N	sar_aio<9>
PC[6]	GPIO	PWM4_N	ATSEL1	RX_CYC2LNA	-
PC[7]	GPIO	PWM5_N	ATSEL2	TX_CYC2PA	-
PD[0]	GPIO	7816_TRX/ UART_TX	-	RX_CYC2LNA	PS_PE<1>/MDEC
PD[1]	GPIO	UART_CTS	-	TX_CYC2PA	PS_PE<2>
PD[2]	SPI_CN	PWM3	I2S_LR	SPI_CN	-
PD[3]	GPIO	7816_TRX/ UART_TX	I2S_SDI	PWM1_N	-
PD[4]	GPIO	PWM2_N	I2S_SDO	SWM	-
PD[5]	GPIO	PWM0_N	-	PWM0	-
PD[6]	GPIO	ATSEL0	UART_RX	CN	-
PD[7]	SPI_CK	7816_TRX/ UART_TX	I2S_BCK	SPI_CK/SCL	PS_PE<3>

Descriptions of each signal are listed in Table 1-5 to Table 1-22.

Table 1-5 PWM Signal Description

Signal	Type	Description
PWM0	DO	PWM channel 0 output
PWM0_N	DO	PWM channel 0 inversion output
PWM1	DO	PWM channel 1 output
PWM1_N	DO	PWM channel 1 inversion output
PWM2	DO	PWM channel 2 output
PWM2_N	DO	PWM channel 2 inversion output
PWM3	DO	PWM channel 3 output
PWM3_N	DO	PWM channel 3 inversion output
PWM4	DO	PWM channel 4 output
PWM4_N	DO	PWM channel 4 inversion output
PWM5	DO	PWM channel 5 output
PWM5_N	DO	PWM channel 5 inversion output

Table 1-6 I2C Signal Description

Signal	Type	Description
I2C_SCK	DIO	I2C SCL
I2C_SDA	DIO	I2C SDA

Table 1-7 I2S Signal Description

Signal	Type	Description
I2S_BCK	DO	I2S bit CLK
I2S_CLK	DO	I2S base CLK
I2S_LR	DO	I2S left and right channel SEL
I2S_SDI	DI	I2S data IN
I2S_SDO	DO	I2S data OUT

Table 1-8 UART Signal Description

Signal	Type	Description
UART_CTS	DI	UART Clear to Send signal

Signal	Type	Description
UART_RTS	DO	UART Ready to Send signal
UART_RX	DI	UART RX
UART_TX	DO	UART TX

Table 1-9 Audio Output Signal Description

Signal	Type	Description
SDM_N0	DO	SDM0 diff output
SDM_P0	DO	SDM0 diff output
SDM_N1	DO	SDM1 diff output
SDM_P1	DO	SDM1 diff output

Table 1-10 SPI Signal Description

Signal	Type	Description
SPI_CK	DIO	SPI CLK
SPI_CN	DIO	SPI CN
SPI_DI	DIO	SPI DI
SPI_DO	DIO	SPI DO

Table 1-11 7816 Signal Description

Signal	Type	Description
7816_CLK	DO	7816 CLK
7816_TRX	DIO	7816 TRX

Table 1-12 DMIC Signal Description

Signal	Type	Description
DMIC_CLK	DO	DMIC CLK
DMIC_DI	DI	DMIC DATA IN

Table 1-13 Swire Signal Description

Signal	Type	Description
SWM	DIO	Swire Master
SWS	DIO	Swire Slave

Table 1-14 AOA/AOD Signal Description

Signal	Type	Description
ATSEL0	DO	Antenna select signal 0
ATSEL1	DO	Antenna select signal 1
ATSEL2	DO	Antenna select signal 2

Table 1-15 External Power Amplifier, Low Noise Amplifier Signal Description

Signal	Type	Description
RX_CYC2LNA	DO	External low noise amplifier
TX_CYC2PA	DO	External power amplifier

Table 1-16 USB Signal Description

Signal	Type	Description
DP	DIO	USB DP
DM	DIO	USB DM

Table 1-17 DECODEC Signal Description

Signal	Type	Description
MDEC	DI	Manchester Decodec

Table 1-18 Audio_in Signal Description

Signal	Type	Description
audio_in	AI	Audio input for microphone or line in

Table 1-19 Low Current Comparator Signal Description

Signal	Type	Description
lc_comp_ain<0>	AI	Low current comparator channel 0

Signal	Type	Description
lc_comp_ain<1>	AI	Low current comparator channel 1
lc_comp_ain<2>	AI	Low current comparator channel 2
lc_comp_ain<3>	AI	Low current comparator channel 3
lc_comp_ain<4>	AI	Low current comparator channel 4
lc_comp_ain<5>	AI	Low current comparator channel 5
lc_comp_ain<6>	AI	Low current comparator channel 6
lc_comp_ain<7>	AI	Low current comparator channel 7

Table 1-20 SAR ADC Signal Description

Signal	Type	Description
sar_aio<0>	AI	SAR ADC input channel 0
sar_aio<1>	AI	SAR ADC input channel 1
sar_aio<2>	AI	SAR ADC input channel 2
sar_aio<3>	AI	SAR ADC input channel 3
sar_aio<4>	AI	SAR ADC input channel 4
sar_aio<5>	AI	SAR ADC input channel 5
sar_aio<6>	AI	SAR ADC input channel 6
sar_aio<7>	AI	SAR ADC input channel 7
sar_aio<8>	AI	SAR ADC input channel 8
sar_aio<9>	AI	SAR ADC input channel 9

Table 1-21 Strong Pull Up Signal Description

Signal	Type	Description
PS_PE<0>	AO	Strong pull up 0 enable
PS_PE<1>	AO	Strong pull up 1 enable
PS_PE<2>	AO	Strong pull up 2 enable
PS_PE<3>	AO	Strong pull up 3 enable

Table 1-22 Crystal Signal Description

Signal	Type	Description
xtl_32k_out	AO	32k xtl output pin
xtl_32k_in	AI	32k xtl input pin

NOTE:

- DI: Digital input
- DO: Digital output
- DIO: Digital input/output
- AI: Analog input
- AO: Analog output
- AIO: Analog input/output

2 Memory and MCU

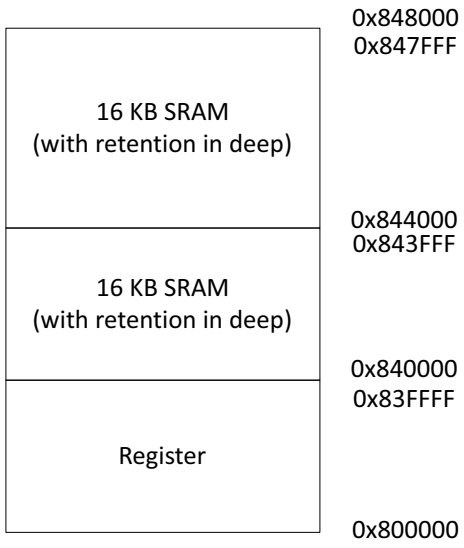
2.1 Memory

The TLSR8270 embeds 32 KB SRAM with retention in deep sleep as data memory, and 512 KB internal FLASH as program memory.

2.1.1 SRAM/Register

SRAM/Register memory map is shown as follows:

Figure 2-1 Physical Memory Map



Register address: 0x800000 ~ 0x83FFFF.

Address for two independent 16 KB SRAMs with retention in deep sleep: 0x840000 ~ 0x843FFF, 0x844000 ~ 0x847FFF.

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

Figure 2-2 Register Space

2.1.2 Flash

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.

2.1.3 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 TLSR8270 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 TLSR8270 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 2-1 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
Audio	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

Mode	Active	Idle	Suspend	Deep Sleep with SRAM Retention	Deep Sleep Without SRAM Retention	Shutdown
(16K+16K) 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 19.3.					

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.
- "full"/"off" for SRAMs:
  - "full": Full speed. In Active, Idle and Suspend mode, the two 16 KB retention SRAMs are powered on and work normally (can be accessed); in Deep Sleep with SRAM retention, the retention SRAMs are powered on, however, the contents of the retention SRAMs can be retained and cannot be accessed.
  - "off": 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 SRAMs are powered down, thus the power consumption is a little higher than in Deep Sleep without SRAM retention, but much lower than in Suspend.

Table 2-2 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 1b'1 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 2-3 Register Configuration for Software Reset

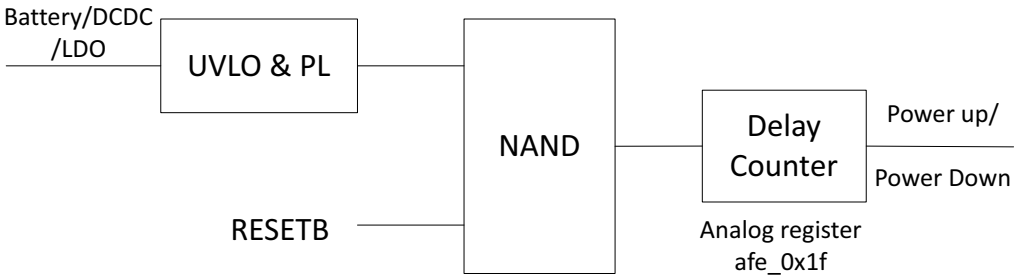
Address	Name	R/W	Description	Default Value
0x60	RST0	RW	Reset control, 1 for reset, 0 for clear [0]: SPI [1]: I2C [2]: RS232, i.e. UART [3]: USB [4]: PWM [5]: QDEC [6]: IR_LEARN [7]: Swire	0x7c
0x61	RST1	RW	[0]: ZB, i.e. Baseband [1]: System Timer [2]: DMA [3]: ALGM [4]: AES [5]: ADC [6]: ALG [7]: PKE (Public Key Engine)	0xff
0x62	RST2	RW	[0]: AIF [1]: Audio [2]: DFIFO [3]: TRNG (True Random Number Generator) [4]: RISC [5]: MCIC [6]: RISC1 (R) [7]: MCIC1 (R)	0xc7
0x6f	PWDNEN	RW	[0]: suspend enable (RW) [4]: clear ramcrc enable (W1C) [5]: reset all (act as watchdog reset) [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 2-3 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 2-4 Analog Register to Control Delay Counters

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

Figure 2-4 Initial Power-Up Sequence

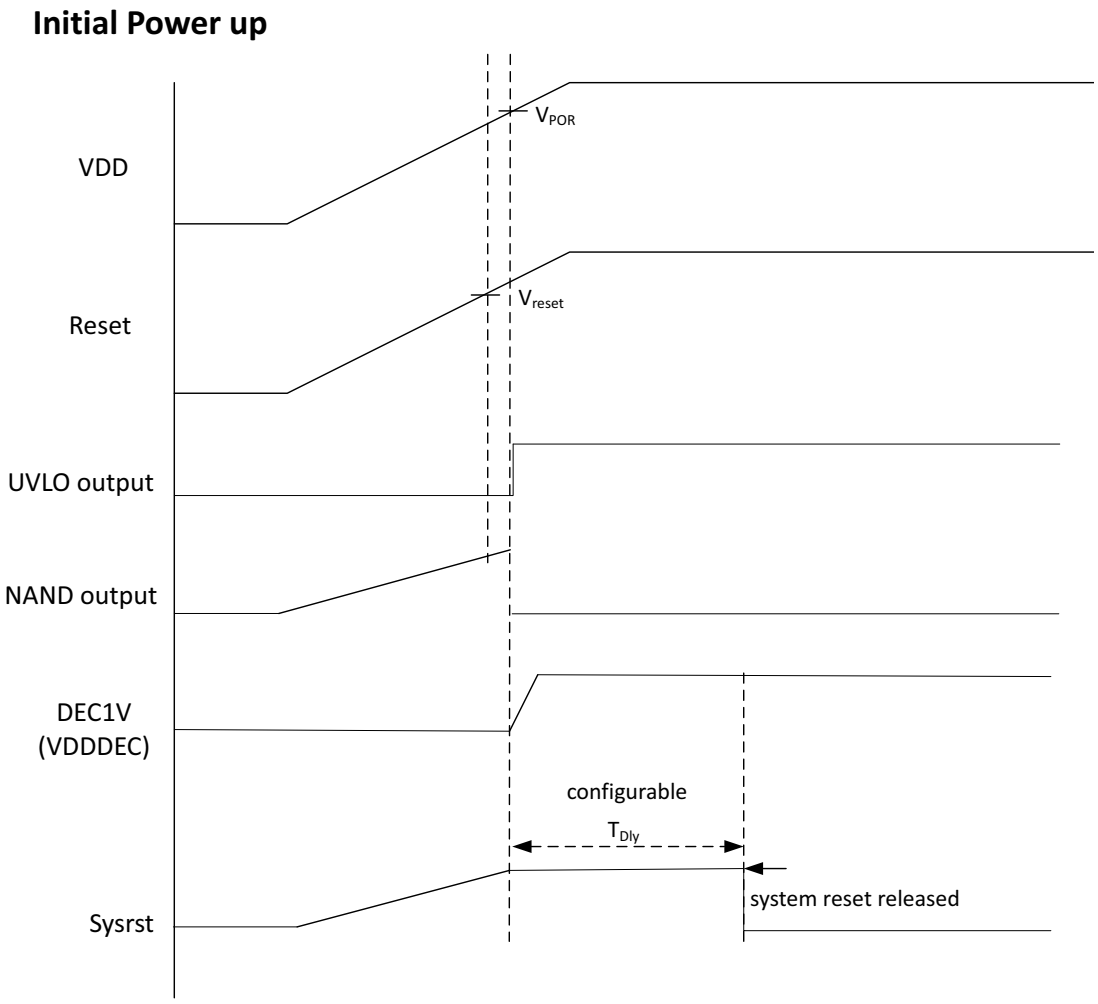


Figure 2-5 Power-Down Sequence

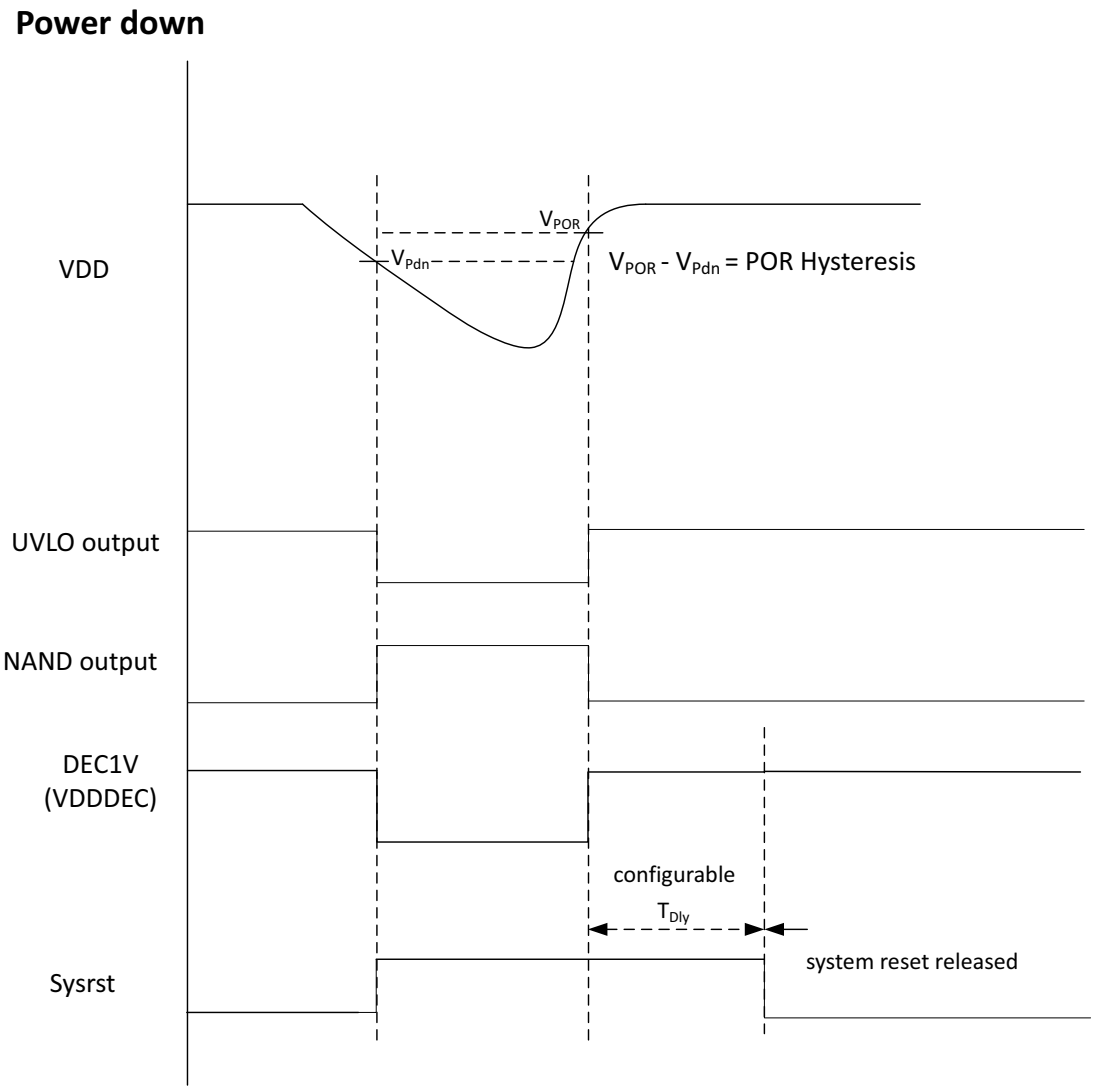


Table 2-5 Characteristics of Initial Power-Up/Power-Down Sequence

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, Audio and USB 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 2-1.)

- In Suspend mode, MCU stalls, all SRAMs are still accessible, the PM module is active, modules such as RF transceiver, Audio 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 two 16 KB retention SRAMs are powered down, while the retention SRAMs 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 SRAMs are 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 and DCDC

The chip embedded DCDC or LDO, depending on which mode is adopted by user, can generate 1.8 V output voltage for internal flash; this DCDC/LDO block also generates 1.4 V output voltage.

Another embedded LDO regulator takes the 1.4 V voltage output from the DCDC/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 DCDC/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.

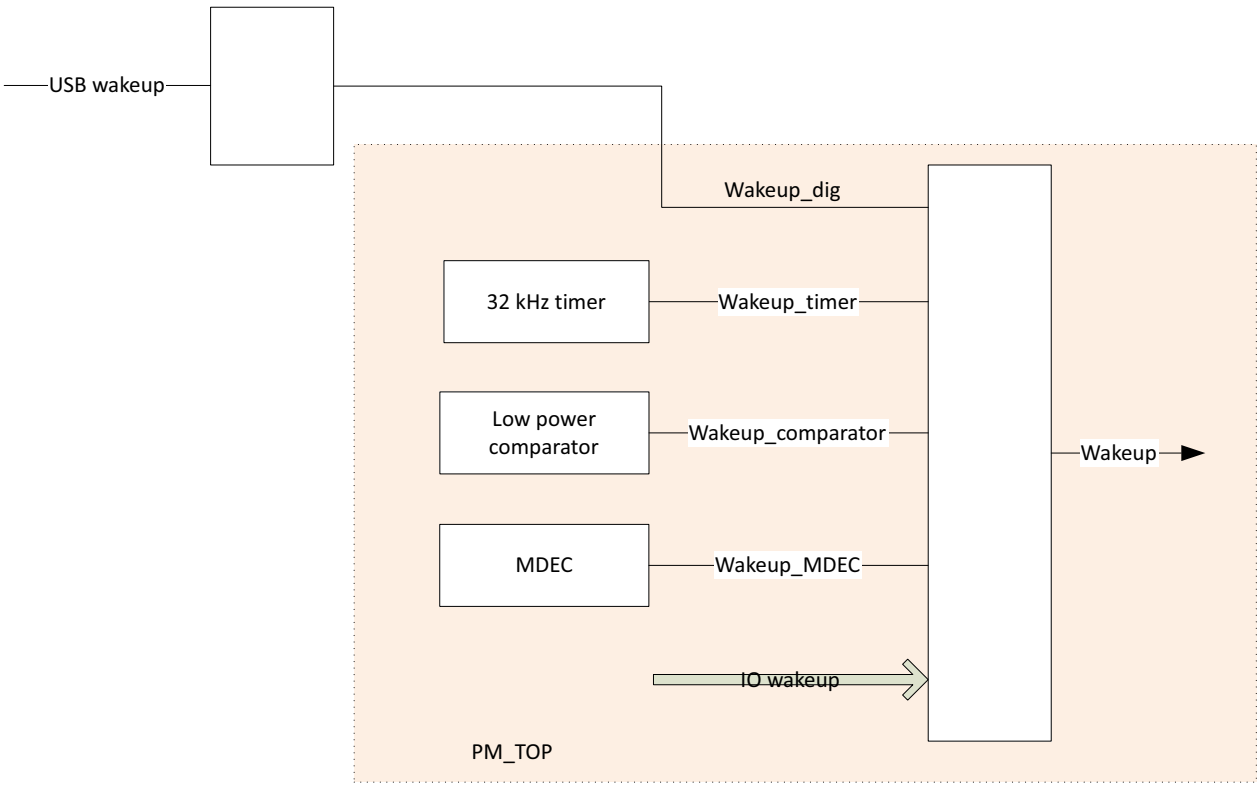
2.5.4 VBAT and VANT Power-Supply Mode

The chip provides two power-supply modes to its PA: VBAT mode and VANT mode.

- In VBAT mode, the PA of the chip is directly supplied by its battery voltage. The maximum output power is related to power supply voltage, for example, the maximum power is 10 dBm or so at 3.3 V power supply, and 6 dBm at 1.8 V.
- In VANT mode, the PA of the chip is supplied with 1.4 V voltage by the embedded DCDC or LDO. In this mode, output power won't change with AVDD basically, and the power stays constantly around 4 dBm. Corresponding to the VBAT mode, the VANT mode is more power-saving at the same TX power.

2.6 Wakeup Sources

Figure 2-6 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 1b'1.

To activate this mode, analog register afe_0x26[4] should also be set as 1b'1.

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_0x26[4] should be set as 1b'1.

2.6.3 Wakeup Source - Low Power Comparator

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

To enable the wakeup source from low power comparator, analog register 0x26[5] should be set as 1b'1. The low power comparator wakeup is active high.

2.6.4 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_0x26[3] should be set as 1b'1 to enable IO wakeup source.

Enabling control analog registers: PA[7:0] enabling control register is afe_0x27[7:0], PB[7:0] enabling control register is afe_0x28[7:0], PC[7:0] enabling control register is afe_0x29[7:0], and PD[7:0] enabling control register is afe_0x2a[7:0]. Total wakeup pins can be up to 32.

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

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

Analog register 0x44[3:0] indicates the wakeup source which triggers system wakeup. After wakeup, the corresponding wakeup status will be set as 1b'1 automatically, and it's needed to write 1 to manually clean the status.

2.6.5 Wakeup Source - MDEC

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

To enable the wakeup source from Manchester Decoder, analog register 0x26[7] should be set as 1b'1.

2.6.6 Register Table

Table 2-6 Analog Registers for Wakeup

Address	R/W	Description	Default Value
afe_0x21	RW	Polarity control registers for IO wakeup 0: high level wakeup, 1: low level wakeup	0x00
afe_0x22	RW		0x00
afe_0x23	RW		0x00
afe_0x24	RW		0x00
afe_0x26	RW	[7] MDEC wakeup enable [6] Low power comparator wakeup enable [5] 32 kHz timer wakeup enable [4] Digital core (USB) wakeup enable [3] IO (pad) wakeup enable [2] Enable/Mask filter for IO (Pad) wakeup 1: Select 16 μs filter to filter out jitter on IO PAD input. 0: IO Pad combinational logic output (disable filter)	0x00
afe_0x27	RW	Enabling control registers for IO wakeup	0x00
afe_0x28	RW		0x00
afe_0x29	RW		0x00
afe_0x2a	RW		0x00

Address	R/W	Description	Default Value
afe_0x44	R	[7] RSVD [6] RSVD [5] RSVD [4] MDEC wakeup status [3] IO (pad) wakeup status [2] Digital core (USB) wakeup status [1] 32 kHz timer wakeup status [0] Low power comparator wakeup status	-

Table 2-7 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

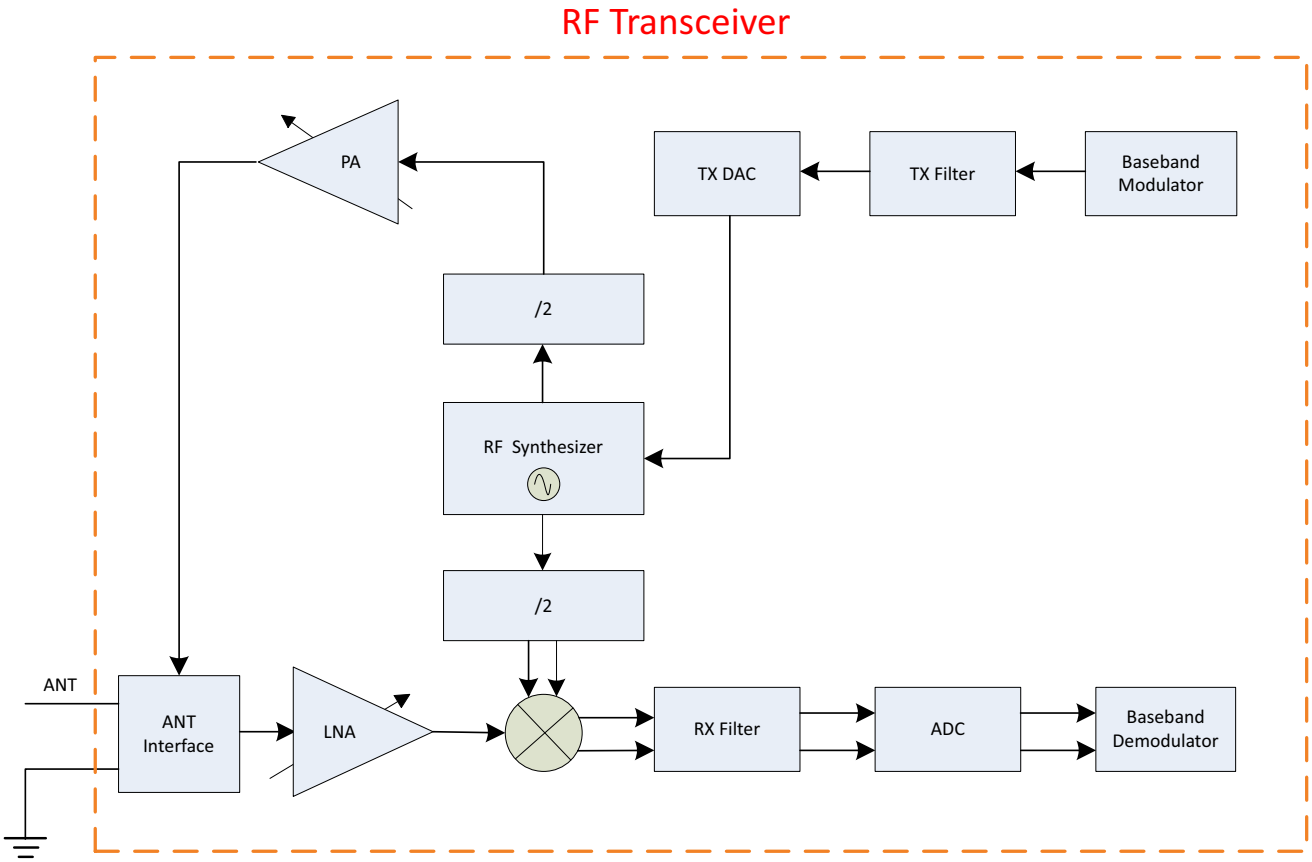
3 BLE/2.4 GHz RF Transceiver

3.1 Block Diagram

The TLSR8270 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, 125 kbps BLE long range mode (S8), 500 kbps BLE long range mode (S2), and proprietary 1 Mbps, 2 Mbps, 250 kbps and 500 kbps mode.

Figure 3-1 Block Diagram of RF Transceiver



To control external PA and LNA, first follow the GPIO lookup table (see Section 7.1.1.1) to configure the specific two pins as TX_CYC2PA and RX_CYC2LNA function, respectively. Note: To use TX_CYC2PA and RX_CYC2LNA function for the two pins, other functions with higher polarity should be disabled at the same time. After the two pins are configured as TX_CYC2PA and RX_CYC2LNA function, the output function is enabled. Generally the two pins are high active: When both the two pins output low level, the external PA and LNA are disabled; when one of the two pins output high level, the external PA/LNA are enabled correspondingly; the two pins won't output high level simultaneously.

Table 3-1 External RF Transceiver Control Example

TX_CYC2PA	RX_CYC2LNA	External RF Transceiver
L	L	Both LNA and PA OFF
L	H	LNA ON
H	L	PA ON
H	H	N/A

The internal PA can deliver a maximum 10 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: 125 kbps, 250 kbps, 500 kbps, 1 Mbps, 2 Mbps.

For the TLSR8270, 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 5.1 specification.

3.3.1 Packet Format

Packet format in standard 1 Mbps BLE mode is shown in Table 3-2.

Table 3-2 Packet Format in Standard 1 Mbps BLE Mode[a]

LSB MSB

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

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

LSB MSB

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

Packet format in standard 500 kbps/125 kbps BLE mode is shown in Table 3-4.

Table 3-4 Packet Format in Standard 500 kbps/125 kbps BLE Mode

LSB MSB

Preamble	Access Address	CI	TERM1	PDU	CRC	TERM2
(10 octets)	(4 octets)	(2 bits)	(3 bits)	(2 ~ 257 octets)	(3 octets)	(3 bits)

Packet format in 2.4 GHz proprietary mode is shown in Table 3-5.

Table 3-5 Packet Format in Proprietary Mode

LSB MSB

Preamble	Address code	Packet Controller + Payload	CRC
(8 bits)	(configurable 3 ~ 5 bytes)	(1 ~ 33 bytes)	(1 ~ 2 bytes)

3.3.2 BLE Location Function

In BLE 1M or 2M mode, BLE location features including AoA (Angel of Arrival) and AoD (Angle of Departure) are supported.

In the location mode of operation, the chip transmits a training sequence concatenated to the normal packet transmissions. In AoA mode of operation, the receiving side has multiple antennas and will be switched during the training sequence period. In AoD mode of operation, the transmitting side has multiple antennas and will be switched during the training sequence period. In either mode, the receiving side will be able to determine based on the phase variations of the received training sequences, the angle of location of the peer device.

3.3.3 RSSI and Frequency Offset

The TLSR8270 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.

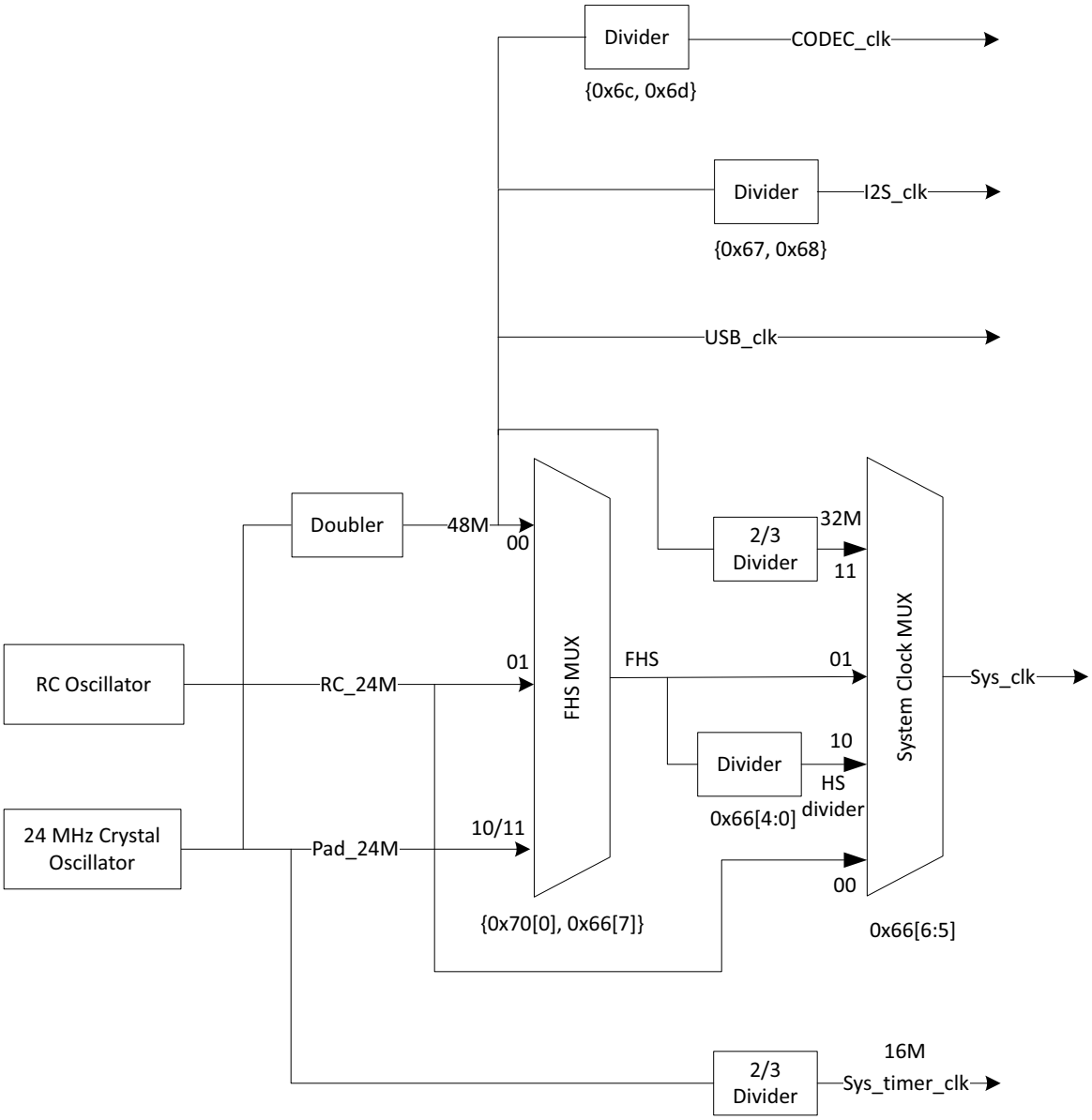
Telink supplies corresponding drivers for user to read RSSI and frequency offset as needed.

4 Clock

4.1 Clock Sources

The TLSR8270 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 CODEC, I2S and USB. The block diagram of the TLSR8270 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), and 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 2b'10 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.

4.3.3 I2S Clock

I2S clock is derived from 48M clock via a frequency divider. The 48M clock is derived from 24M crystal oscillator via a frequency doubler.

Address 0x67[7] should be set as 1b'1 to enable I2S clock. I2S clock frequency dividing factor contains step and mod. Address 0x67[6:0] and 0x68 serve to set I2S clock step[6:0] and mod[7:0] respectively, and mod should be no less than 2*step.

I2S clock frequency, $F_{I2S\ clock}$, equals to 48M * I2S_step[6:0] / I2S_mod[7:0].

4.3.4 CODEC Clock

CODEC clock pin is derived from 48M clock via a frequency divider.

Address 0x6c[7] serves to enable CODEC clock. CODEC clock frequency dividing factor contains step and mod. Address 0x6c[6:0] and 0x6d serve to set CODEC clock step[6:0] and mod[7:0], respectively, and mod should be no less than 2*step.

In this situation, CODEC clock frequency, $F_{CODEC\ clock}$ = 48M * CODEC_step[6:0] / CODEC_mod[7:0].

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
0x64	RW	[0] ZB [1] System Timer [2] DMA [3] ALGM [4] AES [5:6] RSVD [7]: PKE (Public Key Engine)	0x00
0x65	RW	[0] AIF [1] Audio [2] DFIFO [3] TRNG (True Random Number Generator) [4] MC [5] MCIC [6:7] RSVD	0x30

Address	R/W	Description	Default Value
0x66	RW	System clock select [4:0] system clock divider (must exceed 1). If 0x66[6:5] is set as 2b'10, F_{Sysclk} = F_{FHS} / (CLKSEL[4:0]). FHS: refer to 0x70 CLKSEH. [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: 32M clock (48M * 2/3 divider) [7] FHS select (see 0x70[0])	0x06
0x67	RW	[7] I2S clock enable [6:0] I2S step	0x00
0x68	RW	I2S mod I2S clock = 48M*I2S_step[6:0]/I2S_mod[7:0] Mod should be larger than or equal to 2*step.	0x02
0x6c	RW	[7] CODEC clock enable [6:0] CODEC clock step	0x01
0x6d	RW	[7:0] CODEC clock mod CODEC clock = 48M*CODEC_step[6:0]/CODEC_mod[7:0] Mod should be larger than or equal to 2*step.	0x02
0x70	RW	{0x70[0], 0x66[7]} FHS select 2'b00: 48M clock doubled from 24M crystal 2'b01: RC_24M from RC oscillator 2'b1x: Pad_24M from 24M crystal oscillator	0x00
0x73	RW	[1] DMIC clock out select 1: select 32k clock 0: select DMIC interface clock	0x00

5 Timers

5.1 Timer0 ~ Timer2

The TLSR8270 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 2b'00 to select Mode 0; Meanwhile set address 0x620[0] to 1b'1 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 7.1.2 *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 2b'01 to select Mode 1; Meanwhile set address 0x620[3] to 1b'1 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 7.1.2 *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 2b'01 and address 0x621 [0] to 1b'1.

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 2b'11 to select Mode 3, meanwhile set address 0x620[0] to 1b'1 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 1b'1 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 1b'1 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 1b'1 automatically. User can read the watchdog status bit after chip reset to check if the reset source is watchdog, and needs to write 1b'1 to this bit to manually clear the flag.

5.2 32K LTIMER

The TLSR8270 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 TLSR8270 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. 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.	0x00
0x742	RW	[7:0] Byte 2 of system timer counter, write to set initial value.	0x00
0x743	RW	[7:0] Byte 3 of system timer counter, write to set initial value.	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
0x749	R	[0] calibration latch result update irq	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] RSVD [1] enable of system timer [0] write/read mode of 32 kHz timer 1'b1: write; 1'b0: read	0xc1
0x74b	R	[6] read busy status [5:0] RSVD	0x00
0x74c	RW	Byte 0 of 32 kHz Timer write value	0x00
0x74d	RW	Byte 1 of 32 kHz Timer write value	0x00
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

Address	R/W	Description	Default Value
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 TLSR8270, 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_pke, irq_pwm, irq_zb_rt, irq_udc[4:0]} [7] irq_pke [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[1:0], irq_stimer, pm_irq, irq_gpio, usb_reset, usb_250us} [7] RSVD [6] gpio2risc[1] [5] gpio2risc[0] [4] irq_stimer [3] pm_irq_tm [2] irq_gpio [1] usb_reset [0] usb_250us	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_pke (0x641[7]): PKE (Public Key Engine) interrupt
- 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[1:0] (0x642[6] ~ 0x642[5]): gpio2risc[1] ~ gpio2risc[0] interrupt, please refer to Section 7.1.2.
- 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 7.1.2
- 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 1b'1 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_250us for example: First enable the interrupt source by setting address 0x642 bit[0] as 1b'1; then set address 0x643 bit[0] as 1b'1 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_250us IRQ source is valid. Clear this interrupt source by setting address 0x64a bit[0] as 1b'1.

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 1b'1; then set address 0x643 bit[0] as 1b'1 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 1b'1 to manually clear Timer0 status (refer to Section 5.1.1).

7 Interface

7.1 GPIO

The TLSR8270F512ET48 support up to 32 GPIOs. All digital IOs can be used as general purpose IOs.

All GPIOs (including PA[0] ~ PD[7]) have configurable pull-up/pull-down resistor. Please refer to Section 7.1.3 for details.

7.1.1 Basic Configuration

7.1.1.1 GPIO Lookup Table

Table 7-1 GPIO PAD Function Mux

Pad	Default	Register = 3	Register = 2	Register = 1	Register = 0	Register
PA[0]	GPIO	-	UART_RX	PWM0_N	DMIC_DI	0x5a8[1:0]
PA[1]	GPIO	-	I2S_CLK	7816_CLK	DMIC_CLK	0x5a8[3:2]
PA[2]	GPIO	-	PWM0	UART_TX	DO	0x5a8[5:4]
PA[3]	GPIO	-	PWM1	UART_CTS	DI/SDA	0x5a8[7:6]
PA[4]	GPIO	-	PWM2	UART_RTS	CK/SCL	0x5a9[1:0]
PA[5]	DM	-	-	-	DM	0x5a9[3:2]
PA[6]	DP (SWS)	-	-	-	DP (SWS)	0x5a9[5:4]
PA[7]	SWS	-	-	UART_RTS	SWS	0x5a9[7:6]
PB[0]	GPIO	-	ATSEL1	UART_RX	PWM3	0x5aa[1:0]
PB[1]	GPIO	-	ATSEL2	UART_TX	PWM4	0x5aa[3:2]
PB[2]	GPIO	-	RX_CYC2LNA	UART_CTS	PWM5	0x5aa[5:4]
PB[3]	GPIO	-	TX_CYC2PA	UART_RTS	PWM0_N	0x5aa[7:6]
PB[4]	GPIO	-	-	PWM4	SDM_P0	0x5ab[1:0]
PB[5]	GPIO	-	-	PWM5	SDM_N0	0x5ab[3:2]
PB[6]	SPI_DI	-	UART_RTS	SPI_DI/SDA	SDM_P1	0x5ab[5:4]
PB[7]	SPI_DO	-	UART_RX	SPI_DO	SDM_N1	0x5ab[7:6]
PC[0]	GPIO	-	UART_RTS	PWM4_N	I2C_SDA	0x5ac[1:0]
PC[1]	GPIO	-	PWM0	PWM1_N	I2C_SCK	0x5ac[3:2]
PC[2]	GPIO	-	I2C_SDA	7816_TRX/ UART_TX	PWM0	0x5ac[5:4]

Pad	Default	Register = 3	Register = 2	Register = 1	Register = 0	Register
PC[3]	GPIO	-	I2C_SCK	UART_RX	PWM1	0x5ac[7:6]
PC[4]	GPIO	-	PWM0	UART_CTS	PWM2	0x5ad[1:0]
PC[5]	GPIO	-	ATSEL0	UART_RX	PWM3_N	0x5ad[3:2]
PC[6]	GPIO	-	PWM4_N	ATSEL1	RX_CYC2LNA	0x5ad[5:4]
PC[7]	GPIO	-	PWM5_N	ATSEL2	TX_CYC2PA	0x5ad[7:6]
PD[0]	GPIO	–	7816_TRX/ UART_TX	-	RX_CYC2LNA	0x5ae[1:0]
PD[1]	GPIO	–	UART_CTS	-	TX_CYC2PA	0x5ae[3:2]
PD[2]	SPI_CN	–	PWM3	I2S_LR	SPI_CN	0x5ae[5:4]
PD[3]	GPIO	–	7816_TRX/ UART_TX	I2S_SDI	PWM1_N	0x5ae[7:6]
PD[4]	GPIO	–	PWM2_N	I2S_SDO	SWM	0x5af[1:0]
PD[5]	GPIO	–	PWM0_N	-	PWM0	0x5af[3:2]
PD[6]	GPIO	–	ATSEL0	UART_RX	CN	0x5af[5:4]
PD[7]	SPI_CK	–	7816_TRX/ UART_TX	I2S_BCK	SPI_CK/SCL	0x5af[7:6]

Table 7-2 GPIO Setting

Pad	Input	IE	OEN	Output/PE	Polarity	DS	Act as GPIO
PA[0]	0x580[0]	0x581[0]	0x582[0]	0x583[0]	0x584[0]	0x585[0]	0x586[0]
PA[1]	0x580[1]	0x581[1]	0x582[1]	0x583[1]	0x584[1]	0x585[1]	0x586[1]
PA[2]	0x580[2]	0x581[2]	0x582[2]	0x583[2]	0x584[2]	0x585[2]	0x586[2]
PA[3]	0x580[3]	0x581[3]	0x582[3]	0x583[3]	0x584[3]	0x585[3]	0x586[3]
PA[4]	0x580[4]	0x581[4]	0x582[4]	0x583[4]	0x584[4]	0x585[4]	0x586[4]
PA[5]	0x580[5]	0x581[5]	0x582[5]	0x583[5]	0x584[5]	0x585[5]	0x586[5]
PA[6]	0x580[6]	0x581[6]	0x582[6]	0x583[6]	0x584[6]	0x585[6]	0x586[6]
PA[7]	0x580[7]	0x581[7]	0x582[7]	0x583[7]	0x584[7]	0x585[7]	0x586[7]
PB[0]	0x588[0]	0x589[0]	0x58a[0]	0x58b[0]	0x58c[0]	0x58d[0]	0x58e[0]
PB[1]	0x588[1]	0x589[1]	0x58a[1]	0x58b[1]	0x58c[1]	0x58d[1]	0x58e[1]
PB[2]	0x588[2]	0x589[2]	0x58a[2]	0x58b[2]	0x58c[2]	0x58d[2]	0x58e[2]

Pad	Input	IE	OEN	Output/PE	Polarity	DS	Act as GPIO
PB[3]	0x588[3]	0x589[3]	0x58a[3]	0x58b[3]	0x58c[3]	0x58d[3]	0x58e[3]
PB[4]	0x588[4]	0x589[4]	0x58a[4]	0x58b[4]	0x58c[4]	0x58d[4]	0x58e[4]
PB[5]	0x588[5]	0x589[5]	0x58a[5]	0x58b[5]	0x58c[5]	0x58d[5]	0x58e[5]
PB[6]	0x588[6]	0x589[6]	0x58a[6]	0x58b[6]	0x58c[6]	0x58d[6]	0x58e[6]
PB[7]	0x588[7]	0x589[7]	0x58a[7]	0x58b[7]	0x58c[7]	0x58d[7]	0x58e[7]
PC[0]	0x590[0]	afe_0xc0[0]	0x592[0]	0x593[0]/ afe_0xc1[0]	0x594[0]	afe_0xc2[0]	0x596[0]
PC[1]	0x590[1]	afe_0xc0[1]	0x592[1]	0x593[1]/ afe_0xc1[1]	0x594[1]	afe_0xc2[1]	0x596[1]
PC[2]	0x590[2]	afe_0xc0[2]	0x592[2]	0x593[2]/ afe_0xc1[2]	0x594[2]	afe_0xc2[2]	0x596[2]
PC[3]	0x590[3]	afe_0xc0[3]	0x592[3]	0x593[3]/ afe_0xc1[3]	0x594[3]	afe_0xc2[3]	0x596[3]
PC[4]	0x590[4]	afe_0xc0[4]	0x592[4]	0x593[4]/ afe_0xc1[4]	0x594[4]	afe_0xc2[4]	0x596[4]
PC[5]	0x590[5]	afe_0xc0[5]	0x592[5]	0x593[5]/ afe_0xc1[5]	0x594[5]	afe_0xc2[5]	0x596[5]
PC[6]	0x590[6]	afe_0xc0[6]	0x592[6]	0x593[6]/ afe_0xc1[6]	0x594[6]	afe_0xc2[6]	0x596[6]
PC[7]	0x590[7]	afe_0xc0[7]	0x592[7]	0x593[7]/ afe_0xc1[7]	0x594[7]	afe_0xc2[7]	0x596[7]
PD[0]	0x598[0]	0x599[0]	0x59a[0]	0x59b[0]	0x59c[0]	0x59d[0]	0x59e[0]
PD[1]	0x598[1]	0x599[1]	0x59a[1]	0x59b[1]	0x59c[1]	0x59d[1]	0x59e[1]
PD[2]	0x598[2]	0x599[2]	0x59a[2]	0x59b[2]	0x59c[2]	0x59d[2]	0x59e[2]
PD[3]	0x598[3]	0x599[3]	0x59a[3]	0x59b[3]	0x59c[3]	0x59d[3]	0x59e[3]
PD[4]	0x598[4]	0x599[4]	0x59a[4]	0x59b[4]	0x59c[4]	0x59d[4]	0x59e[4]
PD[5]	0x598[5]	0x599[5]	0x59a[5]	0x59b[5]	0x59c[5]	0x59d[5]	0x59e[5]
PD[6]	0x598[6]	0x599[6]	0x59a[6]	0x59b[6]	0x59c[6]	0x59d[6]	0x59e[6]
PD[7]	0x598[7]	0x599[7]	0x59a[7]	0x59b[7]	0x59c[7]	0x59d[7]	0x59e[7]

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 7-1 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 7.1.2.
- Priority: "Act as GPIO" has the highest priority. To configure as multiplexed function, disable GPIO function first.
- afe_0xc0, afe_0xc1, and afe_0xc2 are analog registers; others are digital registers.
- For all unused GPIOs, corresponding "IE" must be set as 0.
- When PA[7] "IE" is set as 1, this pin must be fixed as pull-up/pull-down state (float state is not allowed).
- To use SAR ADC/low power comparator pin function, please refer to corresponding module sections.

7.1.1.2 Multiplexed Functions

Each pin listed in Table 7-1 acts as the function in the "Default Function" column by default.

- PA[5] acts as DM function by default.
- PA[6] acts as DP (SWS) function by default.
- PA[7] acts as SWS function by default.
- PB[6] acts as SPI_DI function by default.
- PB[7] acts as SPI_DO function by default.
- PD[2] acts as SPI_CN function by default.
- PD[7] acts as SPI_CK 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 as 1b'1. After GPIO function is enabled, if the pin is used as output, both the bits in "IE" and "OEN" columns should be set as 1b'0, 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 1b'1, 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.

Example 1: SPI_DO/PWM0/PA[2]

1. The pin acts as GPIO function by default.
   - If the pin is used as general output, both address 0x581[2] (IE) and 0x582[2] (OEN) should be set as 1b'0, then configure address 0x583[2] (Output).
   - If the pin is used as general input, both address 0x581[2] (IE) and 0x582[2] (OEN) should be set as 1b'1, and the input data can be read from address 0x580[2] (Input).

2. To use the pin as SPI_DO function, address 0x586[2] (Act as GPIO) should be set as 1b'0, and 0x5a8[5:4] (Register) should be set as 2b'00.
3. To use the pin as PWM0 function, address 0x586[2] (Act as GPIO) should be set as 1b'0, and 0x5a8[5:4] (Register) should be set as 2b'10.

Example 2: SPI_CN/PWM3/PD[2]

1. The pin acts as SPI_CN function by default.
2. To use it as GPIO function, first set address 0x59e[2] (Act as GPIO) as 1b'1.
   - If the pin is used as general output, both address 0x599[2] (IE) and 0x59a[2] (OEN) should be set as 1b'0, then configure address 0x59b[2] (Output).
   - If the pin is used as general input, both address 0x599[2] (IE) and 0x59a[2] (OEN) should be set to 1b'1, and the input data can be read from address 0x598[2] (Input).
3. To use it as PWM3 function, set address 0x59e[2] (Act as GPIO) as 1b'0, and set 0x5ae[5:4] (Register) to 2b'10.

I2C can also be multiplexed with SPI interface, i.e. I2C_SDA/I2C_SCK can be multiplexed with SPI_DI (DI)/ SPI_CK (CK) respectively.

To select multiplexed SPI/I2C function, please follow the steps below:

Step 1 Disable GPIO function by setting corresponding "Act as GPIO" as 1b'0.

Step 2 Select SPI/I2C function by setting corresponding "Register".

Step 3 Address 0x5b6[7:4] serve to select SPI or I2C output.

Step 4 Address 0x5b7[7:0] serve to select SPI input or I2C input.

Table 7-3 Select Multiplexed SPI/I2C

Pin with Multiplexed SPI/I2C	Act as GPIO	Register	SPI Input Select	I2C Input Select	SPI/I2C Output Select
PA[3]	0x586[3] = 0 Disable GPIO	0x5a8[7:6] = 0 Select DI (I2C_SDA)	5b7[0] 1: as SPI input 0: not as SPI input	5b7[4] 1: as I2C input 0: not as I2C input	0x5b6[4] 1: as SPI/I2C output 0: not as SPI/I2C output
PA[4]	0x586[4] = 0 Disable GPIO	0x5a9[1:0] = 0 Select CK (I2C_SCK)	5b7[1] 1: as SPI input 0: not as SPI input	5b7[5] 1: as I2C input 0: not as I2C input	0x5b6[5] 1: as SPI/I2C output 0: not as SPI/I2C output

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[5:7], PB[0:3]: maximum = 8 mA ("DS" = 1), minimum = 4 mA ("DS" = 0)
- PB[4:7]: maximum = 16 mA ("DS" = 1), minimum = 12 mA ("DS" = 0)
- Other GPIOs (PA[0:4], PC[0:7] and PD[0:7]): maximum = 4 mA ("DS" = 1), minimum = 2 mA ("DS" = 0)

7.1.2 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] interrupt request signal.

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

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

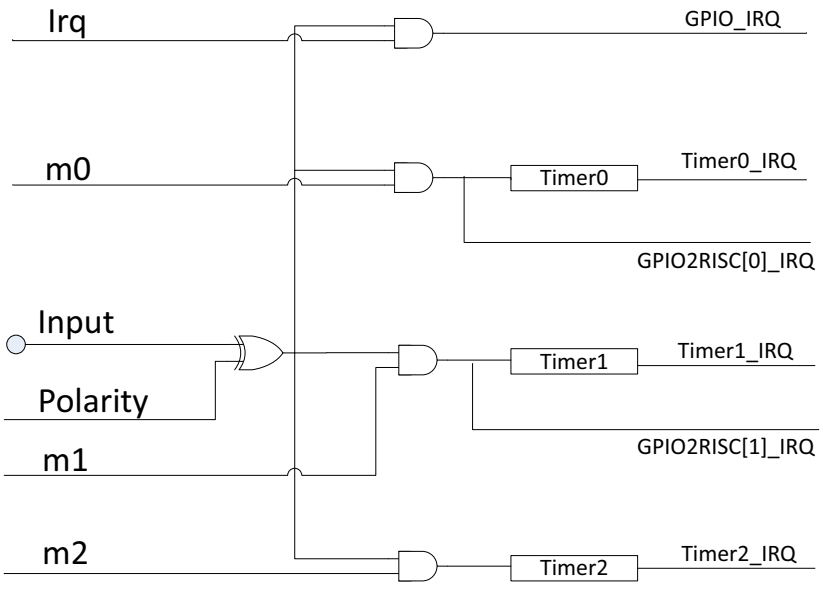
Counting (Mode 1) or control (Mode 2) signal for Timer1 = | ((input ^ polarity) & m1);

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).

Figure 7-1 Logic Relationship Between GPIO and Related Modules



Please refer to Table 7-4 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 0x5b5[3] (irq_enable) to enable GPIO IRQ.

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

User can read addresses 0x5e0 ~ 0x5e3 to see which GPIO asserts GPIO interrupt request signal. Note: 0x5e0[7:0] --> PA[7] ~ PA[0], 0x5e1[7:0] --> PB[7] ~ PB[0], 0x5e2[7:0] --> PC[7] ~ PC[0], 0x5e3[7:0] --> PD[7] ~ PD[0].

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 0x5e8 ~ 0x5eb/0x5f0 ~ 0x5f3/0x5f8 ~ 0x5fb to see which GPIO asserts counting signal (in Mode 1) or control signal (in Mode 2) for Timer0/Timer1/Timer2. Note: Timer0: 0x5e8[7:0] --> PA[7] ~ PA[0], 0x5e9[7:0] --> PB[7] ~ PB[0], 0x5ea[7:0] --> PC[7] ~ PC[0], 0x5eb[7:0] --> PD[7] ~ PD[0]; Timer1: 0x5f0[7:0] --> PA[7] ~ PA[0], 0x5f1[7:0] --> PB[7] ~ PB[0], 0x5f2[7:0] --> PC[7] ~ PC[0], 0x5f3[7:0] --> PD[7] ~ PD[0]; Timer2: 0x5f8[7:0] --> PA[7] ~ PA[0], 0x5f9[7:0] --> PB[7] ~ PB[0], 0x5fa[7:0] --> PC[7] ~ PC[0], 0x5fb[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**".

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

Table 7-4 GPIO IRQ Table

Pin	Input (R)	Polarity 1: Active Low 0: Active High	IRQ	m0	m1	m2
PA[0]	0x580[0]	0x584[0]	0x587[0]	0x5b8[0]	0x5c0[0]	0x5c8[0]
PA[1]	0x580[1]	0x584[1]	0x587[1]	0x5b8[1]	0x5c0[1]	0x5c8[1]
PA[2]	0x580[2]	0x584[2]	0x587[2]	0x5b8[2]	0x5c0[2]	0x5c8[2]
PA[3]	0x580[3]	0x584[3]	0x587[3]	0x5b8[3]	0x5c0[3]	0x5c8[3]
PA[4]	0x580[4]	0x584[4]	0x587[4]	0x5b8[4]	0x5c0[4]	0x5c8[4]
PA[5]	0x580[5]	0x584[5]	0x587[5]	0x5b8[5]	0x5c0[5]	0x5c8[5]
PA[6]	0x580[6]	0x584[6]	0x587[6]	0x5b8[6]	0x5c0[6]	0x5c8[6]
PA[7]	0x580[7]	0x584[7]	0x587[7]	0x5b8[7]	0x5c0[7]	0x5c8[7]

Pin	Input (R)	Polarity 1: Active Low 0: Active High	IRQ	m0	m1	m2
PB[0]	0x588[0]	0x58c[0]	0x58f[0]	0x5b9[0]	0x5c1[0]	0x5c9[0]
PB[1]	0x588[1]	0x58c[1]	0x58f[1]	0x5b9[1]	0x5c1[1]	0x5c9[1]
PB[2]	0x588[2]	0x58c[2]	0x58f[2]	0x5b9[2]	0x5c1[2]	0x5c9[2]
PB[3]	0x588[3]	0x58c[3]	0x58f[3]	0x5b9[3]	0x5c1[3]	0x5c9[3]
PB[4]	0x588[4]	0x58c[4]	0x58f[4]	0x5b9[4]	0x5c1[4]	0x5c9[4]
PB[5]	0x588[5]	0x58c[5]	0x58f[5]	0x5b9[5]	0x5c1[5]	0x5c9[5]
PB[6]	0x588[6]	0x58c[6]	0x58f[6]	0x5b9[6]	0x5c1[6]	0x5c9[6]
PB[7]	0x588[7]	0x58c[7]	0x58f[7]	0x5b9[7]	0x5c1[7]	0x5c9[7]
PC[0]	0x590[0]	0x594[0]	0x597[0]	0x5ba[0]	0x5c2[0]	0x5ca[0]
PC[1]	0x590[1]	0x594[1]	0x597[1]	0x5ba[1]	0x5c2[1]	0x5ca[1]
PC[2]	0x590[2]	0x594[2]	0x597[2]	0x5ba[2]	0x5c2[2]	0x5ca[2]
PC[3]	0x590[3]	0x594[3]	0x597[3]	0x5ba[3]	0x5c2[3]	0x5ca[3]
PC[4]	0x590[4]	0x594[4]	0x597[4]	0x5ba[4]	0x5c2[4]	0x5ca[4]
PC[5]	0x590[5]	0x594[5]	0x597[5]	0x5ba[5]	0x5c2[5]	0x5ca[5]
PC[6]	0x590[6]	0x594[6]	0x597[6]	0x5ba[6]	0x5c2[6]	0x5ca[6]
PC[7]	0x590[7]	0x594[7]	0x597[7]	0x5ba[7]	0x5c2[7]	0x5ca[7]
PD[0]	0x598[0]	0x59c[0]	0x59f[0]	0x5bb[0]	0x5c3[0]	0x5cb[0]
PD[1]	0x598[1]	0x59c[1]	0x59f[1]	0x5bb[1]	0x5c3[1]	0x5cb[1]
PD[2]	0x598[2]	0x59c[2]	0x59f[2]	0x5bb[2]	0x5c3[2]	0x5cb[2]
PD[3]	0x598[3]	0x59c[3]	0x59f[3]	0x5bb[3]	0x5c3[3]	0x5cb[3]
PD[4]	0x598[4]	0x59c[4]	0x59f[4]	0x5bb[4]	0x5c3[4]	0x5cb[4]
PD[5]	0x598[5]	0x59c[5]	0x59f[5]	0x5bb[5]	0x5c3[5]	0x5cb[5]
PD[6]	0x598[6]	0x59c[6]	0x59f[6]	0x5bb[6]	0x5c3[6]	0x5cb[6]
PD[7]	0x598[7]	0x59c[7]	0x59f[7]	0x5bb[7]	0x5c3[7]	0x5cb[7]

7.1.3 Pull-Up/Pull-Down Resistor

All GPIOs (including PA[0] ~ PD[7]) 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_0x15<7:0> 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 1b'1. For the DP/PA[6] 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 7-5 for details.

Take the PA[3] for example: Setting analog register afe_0x0e<7:6> to 2b'01/2b'11/2b'10 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 7-5 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	0x0
Rank x1 x10 x100	**Typical value (depend on actual application)** 18 kOhm 160 kOhm 1 MOhm		
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

Address	Name	Description	Default Value
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
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

Address	Name	Description	Default Value
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
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

Address	Name	Description	Default Value
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

7.2 SWM and SWS

The TLSR8270 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[6] is DP. If swire_usb_en (swire_base+0x1[7]) = 1, when PA[6] (DP) and PA[5] (DM) receive a specific timing sequence (see Figure 7-3), swire_usb_sel will be set to 1, then the Swire slave data will switch to DP and PA[6] will switch to SWS function.

Figure 7-2 Swire Through USB Diagram

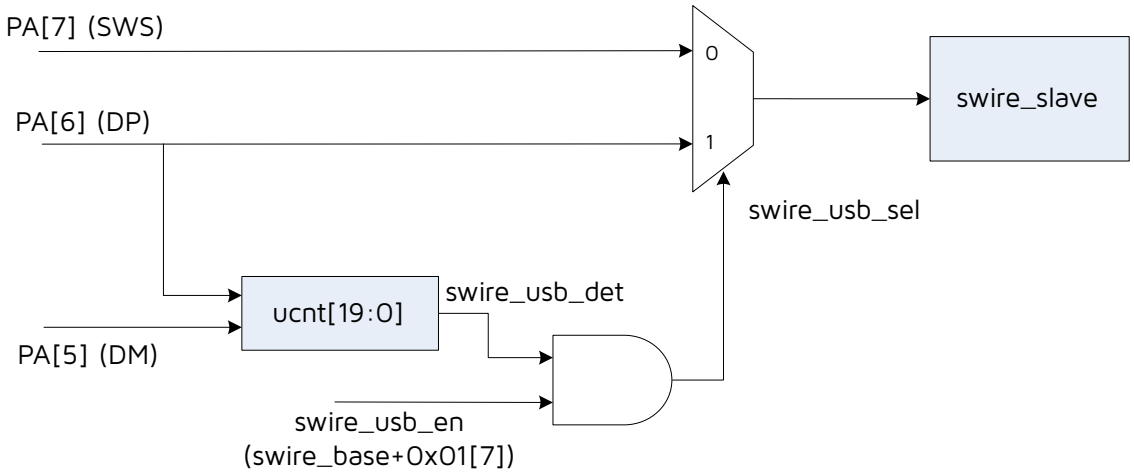
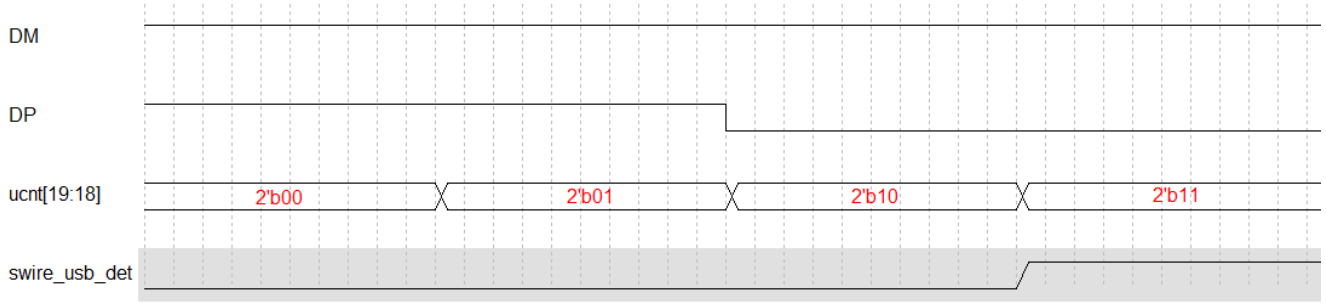


Figure 7-3 shows the timing sequence of enabling Swire through USB. DM should remain high all the time. DP should remain high until ucnt[19:18] = 2'b10, then DP switches to the low level and remains low until ucnt[19:18] = 2'b11, at which point swire_usb_det is set to 1. That is, assuming the system clock is 24M, then the timing sequence should be: DP remains high for about 22 ms and low for about 11 ms.

Figure 7-3 Timing Sequence of Enabling Swire Through USB



7.3 I2C

The TLSR8270 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

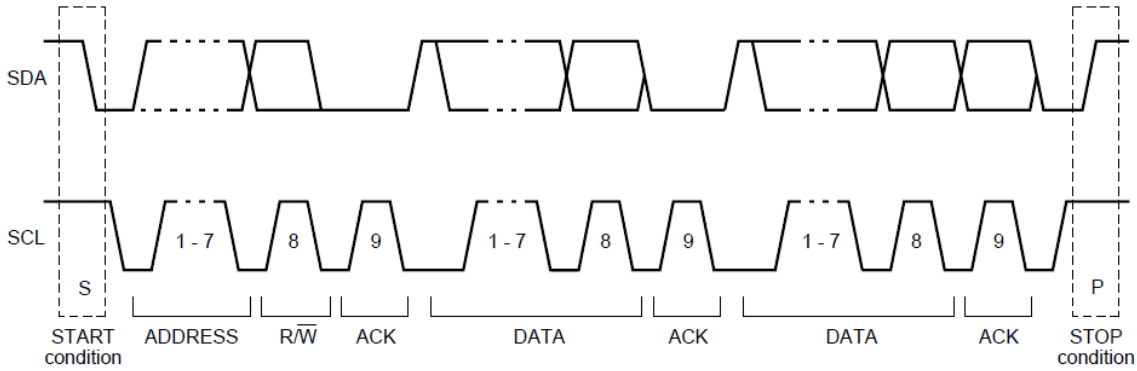
Telink I2C module supports standard-mode (100 kbps) and fast-mode (400 kbps) with restriction that system clock must be by at least 10x of data rate.

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 7-4 I2C Timing Chart



7.3.2 Register Table

Table 7-6 Register Configuration for I2C

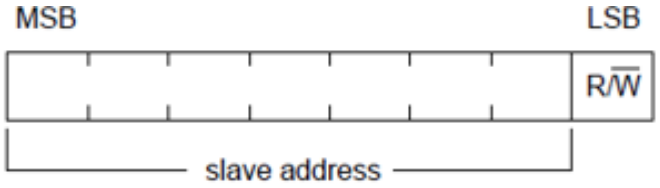
Address	R/W	Description	Default Value
0x00	RW	I2C master clock speed	0x1f
0x01	RW	[7:1]: I2C ID	0x5c
0x02	RW	[0]: master busy [1]: master packet busy [2]: master received status 0 for ACK; 1 for NAK	0x00
0x03	RW	[0]: address auto increase enable [1]: I2C master enable [2]: enable Mapping Mode [3]: r_clk_stretch_en, 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
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

Address	R/W	Description	Default Value
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 TLSR8270 acts as Slave mode by default. I2C slave address can be configured via register I2C_ID (address 0x01) [7:1].

Figure 7-5 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 SCA 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 TLSR8270 according to I2C protocol. I2C module of the TLSR8270 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 TLSR8270, 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 1b'1.

Figure 7-6 Read Format in DMA Mode

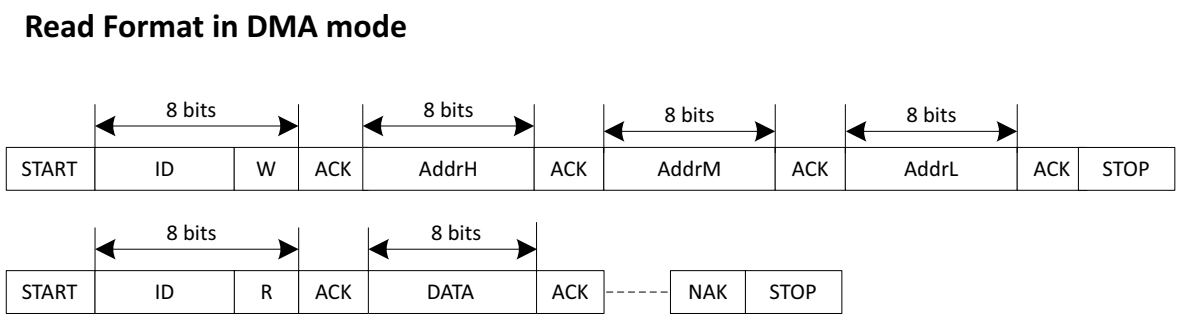
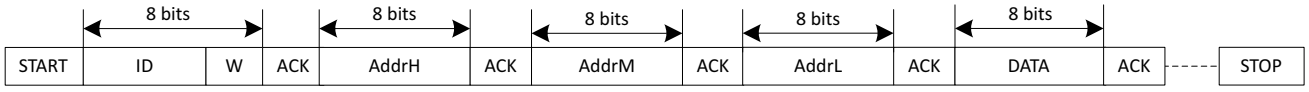


Figure 7-7 Write Format in DMA Mode



7.3.3.2 Mapping Mode

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

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 7-8 Read Format in Mapping Mode

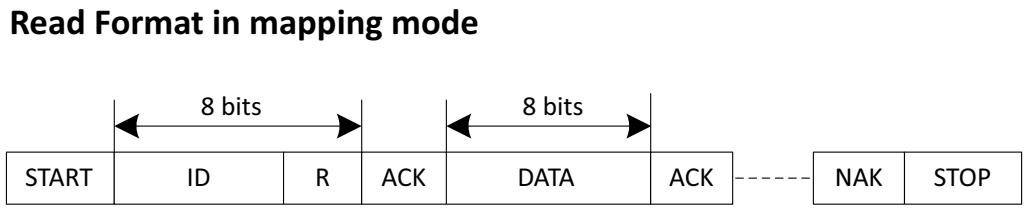
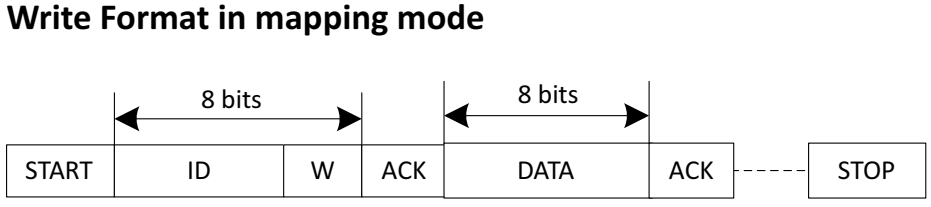


Figure 7-9 Write Format in Mapping Mode



7.3.4 I2C Master Mode

Address 0x03[1] should be set to 1b'1 to enable I2C master mode for the TLSR8270.

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 TLSR8270) 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

The TLSR8270 embeds SPI (Serial Peripheral interface), which could act as Master mode or Slave mode. SPI is a high-speed, full-duplex and synchronous communication bus requiring 4 bus lines including a chip select (CS) line, a data input (DI) line, a data output (DO) line and a clock (CK) line.

7.4.1 Register Table

Table 7-7 Register Configuration for SPI

Address	R/W	Description	Default Value
0x08	RW	[7:0]: SPI data access	0x00
0x09	RW	[0]: mst_csn, control SPI_CSN output when SPI acts as Master [1]: enable master mode [2]: SPI data output disable [3]: 1 for read command; 0 for write command [4]: address auto increase [5]: share_mode [6]: busy status	0x11
0x0a	RW	[6:0]: SPI clock speed [7]: SPI function mode, p_csn, p_scl, p_sda and p_sdo function as SPI if 1	0x05
0x0b	RW	[0]: inverse SPI clock output [1]: data delay half clk	0x00

7.4.2 SPI Master Mode

SPI for the TLSR8270 supports both Master mode and Slave mode and acts as Slave mode by default. Address 0x09 bit[1] should be set to 1b'1 to enable SPI Master mode. Register SPISP is to configure SPI pin and clock: setting address 0x0a bit[7] to 1 is to enable SPI function mode, and corresponding pins can be used as SPI pins; SPI clock = system clock/((clock speed configured in address 0x0a bit[6:0] +1)*2).

Address 0x08 serves as the data register. One reading/writing operation of 0x08 enables the SPI_CK pin to generate 8 SPI clock cycles.

Telink SPI supports four standard working modes: Mode 0 ~ Mode 3. Register SPIMODE (address 0x0b) serves to select one of the four SPI modes:

Table 7-8 SPI Master Mode

SPI Mode	CPOL/CPHA	SPIMODE Register (Address 0x0b)
Mode 0	CPOL = 0, CPHA = 0	bit[0] = 0, bit[1] = 0
Mode 1	CPOL = 0, CPHA = 1	bit[0] = 0, bit[1] = 1
Mode 2	CPOL = 1, CPHA = 0	bit[0] = 1, bit[1] = 0
Mode 3	CPOL = 1, CPHA = 1	bit[0] = 1, bit[1] = 1
CPOL: Clock Polarity When CPOL = 0, SPI_CLK keeps low level in idle state; When CPOL = 1, SPI_CLK keeps high level in idle state. CPHA: Clock Phase When CPHA = 0, data is sampled at the first edge of clock period When CPHA = 1, data is sampled at the latter edge of clock period		

Address 0x09 bit[0] is to control the CS line: when the bit is set to 1, the CS level is high; when the bit is cleared, the CS level is low.

Address 0x09 bit[2] is the disabling bit for SPI Master output. When the bit is cleared, MCU writes data into address 0x08, then the SPI_DO pin outputs the data bit by bit during the 8 clock cycles generated by the SPI_CK pin. When the bit is set to 1b'1, SPI_DO output is disabled.

Address 0x09 bit[3] is the enabling bit for SPI Master reading data function. When the bit is set to 1b'1, MCU reads the data from address 0x08, then the input data from the SPI_DI pin is shifted into address 0x08 during the 8 clock cycles generated by the SPI_CK pin. When the bit is cleared, SPI Master reading function is disabled.

Address 0x09[5] is the enabling bit for share mode, i.e. whether SPI_DI and SPI_DO share one common line.

User can read address 0x09 bit[6] to get SPI busy status, i.e. whether the 8 clock pulses have been sent.

7.4.3 SPI Slave Mode

SPI for the TLSR8270 acts as Slave mode by default. SPI Slave mode supports DMA. User could access registers of the TLSR8270 by SPI interface. It's noted that system clock of TLSR8270 shall be at least 5x faster than SPI clock for reliable connection. Address 0x0a should be written with data 0xa5 by the SPI host to activate SPI Slave mode. SPI Salve only supports Mode 0 and Mode 3.

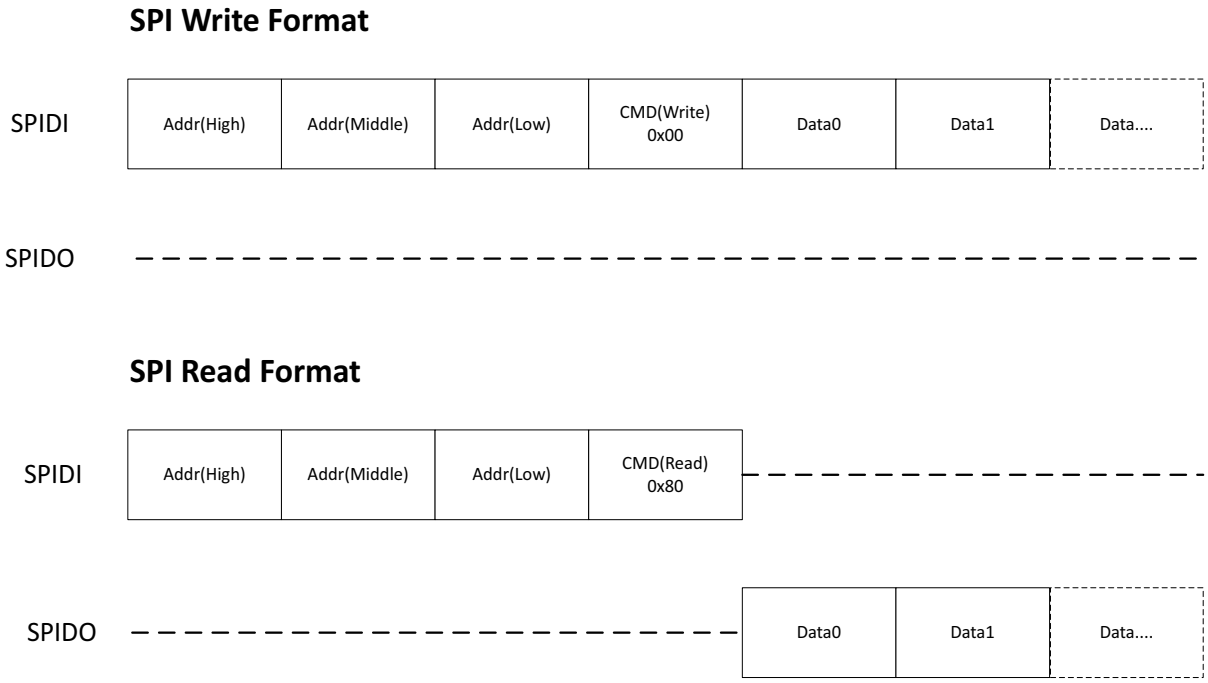
Table 7-9 SPI Slave Mode

SPI Slave Mode	CPOL/CPHA
Mode 0	CPOL = 0, CPHA = 0
Mode 3	CPOL = 1, CPHA = 1

SPI Slave Mode	CPOL/CPHA
Receive data at positive edge of SPI MCLK clock. Send data at negative edge of SPI MCLK clock.	

Address 0x09[4] is dedicated for SPI Slave mode and indicates address auto increment. SPI write command format and read command format are illustrated in the figure below:

Figure 7-10 SPI Write/Read Command Format



7.4.4 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, certain restrictions apply.

See Section 7.3.5 for detailed instructions.

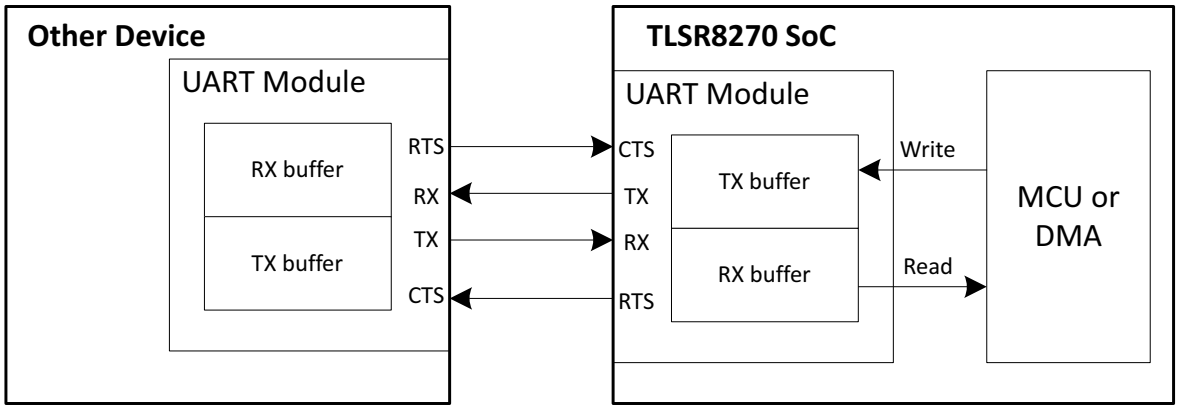
7.5 UART

The TLSR8270 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.

Hardware flow control is supported via RTS and CTS.

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.

Figure 7-11 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.

If RX buffer of the TLSR8270 UART is close to full, the TLSR8270 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 TLSR8270 receives a signal from pin CTS, it indicates that RX buffer of other device is close to full and the TLSR8270 should stop sending data.

Table 7-10 Register Configuration for UART

Address	R/W	Description	Default Value
0x90	RW	Write/read buffer[7:0]	0x00
0x91	RW	Write/read buffer[15:8]	0x00
0x92	RW	Write/read buffer[23:16]	0x00
0x93	RW	Write/read buffer[31:24]	0x00
0x94	RW	[7:0]: uart clk div register uart_clk_div[7:0]	0xff
0x95	RW	[6:0]: uart clk div register uart_clk_div[14:8] uart_sclk = sclk/(uart_clk_div[14:0]+1) [7]: uart_clk_div[15] 1: enable clock divider, 0: disable.	0x0f
0x96	RW	[3:0]: bwpc, bit width, should be larger than 2 Baudrate = uart_sclk/(bwpc+1) [4]: rx dma enable [5]: tx dma enable [6]: rx interrupt enable [7]: tx interrupt enable	0x0f

Address	R/W	Description	Default Value
0x97	RW	[0]: cts select, 0: cts_i, 1: cts _i inverter [1]: cts enable, 1: enable, 0: disable [2]: Parity, 1: enable, 0 :disable [3]: even Parity or odd [5:4]: stop bit 00: 1 bit, 01: 1.5 bits, 1x: 2 bits [6]: ttl [7]: uart tx, rx loopback	0x0e
0x98	RW	[3:0]: rts trig level [4]: rts Parity [5]: rts manual value [6]: rts manual enable [7]: rts enable	0xa5
0x99	RW	[3:0]: rx_irq_trig level [7:4]: tx_irq_trig level	0x44
0x9a	RW	[7:0]: R_rxtimeout_o[7:0] 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	[1:0]: R_rxtimeout_o[9:8] 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. [5]: p7816_en_o [6]: mask_txdone [7]: mask_err	0x01
0x9c	R	[3:0]: rx_buf_cnt [7:4]: tx_buf_cnt	0x00

Address	R/W	Description	Default Value
0x9d	R	[2:0]: rbcnt [3]: irq_o [6:4]: wbcnt [6]: write 1 to clear rx [7]: rx_err, write 1 to clear tx	0x00
0x9e	R	[0]: txdone [1]: tx_buf_irq [2]: rxdone [3]: rx_buf_irq	0x00
0x9f	R	[2:0]: tstate_i [7:4]: rstate_i	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 1b'1 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 TLSR8270 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 TLSR8270 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	RW	[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	RW	[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 1b'0 - high level first 1b'1 - 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 (if pola[3] = 1)	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

Address	R/W	Description	Default Value
0x7b5	R	[15:8] PWM0 cnt value	0x00
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	R	[15:8] PWM3 cnt value	0x00
0x7bc	R	[7:0] PWM4 cnt value	0x00
0x7bd	R	[15:8] PWM4 cnt value	0x00
0x7be	R	[7:0] PWM5 cnt value	0x00
0x7bf	R	[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	0x55
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	0x55
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]: write 1 to clear data in FIFO	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:

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

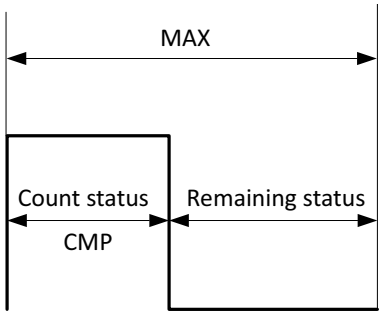
8.4 PWM Waveform, Polarity and Output Inversion

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]).

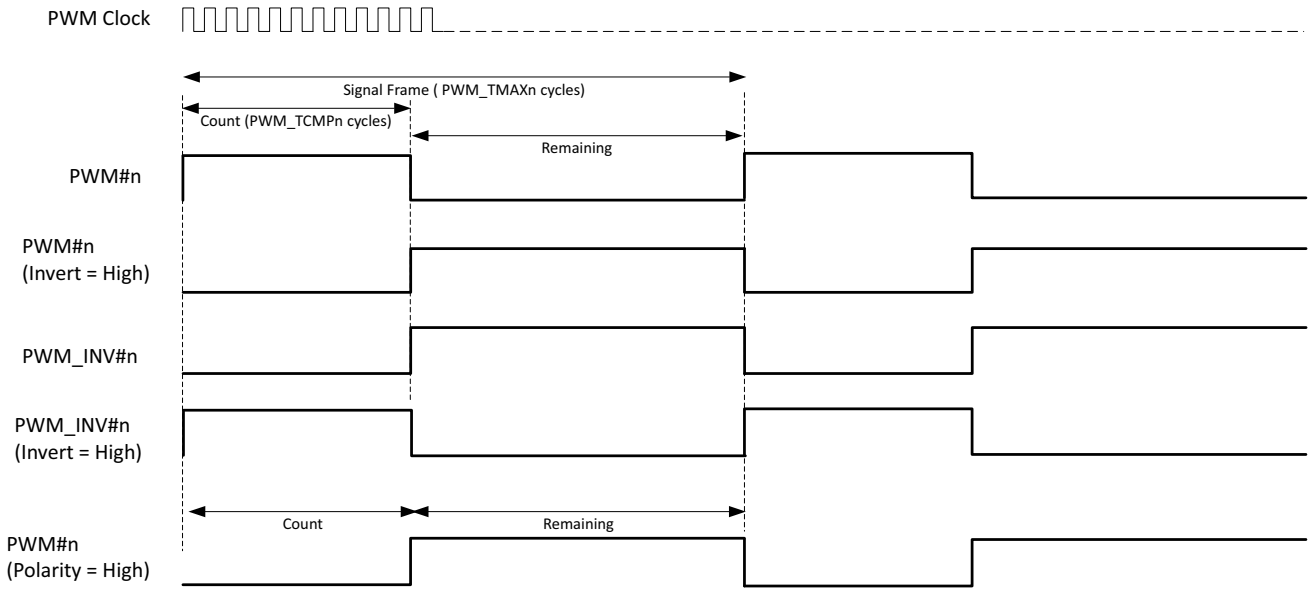
8.4.2 Invert PWM Output

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.

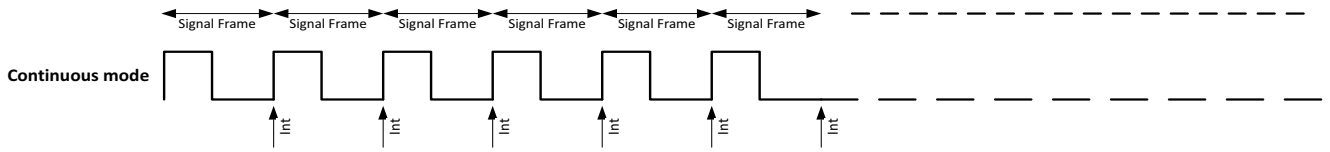
8.5.2 Continuous Mode

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 1b'1. If the interrupt is enabled by setting PWM_MASK0 (address 0x7b0[2:7]) as 1b'1, a frame interruption will be generated. User needs to write 1b'1 to the flag bit to manually clear it.

Figure 8-3 Continuous Mode



8.5.3 Counting Mode

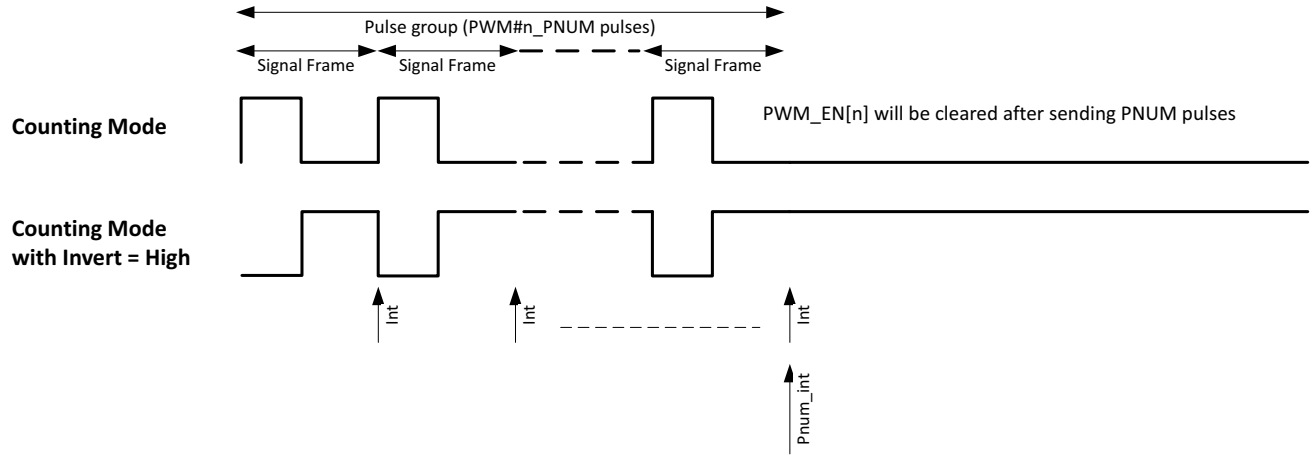
Only PWM0 supports Counting mode. Address 0x783[3:0] should be set as 4b'0001 to select PWM0 counting mode.

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 1b'1. If the interrupt is enabled by setting PWM_MASK0 (address 0x7b0[2]) as 1b'1, a frame interruption will be generated. User needs to write 1b'1 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 1b'1. If the interrupt is enabled by setting PWM_MASK0 (address 0x7b0[0]) as 1b'1, a Pnum interruption will be generated. User needs to write 1b'1 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 4b'0011 to select PWM0 IR mode.

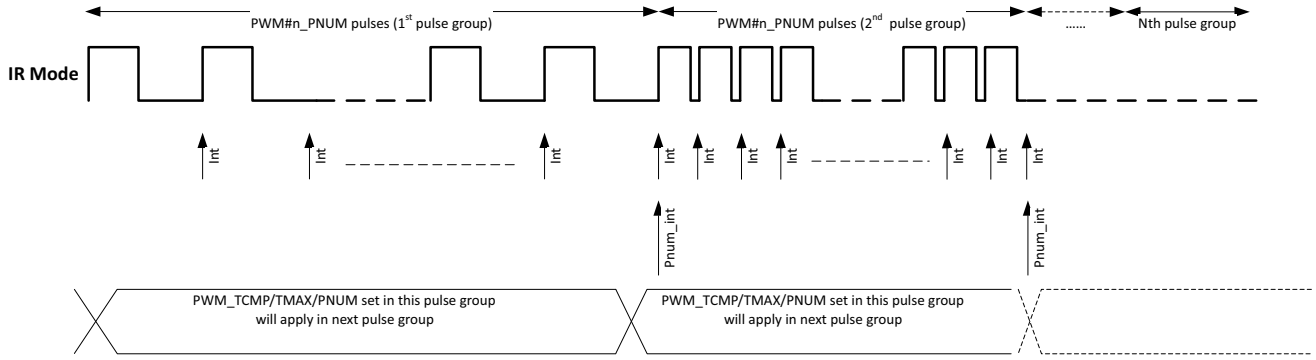
In this mode, specified number of frames is defined as one pulse group. In contrast to Counting mode where PWM0 stops after first pulse group is finished, PWM0 will constantly send pulse groups in IR mode.

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 1b'1. A frame interruption/Pnum interruption will be generated (if enabled by setting address 0x7b0[2]/0x7b0[0] as 1b'1).

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 4b'0111 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:

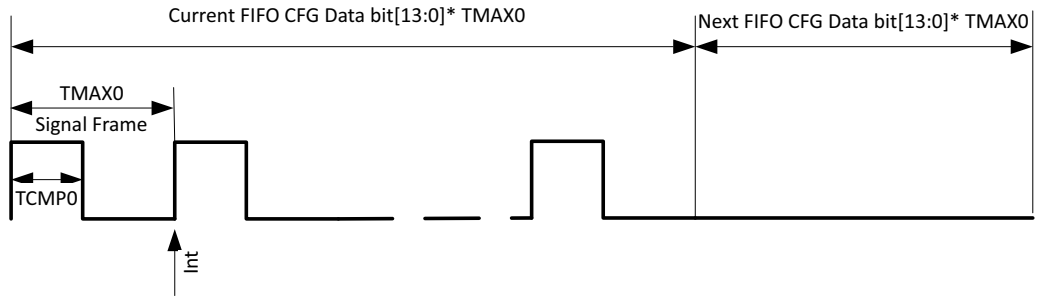
- bit[13:0] defines PWM pulse number of current group.
- bit[14] determines duty cycle and period for current PWM pulse group.
  - 0: use configuration of TCMP0 and TMAX0 in 0x794 ~ 0x797;
  - 1: use configuration of TCMP0_SHADOW and TMAX0_SHADOW in 0x7c4 ~ 0x7c7.
- bit[15] determines whether current 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

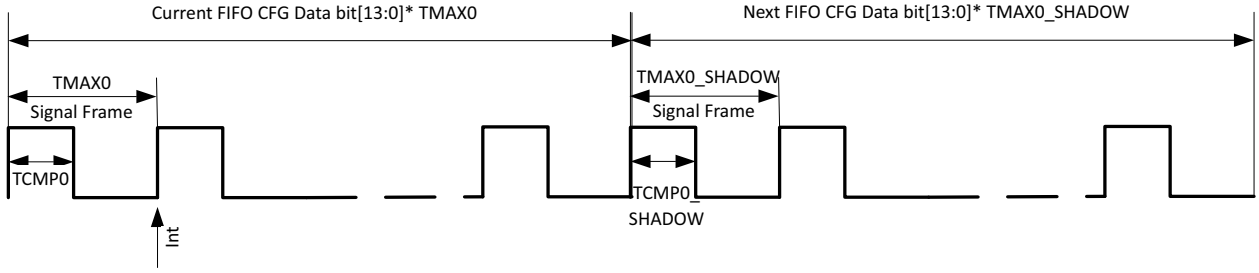


Current FIFO CFG Data bit[15] = 1

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 1b'1 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 4b'1111 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 1b'1 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

1b'1. If the interrupt is enabled by setting PWM_MASK1 (address 0x7b2[0]) as 1b'1, a FIFO mode stop interrupt will be generated. User needs to write 1b'1 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 1b'1. If the interrupt is enabled by setting PWM_MASK1 (address 0x7b2[0]) as 1b'1, a FIFO mode stop interrupt will be generated. User needs to write 1b'1 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 1b'1. To enable various PWM interrupt sources, PWM_MASK0 (address 0x7b0[7:0]) and PWM_MASK1 (address 0x7b2[0]) should be set as 1b'1 correspondingly.

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

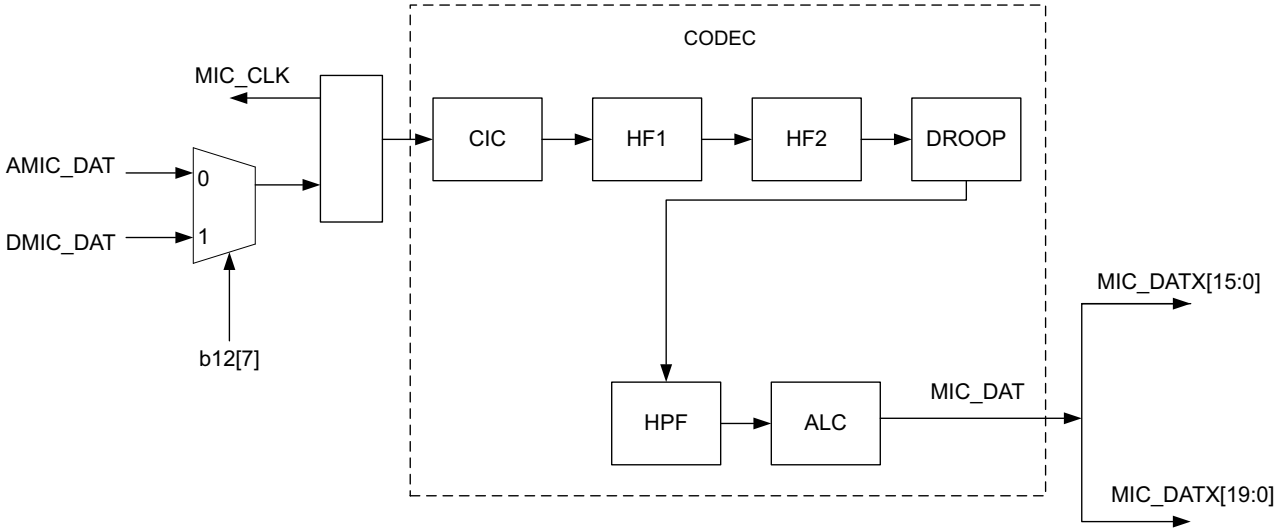
9 Audio

9.1 Audio Input Path

9.1.1 Audio Input Process

Figure 9-1 shows the audio input process.

Figure 9-1 Audio Input Process



9.1.1.1 CODEC

As shown in Figure 9-1, CODEC consists of a CIC filter, 2 Half-wave filters, a compensation filter, and a high pass filter. CODEC is used to down-sample and filter compensate data collected by ADC. User need to enable codec (0xb8b[1]), set output frequency (0xb8a[5:1]), set codec clock mode, enable clock (0xb8a[0], 0xb8a[7:6]), check the table below for detail.

Table 9-1 CODEC Frequency Table

MCLK CLKDIV2 = 0	MCLK CLKDIV2 = 1	ADC SAMPLE RATE	USB	SR [4:0]
USB Mode ('*' indicates backward compatibility with WM8731)				

MCLK CLKDIV2 = 0	MCLK CLKDIV2 = 1	ADC SAMPLE RATE	USB	SR [4:0]
12.000 MHz	24.000 MHz	8 kHz (MCLK/1500)	1	00110 *
		8 kHz (MCLK/1500)	1	00100 *
		8.0214 kHz (MCLK/1496)	1	10111 *
		8.0214 kHz (MCLK/1496)	1	10101 *
		11.0259 kHz (MCLK/1088)	1	11001
		12 kHz (MCLK/1000)	1	01000
		16 kHz (MCLK/750)	1	01010
		22.0588 kHz (MCLK/544)	1	11011
		24 kHz (MCLK/500)	1	11100
		32 kHz (MCLK/375)	1	01100 *
		44.118 kHz (MCLK/272)	1	10011 *
		44.118 kHz (MCLK/272)	1	10001 *
		48 kHz (MCLK/250)	1	00010 *
		48 kHz (MCLK/250)	1	00000 *
		88.235 kHz (MCLK/136)	1	11111 *
		96 kHz (MCLK/125)	1	01110 *

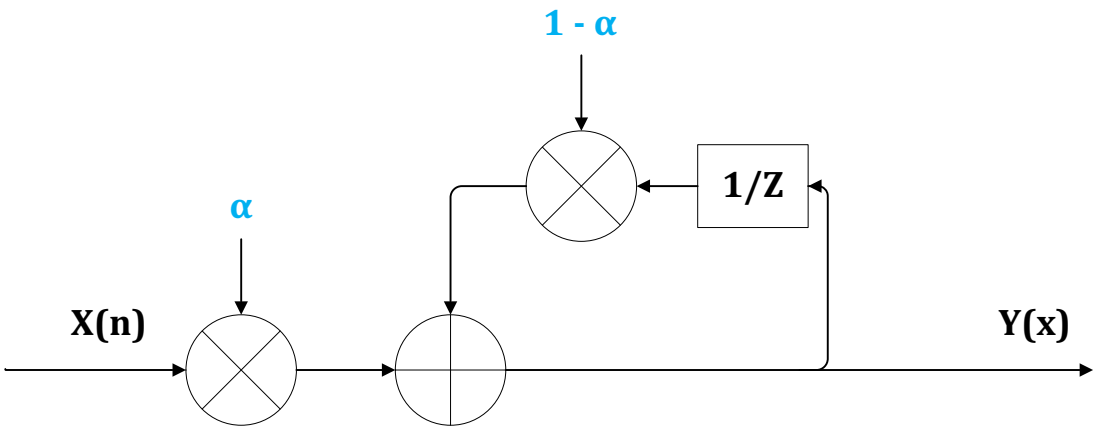
9.1.1.2 ALC

ALC module consists auto and manual digital regulate.

Auto Regulate in Digital Mode

First, detect input data envelop with Average filter, adjust parameter α = 2^(-K1), K1(0xb85[7:4]) to change the speed if needed. Figure 9-2 below shows the structure of Average filter.

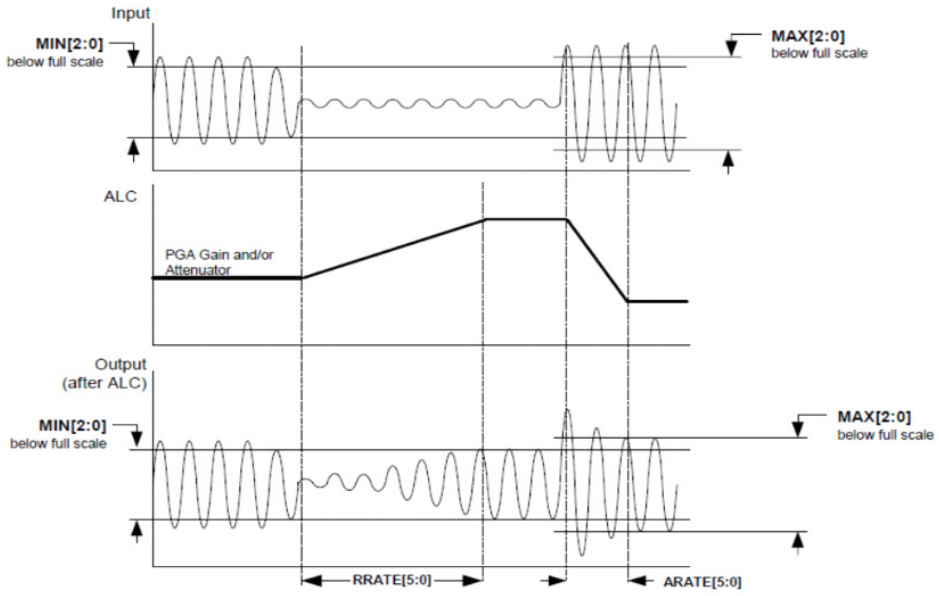
Figure 9-2 Average Filter Envelope Detect



Set ALC_SEL(0xba1[6:5]) to 2'b00 to disable ALC, 2'b01 to set ALC to right channel mode, 2'b10 to set ALC to left channel mode, and 2'b11 to set ALC to stereo mode. As shown in Figure 9-3, compare the detected envelop with regulate reference value ALCL(0xba0[3:0]), reduce it if it is bigger than ALCL, otherwise increase it. Change the reducing/increasing speed by adjusting ATK(0xba2[3:0]) and DCY(0xba2[7:4]). When the gain is decreasing (DCY), set ALC_HLD(0xba1[3:0) to hold the gain, if the gain is higher than MAXGAIN(0xba0[7:4]), then freeze it to MAXGAIN, if it is lower than MINGAIN(0xba4[2:0]), freeze it to MINGAIN.

Noise Gate, together with ALC, is to prevent noise amplification. NGAT(0xba3[0]) is the enable trigger, NGTH(0xba3[7:3])is the programmable noise gate threshold, when the input signal is lower than NGTH, set NGG(0xba3[2:1]) to 2'b00 to keep the gain, set it to 2'b01 to mute the signal, 2'b10 to soft mute the signal.

Figure 9-3 ATK/DCY Processing



Manual Regulate in Digital Mode

Adjust the gain of the output data of HPF in MIC input path by configure 0xb12[5:0].

- 0xb12[1:0] = 01: Input data*1.25

- 0xb12[1:0] = 10: Input data*1.5
- 0xb12[1:0] = 11: Input data*1.75

0xb12[5:2] is used to shift the processed data.

- 0xb12[5:2] = 8: no shift
- 0xb12[5:2] + 1: shift 1 bit left
- 0xb12[5:2] - 1: shift 1 bit right

9.1.2 Audio Input Path

Figure 9-4 below shows the audio input path.

There are four types of audio input path: Digital microphone (DMIC), Codec (I2S), USB and analog input channel (AMIC), which is selectable by writing address 0xb11[3:2], 0xb11[5:4].

Address 0xb11[4] should be set as 1b'1/1b'0 to select mono/stereo input for audio input processing module.

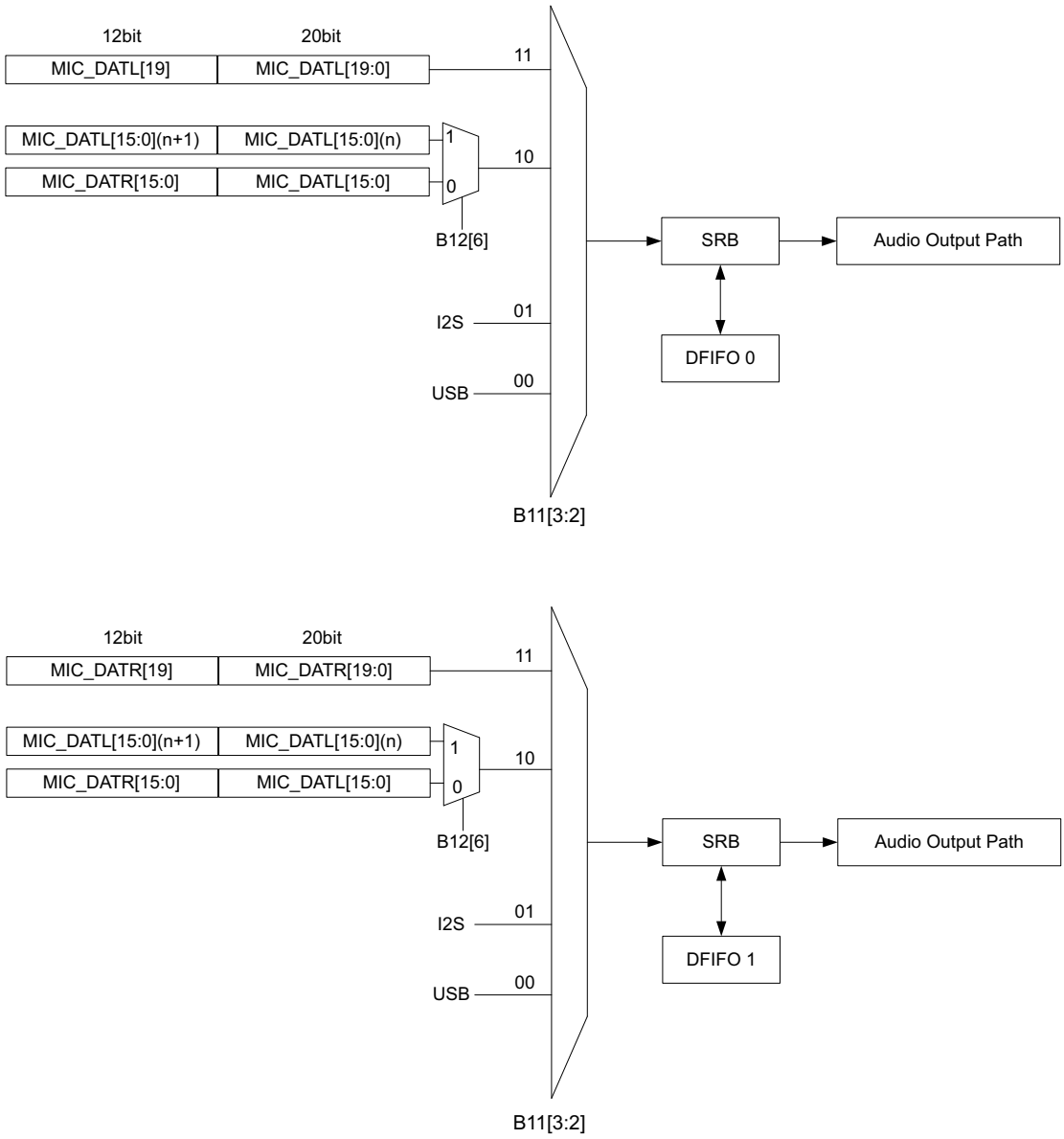
The audio data flow direction is shown in the table below.

Table 9-2 Audio Data Flow Direction

Data Path		Target SRAM		
		FIFO0	FIFO1	FIFO2[a]
DMIC	CIC/HF1/HF2/DROOP/HPF/ALC	√	√	×
AMIC		√	√	×
USB	Direct to SRAM	√	√	×
I2S		√	√	×
ADC CH		×	×	√

a. FIFO2 is not shown in Figure 9-4.

Figure 9-4 Audio Input Path



9.1.2.1 AMIC Input

When 0xb12[7] is set to 0, the system is set to AMIC input mode, AMIC has 2 formats of inputs, 16 bit and 20 bit, while 0xb11[3:2] are the select bits.

16 Bit AMIC Input

Set 0xb11[3:2] to 0x10, FIFO 0 will choose 16 bit MIC input. The SoC supports only single channel for AMIC input, so 0xb12[6] should be set to 1, to enable mono mode (mono mode will merge 2 16-bit data from single channel into 1 32-bit data, while stereo mode will merge 2 16-bit data from 2 channels into 1 32-bit data).

20 Bit AMIC Input

In this case, 0xb12[6] need to be set to 0, for no need to merge 20 bit data. Set 0xb11[3:2] to 0x11, FIFO 0 will choose 20 bit MIC input (only 20 bits are effective).

9.1.2.2 DMIC Input

When 0xb12[7] is set to 1, the system is set to DMIC input mode, DMIC has 2 formats of inputs, 16 bit and 20 bit, while 0xb11[3:2] are the select bits.

16 Bit DMIC Input

Set 0xb11[3:2] to 0x10, FIFO 0 will choose 16 bit MIC input. When DMIC input is single-channel, set 0xb12[6] to 1 to enable mono mode, when DMIC input is dual-channel, set 0xb12[6] to 0 to enable stereo mode.

20 Bit DMIC Input

In this case, 0xb12[6] need to be set to 0, for no need to merge 20 bit data. Set 0xb12[3:2] to 0x11, FIFO 0 will choose 32 bit MIC input (only 20 bits are effective), and the data from left channel will be written into SRAM. Set 0xb12[5:4] to 0x11, FIFO 1 will choose 32 bit MIC input (only 20 bits are effective), and the data from right channel will be written into SRAM.

9.1.2.3 I2S Input

Address 0xb11[3:2] /0xb11[5:4]should be set as 2b'01 to select I2S as DFIFO0/DFIFO1 audio input.

Digital I2S audio interface supports Master mode only, 16-bit data width, and variable sampling rate: 8K/16K/ 22.05K/24K/32K/44.1K/48K. The sampling rate is determined by I2S clock.

For I2S clock configuration, please refer to Section 4.3.3.

Address 0x560[5]/[4]/[1] should be set to "1" to enable I2S interface, I2S Recorder and I2S Player, respectively. I2S interface includes one configurable clock line, one data line and one channel selection line. Data generated by the audio codec will be written into FIFO after implementing conversion via I2S Recorder and audio input processing.

9.1.2.4 USB Host Input

Address 0xb11[3:2]/0xb11[5:4] should be set as 2b'00 to select USB as DFIFO0/DFIFO1 audio input.

Packet transmitted by USB Host will be written into FIFO after implementing conversion via USB Interface and audio input processing.

Address 0x560[3] should be set to 1b'1 to enable ISO player.

9.1.2.5 DFIFO

As shown in Table 9-2, for any type of audio input path, the data will be finally written into DFIFO (DMA FIFO) 0, 1 or 2.

Address 0xb10[0]/[1]/[2] should be set as 1b'1 to enable audio input of DFIFO 0 ~ 2.

DFIFO supports auto mode and manual mode. It's highly recommended to clear address 0xb2c[0] to select auto mode.

Take DFIFO0 as an example:

- Address 0xb00, 0xb01 and 0xb03 serve to set base address for DFIFO0, i.e. starting address to write/ read data into/from DFIFO0.
- Address 0xb02 serves to set depth (i.e. the maximum data number) for DFIFO0. Suppose address 0xb02 is set as 0x01, then the DFIFO0 depth is 4 words, i.e. 16 bytes.
- Current data number (difference value of write-pointer and read-pointer) in DFIFO0 can be read from address 0xb20 and 0xb21.

- User can check current DFIFO0 read pointer/write pointer location by reading address 0xb14 ~ 0xb15/ 0xb16 ~ 0xb17.
- When current data number in DFIFO0 is less than the underflow threshold set in address 0xb0c, address 0xb13 bit[0] and bit[4] will be set as 1b'1 successively, and a FIFO0 low interrupt will be generated if enabled via 0xb10[4].
  Address 0xb13[4] will be automatically cleared when the data number in DFIFO0 is no less than the threshold; address 0xb13[0] needs to be cleared manually.
- When current data number in DFIFO0 is more than the overflow threshold set in address 0xb0d, address 0xb13 bit[1] and bit[5] will be set as 1b'1 successively, and a FIFO0 high interrupt will be generated if enabled via 0xb10[5].
  Address 0xb13[5] will be automatically cleared when the data number in DFIFO0 is no more than the threshold; address 0xb13[1] needs to be cleared manually.
- When current data number in DFIFO1 is more than the overflow threshold set in address 0xb0e, address 0xb13 bit[2] and bit[6] will be set as 1b'1 successively, and a FIFO0 high interrupt will be generated if enabled via 0xb10[6].
  Address 0xb13[6] will be automatically cleared when the data number in DFIFO0 is no more than the threshold; address 0xb13[2] needs to be cleared manually.
- When current data number in DFIFO2 is more than the overflow threshold set in address 0xb0f, address 0xb13 bit[3] and bit[7] will be set as 1b'1 successively, and a FIFO0 high interrupt will be generated if enabled via 0xb10[7].
  Address 0xb13[7] will be automatically cleared when the data number in DFIFO0 is no more than the threshold; address 0xb13[3] needs to be cleared manually.

9.1.3 Register Configuration

Table 9-3 Audio Input Registers

Address	R/W	Description	Default Value
0x67	RW	[7]: I2S clock enable [6:0]: I2S step	0x00
0x68	RW	I2S mod I2S clock = 48M*I2S_step[6:0]/I2S_mod[7:0], Mod should be larger than or equal to 2*step.	0x02
0x6c	RW	[7]: codec MCLK enable [6:0]: codec step	0x01
0x6d	RW	codec mod MCLK = 48M*codec_step[6:0]/codec_mod[7:0], Mod should be larger than or equal to 2*step. Always set codec clk 24M/12M	0x02
0xb00	RW	FIFO0 base address [7:0]	0x00

Address	R/W	Description	Default Value
0xb01	RW	FIFO0 base address [15:8]	0x40
0xb02	RW	FIFO depth = FIFO0_DEPTH*4words	0x7f
0xb03	RW	FIFO0 base address [18:16]	0x04
0xb04	RW	FIFO1 base address [7:0]	0x00
0xb05	RW	FIFO1 base address [15:8]	0x48
0xb06	RW	FIFO depth = FIFO1_DEPTH*4words	0x7f
0xb07	RW	FIFO1 base address [18:16]	0x04
0xb08	RW	FIFO2 base address [7:0]	0x00
0xb09	RW	FIFO2 base address [15:8]	0x3c
0xb0a	RW	FIFO depth = FIFO2_DEPTH*4words	0x3f
0xb0b	RW	FIFO2 base address [18:16]	0x04
0xb0c	RW	FIFO0 low level	0x20
0xb0d	RW	FIFO0 high level	0x60
0xb0e	RW	FIFO1 high level	0x20
0xb0f	RW	FIFO2 high level	0x20
0xb10	RW	DFIFO enable [0]: enable audio input of FIFO0 [1]: enable audio input of FIFO1 [2]: enable audio input of FIFO2 [3]: enable audio output of FIFO0 [4]: FIFO0 low interrupt enable [5]: FIFO0 high interrupt enable [6]: FIFO1 high interrupt enable [7]: FIFO2 high interrupt enable	0xf9

Address	R/W	Description	Default Value
0xb11	RW	[1:0]: change sampling point, set 2'b01 (rsvd for software) [3:2]: FIFO0 input sel: 2'b00: USB 2'b01: I2S 2'b10: 16-bit mic in 2'b11: 20-bit mic l in [5:4]: FIFO1 input sel 2'b00: USB 2'b01: I2S 2'b10: 16-bit mic in 2'b11: 20-bit mic r in [6]: mic l channel en [7]: mic r channel en	0x21

Address	R/W	Description	Default Value
0xb12	RW	[5:0]: mic vol control 6'h00: -48 dB 6'h04: -42 dB 6'h08: -36 dB 6'h0c: -30 dB 6'h10: -24 dB 6'h14: -18 dB 6'h18: -12 dB 6'h1c: -6 dB 6'h20: 0 dB 6'h24: 6 dB 6'h28: 12 dB 6'h2c: 18 dB 6'h30: 24 dB 6'h34: 30 dB 6'h38: 36 dB 6'h3c: 42 dB [6]: r_mono_en 0: mic mono 1: mic stereo [7]: mic_sel 0: AMIC 1: DMIC	0x20
0xb13	R	[0]: FIFO0 low interrupt flag. Write 1 to clear. [1]: FIFO0 high interrupt flag. Write 1 to clear. [2]: FIFO1 high interrupt flag. Write 1 to clear. [3]: FIFO2 high interrupt flag. Write 1 to clear. [4]: FIFO0 low [5]: FIFO0 high [6]: FIFO1 high [7]: FIFO2 high	0x00
0xb14	R	FIFO READ PTR low byte	0x00
0xb15	R	[3:0]: FIFO READ PTR high byte	0x00
0xb16	R	FIFO WRITE PTR low byte	0x00

Address	R/W	Description	Default Value
0xb17	R	[3:0]: FIFO WRITE PTR high byte	0x00
0xb18	R	FIFO READ PTR low byte	0x00
0xb19	R	[3:0]: FIFO READ PTR high byte	0x00
0xb1a	R	FIFO WRITE PTR low byte	0x00
0xb1b	R	[3:0]: FIFO WRITE PTR high byte	0x00
0xb1c	R	FIFO READ PTR low byte	0x00
0xb1d	R	[3:0]: FIFO READ PTR high byte	0x00
0xb1e	R	FIFO WRITE PTR low byte	0x00
0xb1f	R	[3:0]: FIFO WRITE PTR high byte	0x00
0xb20	R	FIFO DATA NUMBER low byte	0x00
0xb21	R	FIFO DATA NUMBER high byte	0x00
0xb22	-	Reserved	-
0xb23	-	Reserved	-
0xb24	R	FIFO DATA NUMBER low byte	0x00
0xb25	R	FIFO DATA NUMBER high byte	0x00
0xb26	-	Reserved	-
0xb27	-	Reserved	-
0xb28	R	FIFO DATA NUMBER low byte	0x00
0xb29	R	FIFO DATA NUMBER high byte	0x00
0xb2a	-	Reserved	-
0xb2b	-	Reserved	-
0xb2c	RW	[0]: 0: FIFO auto mode, 1: enable FIFO manual mode	0x00
0xb2d	-	Reserved	-
0xb2e	-	Reserved	-
0xb2f	-	Reserved	-
0xb30	W	FIFO manual mode data in[7:0]	0x00
0xb31	W	FIFO manual mode data in[15:8]	0x00
0xb32	W	FIFO manual mode data in[23:16]	0x00

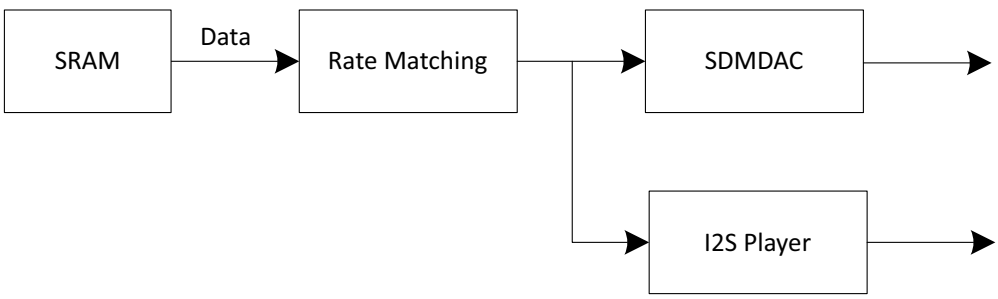
Address	R/W	Description	Default Value
0xb33	W	FIFO manual mode data in[31:24]	0x00
0xb34 ~ 0xb3f	-	Reserved	-
0xb80	RW	[0]: hpf_en [1]: RSVD [2]: RSVD [7:6]: alc K3	0x05
0xb84 ~ 0xb81	-	Reserved	-
0xb85	RW	[7:4]: alc K1	0x5e
0xb86	-	Reserved	-
0xb87	RW	[7:4]: alc K2	0x0e
0xb89 ~ 0xb88	-	Reserved	-
0xb8a	RW	[0]: clk mode 1: usb mode 0: normal mode (not support) [5:1]: clk sr [6]: codec clk div2 [7]: codec clk en	0x00
0xb8b	RW	[0]: codec dec en	0x00
0xb9f ~ 0xb8c	-	Reserved	-
0xba0	RW	[3:0]: ALC ALCL [7:4]: ALC MAXGAIN	0x7b
0xba1	RW	[3:0]: ALC HLD	0x00
0xba2	RW	[3:0]: ALC ATK [7:4]: ALC DCY	0x32
0xba3	RW	[7:3]: ALC NGTH [2:1]: ALC NGG [0]: ALC NGAT	0x00
0xba4	RW	[2:0]: ALC MINGAIN	0x02

Address	R/W	Description	Default Value
Ana_0xe4	RW	[0]: TST2MIC [1]: TST2PGA [2]: TST2VMID [4:3]: RSVD	0x00
Ana_0xe5	RW	[0]: ASDMDEM_EN [1]: ASDM_DITHEN [3:2]: ASDM_DITH [4]: ASDM_DITHIN	0x00
Ana_0xe6	RW	[0]: CTR_IREF [1]: LVEN [3:2]: IBSEL [4]: INMUTE_PGA [5]: MUTE_PGA	0x00
Ana_0xe7	RW	[0]: PDBIAS [1]: PDPGABOOST [2]: PD_ASDM [3]: PD_INPPGA [4]: PD_PGABUF	0x1f
Ana_0xe8	RW	[3:0]: PGAVOL_IN [5:4]: VMDSCL	0x00

9.2 Audio Output Path

Audio output path mainly includes Rate Matching module, SDMDAC (Sigma-Delta Modulation DAC) and I2S Player. The audio data fetched from SRAM is processed by the Rate Matching module, then transferred to the SDM/I2S Player as the input signal.

Figure 9-5 Audio Output Path



9.2.1 Rate Matching

The rate matching block performs clock rate conversion and data synchronization between two domains: the input audio data is fetched from SRAM which works in system clock domain with 24 MHz/32 MHz/48 MHz clocks and the SDM/I2S which works between 4 MHz and 8 MHz.

When needed, the audio data from SRAM is interpolated to the SDM/I2S input rate. If the audio sampling rate is ClkUsbIn (e.g. 48 kHz), and the working clock of SDM/I2S is aclk_i, then the interpolation ratio is given as follows:

$$\frac{ClkUsbIn}{aclk_i} = \frac{step_i}{0x80000}$$

Where step_i[19:0] is configured in addresses 0x567 ~ 0x565.

Linear interpolation or delay interpolation is used as shown below.

Figure 9-6 Linear Interpolation

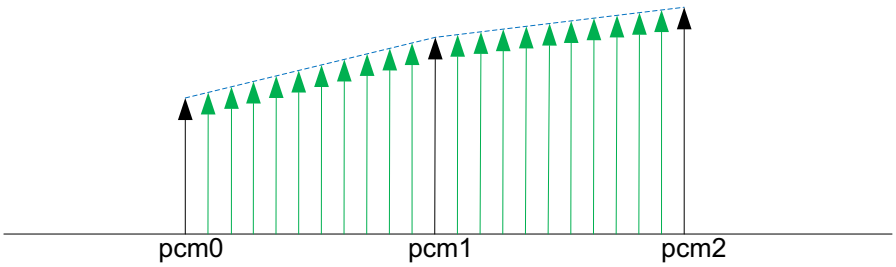
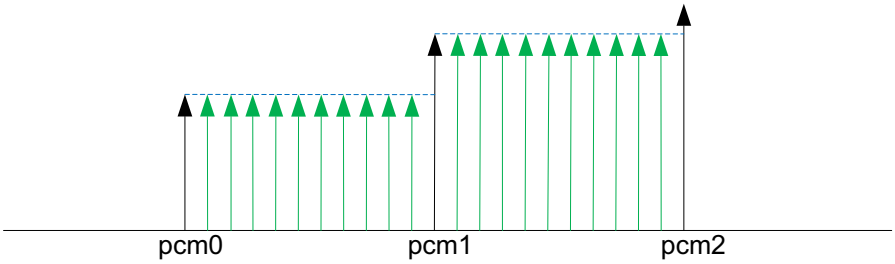


Figure 9-7 Delay Interpolation

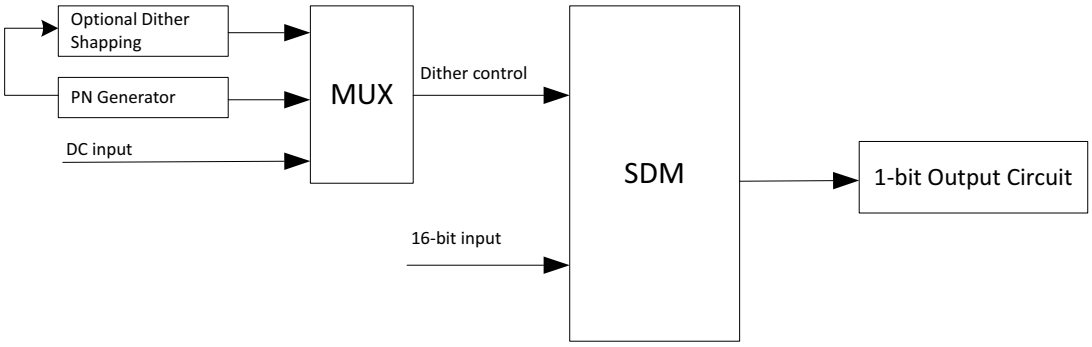


9.2.2 SDM

The SDM takes 16-bit audio data from SRAM and provides 1-bit modulated output. Only a simple passive filter network is needed to drive audio device directly.

Dither control can be added to the SDM to avoid spurs in output data. There are three dithering options: PN sequence, PN sequence with Shapping, and DC constant; only one type of input is allowed any time.

Figure 9-8 Block Diagram of SDM



9.2.3 Register Configuration

Address 0x560[4:1] should be set to "1" to enable I2S recorder/ISO player/SDM player/I2S player, while bit[0] is to select either mono or stereo audio output. Address 0x560[7] should be set to "1" to enable the HPF in audio output path.

Register VOL_CTRL (address 0x562) serves to adjust volume level.

Address 0x563[2] serves to select either linear interpolation or delay interpolation for the rate matching block: Setting bit[2] to "1" is to select linear interpolation, while clearing the bit is to select delay interpolation.

Input for SDM Dither control is selectable via addresses 0x56b[6:5]), 0x563[6:5] and 0x568 ~ 0x569.

For the left channel:

1. Address 0x56b[5] should be set to 1b'1 to select constant DC input. When DC input is used, addresses 0x56c ~ 0x56d serve to configure the input constant value.
2. Address 0x56b[5] should be set to 1b'0 to use PN generator. Address 0x563[5] serves to enable/mask dither shapping module. There are two PN generators to generate random dithering sequence; address 0x568 bit[6]/bit[5] is enabling bit of the two PN generators.
   - To select PN sequence as input, address 0x56b[5] and 0x563[5] should be set to 0, 0x568[6]/[5]/[6:5] should be set to 1.
   - To select PN sequence with Shapping as input, address 0x56b[5] should be set to 0, 0x563[5] and 0x568[6]/[5]/[6:5] should be set to 1.

When PN sequence or PN with Shapping is used, address 0x568[4:0]/0x569[4:0] determines the number of bits (ranging from 0 to 16) used in PN1/PN2 generator.

For the right channel:

1. Address 0x56b[6] should be set to 1b'1 to select constant DC input. When DC input is enabled, addresses 0x56e ~ 0x56f serve to configure the input constant value.
2. Address 0x56b[6] should be set to 1b'0 to use PN generator. Address 0x563[6] serves to enable/mask dither shapping module. There are two PN generators to generate random dithering sequence; address 0x569 bit[6]/bit[5] is enabling bit of the two PN generators.
   - To select PN sequence as input, address 0x56b[6] and 0x563[6] should be set to 0, 0x569[6]/[5]/[6:5] should be set to 1.
   - To select PN sequence with Shapping as input, address 0x56b[6] should be set to 0, 0x563[6] and 0x569[6]/[5]/[6:5] should be set to 1.

When PN sequence or PN with Shapping is used, address 0x56a[4:0]/0x56b[4:0] determines the number of bits (ranging from 0 to 16) used in PN1/PN2 generator.

Address 0x567, 0x566 and 0x565[7:4] are to set step_i[19:0] for the rate matching block, while address 0x564 is to tune the step_i value. The step_i should be in accordance with the aclk_i provided by SDM/I2S clock.

Table 9-4 Register Configuration Related to Audio Output Path

Address	R/W	Description	Default Value
0x560	RW	[0]: 1 - mono mode audio output, 0 - stereo mode audio output [1]: 1 - enable I2S player, 0 - disable I2S player [2]: 1 - enable SDM player, 0 - disable SDM player [3]: 1 - enable ISO player, 0 - disable ISO player [4]: 1 - enable I2S recorder, 0 - disable I2S recorder [5]: 1 - enable interface of I2S, 0 - disable interface of I2S [6]: 1 - enable GRP, 0 - disable GRP [7]: 1 - enable HPF, 0 - disable HPF	0x04
0x561	RW	[7:0]: Middle of GRP	0x40
0x562	RW	[0]: Add a quarter [1]: Add a half [6:2]: shift left [7]: 1 - mute, 0 - normal	0x40
0x563	RW	[0]: 1 - not multiply 2 when PWM, 0 - mutiply 2 [1]: 1 - PWM, 0 - not PWM [2]: 1 - linear interpolate, 0 - delay interpolate [4:3]: Reserved [5]: 1 - left Shapping used, 0 - left Shapping not used [6]: 1 - right Shapping used, 0 - right Shapping not used [7]: I2S input left/right channel swap	0x64
0x564	RW	[7:0]: tune step_i for rate matching block	0x01
0x565	RW	[3:0]: factor to generate I2S clock [7:4]: low 4 bits of rate matching block step_i[3:0]	0x90
0x566	RW	[7:0]: middle byte of rate matching block step_i[11:4]	0xc4
0x567	RW	[7:0]: high byte of rate matching block step_i[19:12]	0x00

Address	R/W	Description	Default Value
0x568	RW	[4:0]: bits used in pn1 of left channel, range from 0 to 16 [5]: 1 - pn2 of left enable, 0 - pn2 of left disable [6]: 1 - pn1 of left enable, 0 - pn1 of left disable	0x50
0x569	RW	[4:0]: bits used in pn2 of left channel, range from 0 to 16 [5]: 1 - pn2 of right enable, 0 - pn2 of right disable [6]: 1 - pn1 of right enable, 0 - pn1 of right disable	0x40
0x56a	RW	[4:0]: bits used in pn1 of right channel, range from 0 to 16 [5]: 1 - exchange data in between SDMs, 0 - not exchanged	0x10
0x56b	RW	[4:0]: bits used in pn2 of right channel, range from 0 to 16 [5]: 1 - left channel use const value, 0 - left channel use pn [6]: 1 - right channel use const value, 0 - right channel use pn	0x00
0x56c	RW	[7:0]: low byte of left channel const, i.e, const_l[7:0]	0x00
0x56d	RW	[7:0]: high byte of left channel const, i.e.const_l[15:8]	0x00
0x56e	RW	[7:0]: low byte of right channel const, i.e const_r[7:0]	0x00
0x56f	RW	[7:0]: high byte of right channel const, i.e const_r[15:8]	0x00

10 Quadrature Decoder

The TLSR8270 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 PortD, PortC, PortB and PortA 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

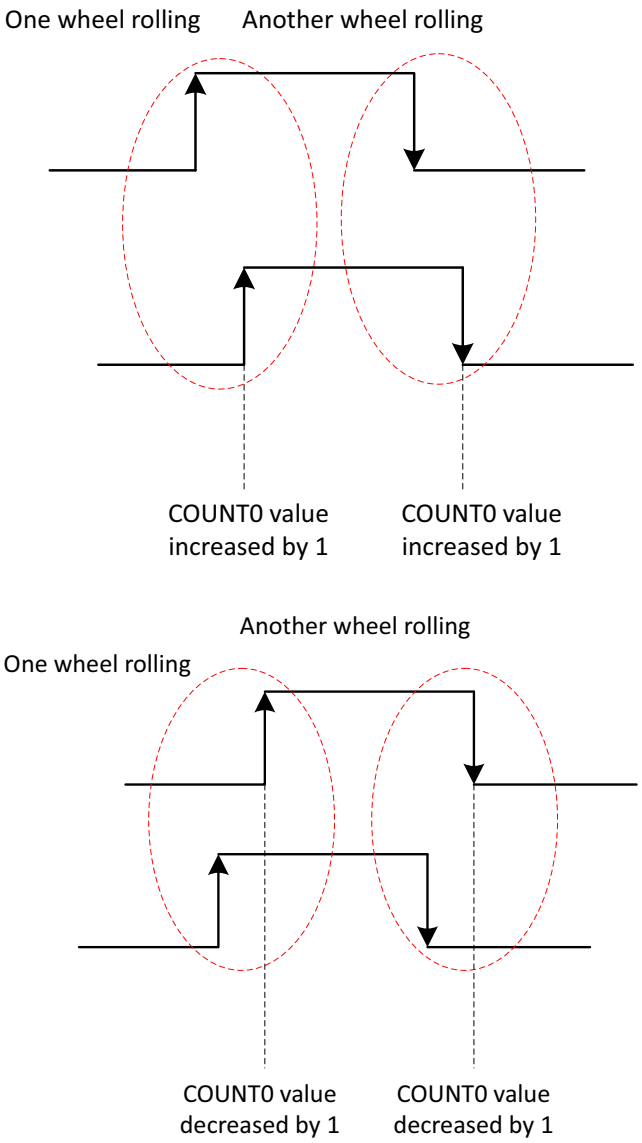
The QDEC embeds an internal hardware counter, which is not connected with bus.

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.

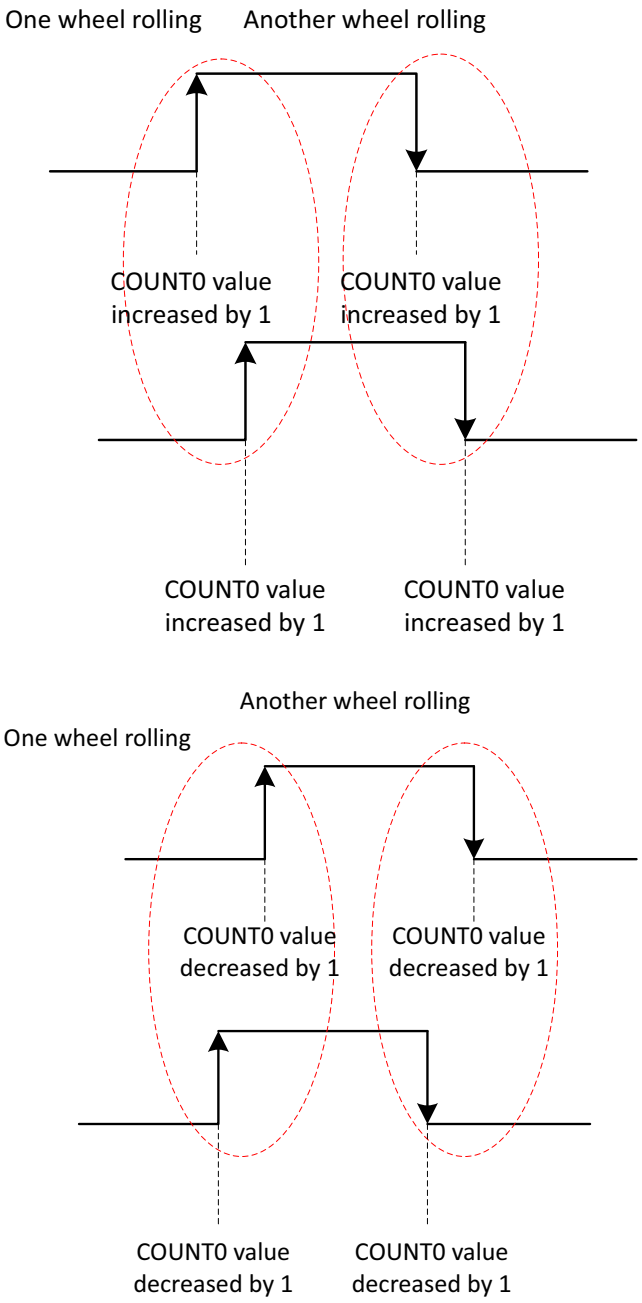
If address 0xd7[0] is cleared to select common mode, the QDEC Counter value (real time counting value) is increased/decreased by 1 only when the same rising/falling edges are detected from the two phase signals.

Figure 10-1 Common Mode



If address 0xd7[0] is set to 1b'1 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

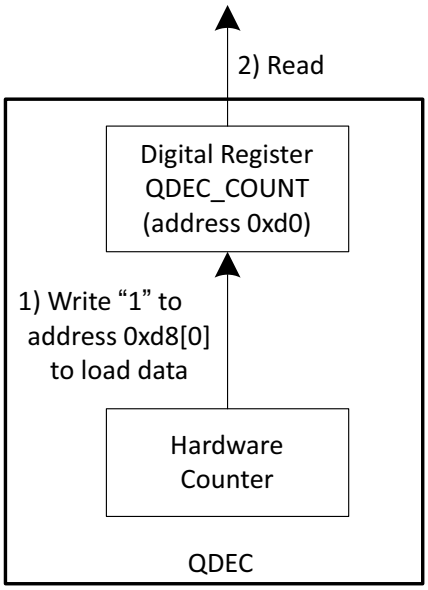


10.3 Read Real Time Counting Value

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 1b'1 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.

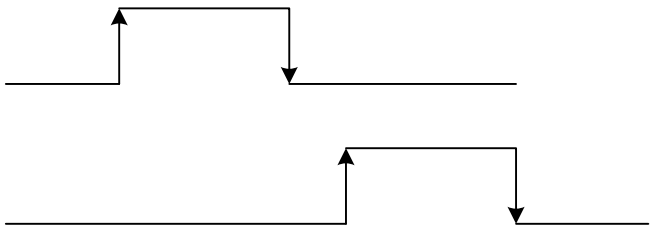
10.5 Other Configuration

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

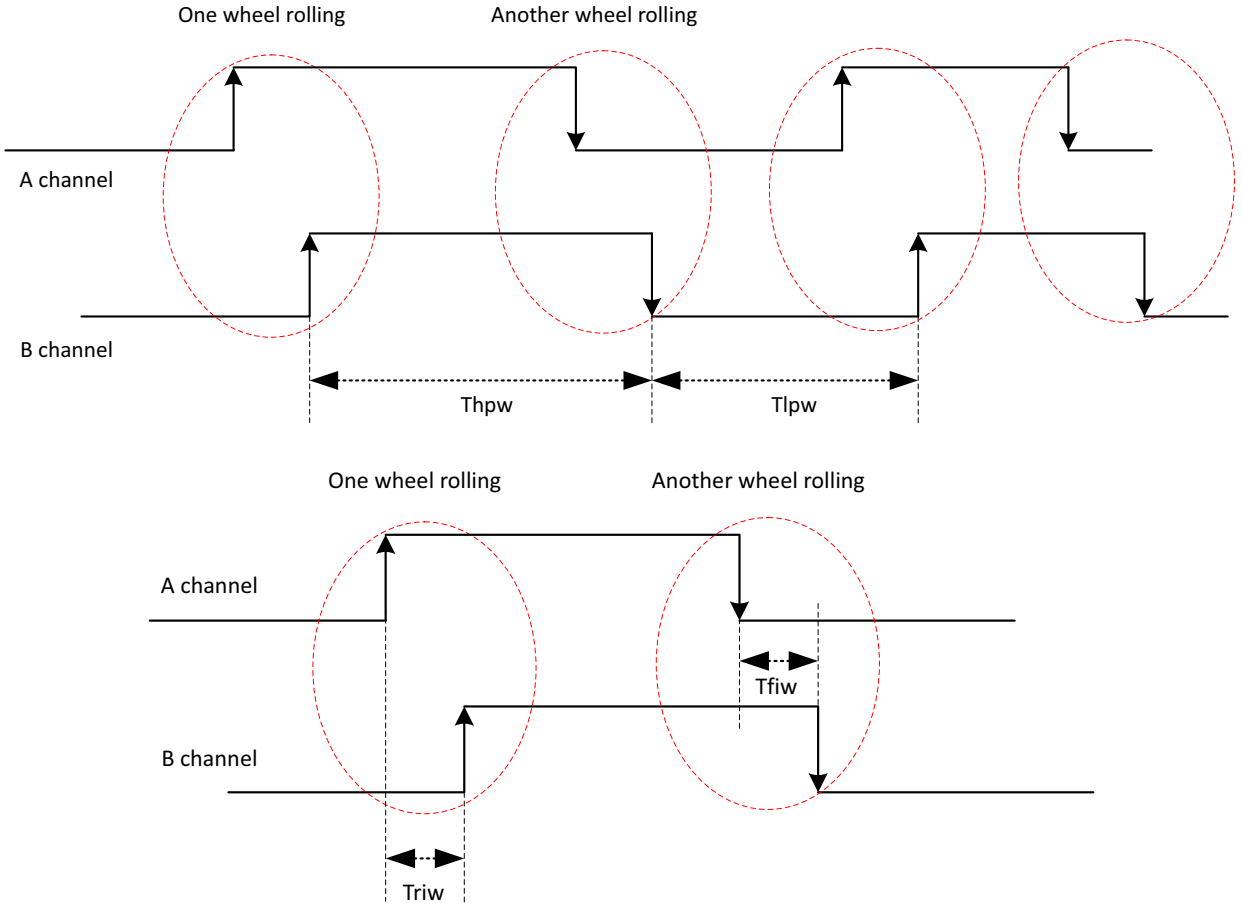


Table 10-2 Timing

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	[2:0]: filter time (can filter 2^n *clk_32k*2 width deglitch) [4]: pola, input signal pola 0 - no signal is low, 1 - no signal is high [5]: shuttle mode 1 - enable shuttle mode	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	RW	[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 Manchester Decoder

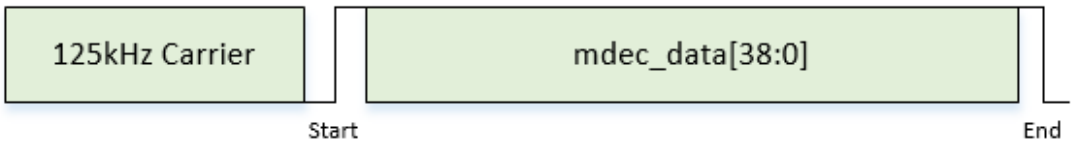
The TLSR8270 integrates one Manchester Decoder (MDEC). The MDEC is designed to decode the input Manchester code, data after Manchester coding, into binary data.

11.1 Frame Format

The MDEC's input sequence includes a Carrier signal, a Start flag, a 39-bit mdec_data filed (mdec_data[38:0]), and an End flag.

- Carrier signal duration should be no less than 3 ms.
- Support duty cycle of 50% ~ 90%.
- Period for each bit is 408 μs.
- The Start flag is Manchester code 1, a positive edge from low level to high level.
- The End flag is Manchester code 0, a negative edge from high level to low level.

Figure 11-1 Frame Format



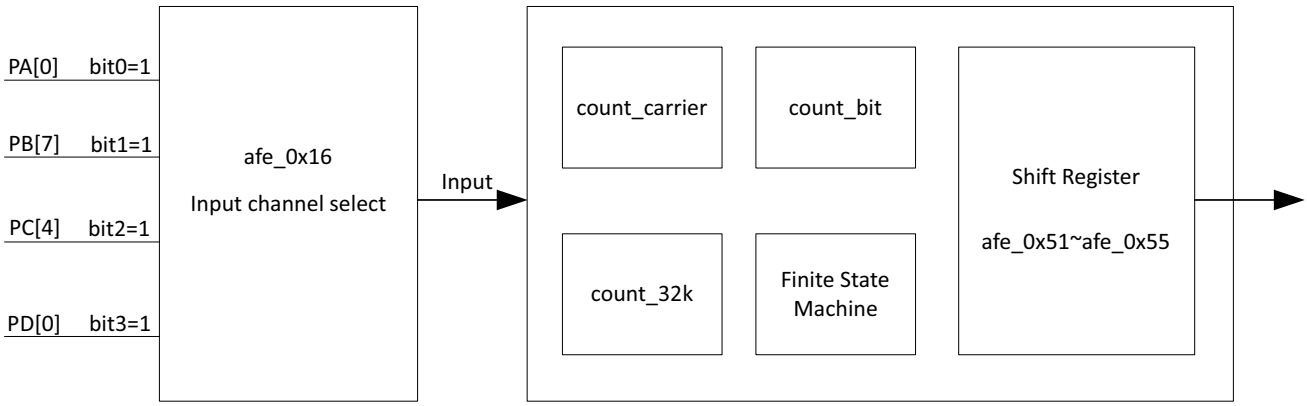
11.2 Function Description

11.2.1 Block Diagram

The MDEC uses 32 kHz clock, and it mainly embeds a finite State machine, three counters, and a Shift Register to implement its function, including:

- Finite State Machine: It includes Idle state, Carrier state, Start state, Data state, and End state.
- count_carrier: This counter serves to detect carrier signal in Idle state. When a carrier signal is detected, the MDEC's state machine enters Start state.
- count_32k: After entering Start state, this counter serves to calculate the interval between two adjacent positive edges, so as to judge the input data.
- count_bit: This counter serves to record the number of bits that have been decoded, so as to judge whether data decoding of a frame is finished. When the bit number reaches 39, it indicates decoding is finished.
- Shift Register: This register serves to store binary data after decoding.

Figure 11-2 Function Block Diagram



11.2.2 Reset MDEC

The analog register afe_0x16 bit[4] serves to reset the MDEC module. To use the MDEC, it's needed to set this bit as 1b'0.

11.2.3 Select Input Channel

User can input the Manchester code from specific GPIO pin into the MDEC.

The analog register afe_0x16 bit[3] serves to select PD[0], PC[4], PB[7] and PA[0] as input channel, respectively.

11.2.4 Read Result Data

Data after decoding, mdec_data[38:0], are available in the Shift Register, i.e. the analog registers afe_0x51 ~ afe_0x55.

After data decoding of a frame is finished, if the 4-bit mdec_data[38:35] in the analog register afe_0x51[7:4]) is consistent with the mdec_match_value written in the analog register afe_0x17[3:0], a MCU wakeup signal will be generated.

11.3 Register Table

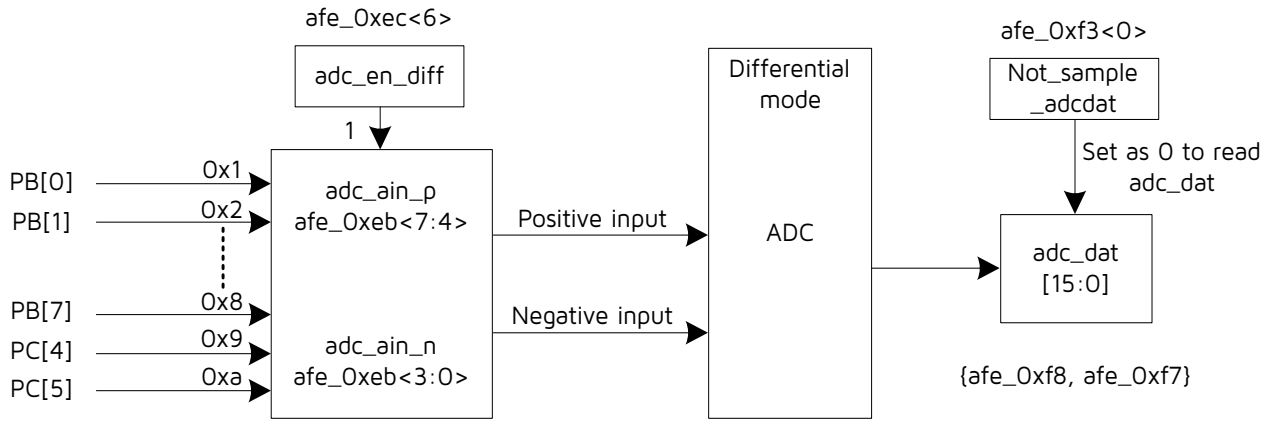
Table 11-1 Analog Registers for MDEC

Address	Bit Range	R/W	Description	Default Value
afe_0x16	[4]	RW	Reset MDEC 1: Reset MDEC and clear MDEC wakeup status (afe_0x44[4]); to use MDEC, please set as 0.	0x1
	[3]	RW	Select PD[0] as data input	0x0
	[2]	RW	Select PC[4] as data input	0x0
	[1]	RW	Select PB[7] as data input	0x0
	[0]	RW	Select PA[0] as data input	0x0
afe_0x17	[3:0]	RW	mdec_match_value	0x2
afe_0x44	[4]	R	MDEC wakeup status	-
afe_0x51	[7:4]	R	mdec_data[38:35]	-
	[3]	R	RSVD	-
	[2:0]	R	mdec_data[34:32]	-
afe_0x52	[7:0]	R	mdec_data[31:24]	-
afe_0x53	[7:0]	R	mdec_data[23:16]	-
afe_0x54	[7:0]	R	mdec_data[15:8]	-
afe_0x55	[7:0]	R	mdec_data[7:0]	-

12 SAR ADC

The TLSR8270 integrates one SAR ADC module, which can be used to sample analog input signals such as battery voltage and temperature sensor.

Figure 12-1 Block Diagram of ADC



12.1 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 1b'0.

12.2 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>).

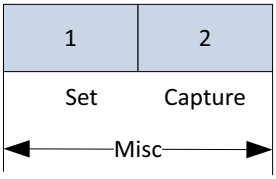
$$ADC\ clock\ frequency\ (marked\ as\ F_{ADC\_clk}) = 24\ MHz/(adc\_clk\_div+1)$$

12.3 ADC Control in Auto Mode

12.3.1 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>).

12.3.2 "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 1b'1 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.

12.3.3 "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 1b'1 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 1b'0 (default). If the afe_0xf3<0> is set as 1b'1, 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.*

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

12.3.4 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 1b'1, 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*

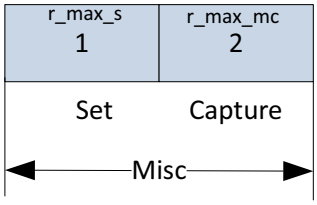


Table 12-1 Overall Register Setting

Function	Register Setting
Power on the ADC	afe_0xfc<5> = 1b'0
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> = 1b'1
Set the max state index as "2"	afe_0xf2<5:4> = 2b'10
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

Function	Register Setting
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
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

12.4 Register Table

Table 12-2 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	-

Address	Description	Default Value
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: C[5] 0xb: RSVD 0xc: RSVD 0xd: tempsensor_n (Temperature sensor negative output) 0xe: Ground 0xf: Ground	0000
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: C[5] 0xb: RSVD 0xc: RSVD 0xd: tempsensor_p (Temperature sensor positive output) 0xe: RSVD 0xf: RSVD	0000

Address	Description	Default Value
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	-
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	-

Address	Description	Default Value
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	-
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

13 Temperature Sensor

The TLSR8270 integrates a temperature sensor and it's used in combination with the SAR ADC to detect real-time temperature.

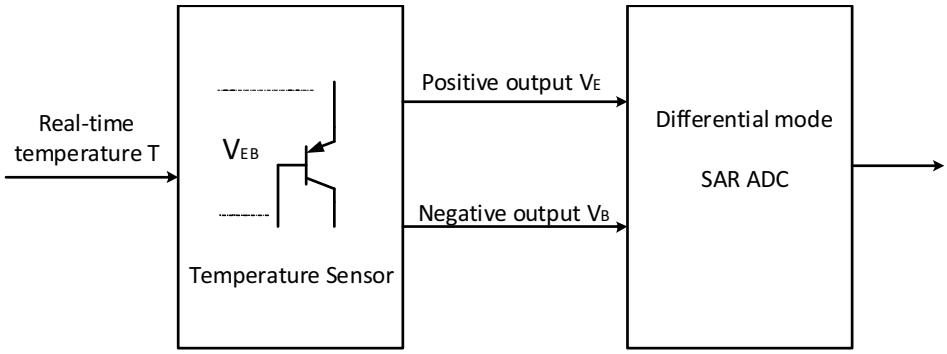
The temperature sensor is disabled by default. The analog register afe_0x06<2> should be set as 1b'0 to enable the temperature sensor.

Table 13-1 Analog Register for Temperature Sensor

Address	Name	Description	Default Value
afe_0x06<2>	pd_temp_sensor_3V	Power on/down temperature sensor: 0: Power up 1: Power down	1

The temperature sensor embeds a pnp transistor. It takes the real-time temperature (T) as input, and outputs voltage drop (V_{EB}) signals of pnp transistor as positive and negative output respectively.

Figure 13-1 Block Diagram of Temperature Sensor



The voltage drop V_{EB} signals is determined by the real-time temperature T, as shown below:

$$V_{EB} = 884\ mV - 1.4286\ mV/℃ * (T - (-40℃))$$
$$= 884\ mV - 1.4286\ mV/℃ * (T + 40℃)$$

In this formula, "884 mV" indicates the value of VEB at the temperature of "-40°C".

To detect the temperature, the positive and negative output of the temperature sensor should be enabled as the input channels of the SAR ADC. The ADC will convert the V_{EB} signals into digital signal.

The ADC should be configured as differential mode, and the positive and negative output of the temperature sensor should be configured as differential input of the ADC. The ADC should initiate one operation and obtain one output signal (ADCOUT); therefore,

$$V_{EB} = \frac{ADCOUT}{2^N - 1} \times V_{REF}$$

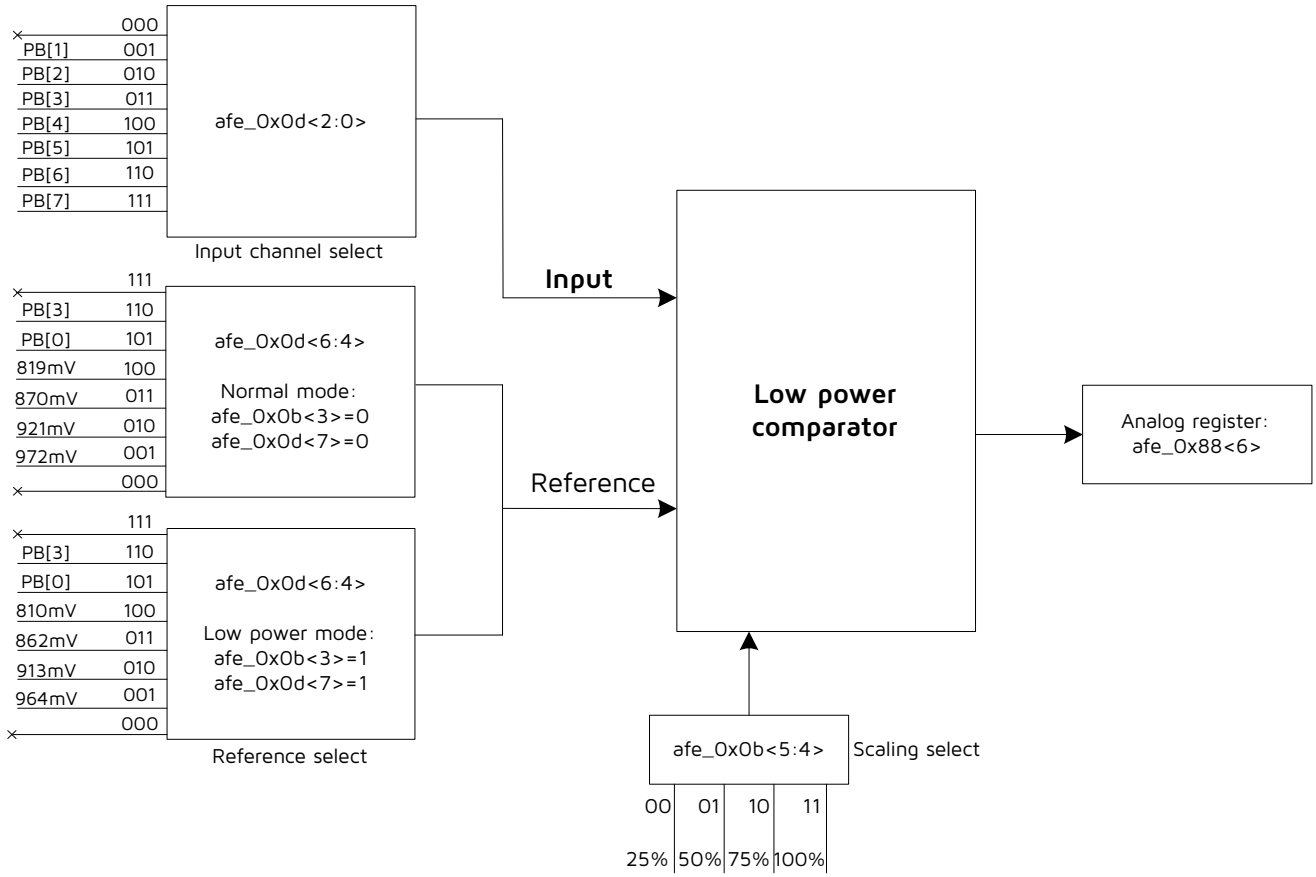
In the formula, "N" and "V_{REF}" indicate the selected resolution and reference voltage of the SAR ADC. Then the real-time temperature T can be calculated according to the V_{EB}.

14 Low Power Comparator

The TLSR8270 embeds a low power comparator. This comparator takes two inputs: input derived from external PortB (PB[1] ~ PB[7]), and reference input derived from internal reference, PB[0], PB[3] or float.

By comparing the input voltage multiplied by selected scaling coefficient with reference input voltage, the low power comparator will output high or low level accordingly.

Figure 14-1 Block Diagram of Low Power Comparator



14.1 Power On/Down

The low power comparator is powered down by default.

The analog register afe_0x06<1> serves to control power state of the low power comparator: By clearing this bit, this comparator will be powered on; by setting this bit to 1b'1, this comparator will be powered down.

To use the low power comparator, first set afe_0x06<1> as 1b'0, then the 32K RC clock source is enabled as the comparator clock.

14.2 Select Input Channel

Input channel is selectable from the PortB (PB[1] ~ PB[7]) via the analog register afe_0x0d<2:0>.

14.3 Select Mode and Input Channel for Reference

Generally, it's needed to clear both the afe_0x0b<3> and afe_0x0d<7> to select the normal mode. In normal mode, the internal reference is derived from UVLO and has higher accuracy, but current bias is larger (10 μA); reference voltage input channel is selectable from internal reference of 972 mV, 921 mV, 870 mV and 819 mV, as well as PB[0], PB[3] and float.

To select the low power mode, both the afe_0x0b<3> and afe_0x0d<7> should be set as 1b'1. In low power mode, the internal reference is derived from Bandgap and has lower accuracy, but current bias is decreased to 50 nA; reference voltage input channel is selectable from internal reference of 964 mV, 913 mV, 862 mV and 810 mV, as well as PB[0], PB[3] and float.

14.4 Select Scaling Coefficient

Equivalent reference voltage equals the selected reference input voltage divided by scaling coefficient. The analog register afe_0x0b<5:4> serves to select one of the four scaling options: 25%, 50%, 75% and 100%.

14.5 Low Power Comparator Output

The low power comparator output is determined by the comparison result of the value of [input voltage *scaling] and reference voltage input. The comparison principle is shown as below:

- If the value of [input voltage *scaling] is larger than reference voltage input, the output will be low ("0").
- If the value of [input voltage *scaling] is lower than reference voltage input, the output will be high ("1").
- If the value of [input voltage *scaling] equals reference voltage input, or input channel is selected as float, the output will be uncertain.

User can read the output of the low power comparator via the analog register afe_0x88<6>.

The output of the low power comparator can be used as signal to wakeup system from low power modes.

14.6 Register Table

Table 14-1 Analog Register Table Related to Low Power Comparator

Address	Description	Default Value
afe_0x06<1>	Power on/down low power comparator: 0: Power up 1: Power down	1
afe_0x0b<3>	Reference mode select: 0: Normal mode (current bias 10 μA) 1: Low power mode (current bias 50 nA) See afe_0x0d<7>.	1

Address	Description	Default Value
afe_0x0b<5:4>	Reference voltage scaling: 00: 25% 01: 50% 10: 75% 11: 100%	01
afe_0x0d<2:0>	Input Channel select: 000: RSVD 001: B[1] 010: B[2] 011: B[3] 100: B[4] 101: B[5] 110: B[6] 111: B[7]	000
afe_0x0d<3>	Reserved	0
afe_0x0d<6:4>	Reference select: Normal mode Low power mode 000: Float 000: Float 001: 972 mV 001: 964 mV 010: 921 mV 010: 913 mV 011: 870 mV 011: 862 mV 100: 819 mV 100: 810 mV 101: B[0] 101: B[0] 110: B[3] 110: B[3] 111: Float 111: Float	000
afe_0x0d<7>	Enable or disable 10 μA current bias: 0: Enable 10 μA current bias 1: Disable 10 μA current bias	1

15 AES

The TLSR8270 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.

15.1 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 0x550 ~ 0x55f).
- Set operation method of AES module via register AES_CTRL: set address 0x540[0] as 1b'1 for decryption method, while clear this bit for encryption method.
- For encryption method, write registers AES-DAT0 ~ AES-DAT3 (address 0x548~0x54b) for four times to set the 128-bit plaintext. After encryption, the 128-bit ciphertext can be obtained by reading address 0x548 ~ 0x54b for four times.
- For decryption method, write registers AES-DAT0 ~ AES-DAT3 (address 0x548 ~ 0x54b) for four times to set the 128-bit ciphertext. After decryption, the 128-bit plaintext can be obtained by reading address 0x548 ~ 0x54b for four times.
- Address 0x540 bit[1] and bit[2] are read only bits: bit[1] will be cleared automatically after quartic writing of address 0x548 ~ 0x54b; bit[2] will be set as 1 automatically after encryption/decryption, and then cleared automatically after quartic reading of address 0x548 ~ 0x54b.

15.2 DMA Mode

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

15.3 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 0x540[7] is set as 1b'1 to enable AES-CCM mode, AES output will also take previous encryption and decryption result into consideration.

15.4 Register Table

Table 15-1 Register Table Related to AES

Address	R/W	Description	Default Value
0x540	RW	[0] Select decrypt/encrypt 1: decrypt, 0: encrypt [1] Read-only 1: input data needed, 0: input data ready [2] Read-only 0: output data not ready, 1: output data ready [7] 1: enable AES-CCM mode	0x02
0x548	RW	Byte 0 of input/output data	0x00
0x549	RW	Byte 1 of input/output data	0x00
0x54a	RW	Byte 2 of input/output data	0x00
0x54b	RW	Byte 3 of input/output data	0x00
0x550	RW	[7:0] KEY0	0x00
0x551	RW	[7:0] KEY1	0x00
0x552	RW	[7:0] KEY2	0x00
0x553	RW	[7:0] KEY3	0x00
0x554	RW	[7:0] KEY4	0x00
0x555	RW	[7:0] KEY5	0x00
0x556	RW	[7:0] KEY6	0x00
0x557	RW	[7:0] KEY7	0x00
0x558	RW	[7:0] KEY8	0x00
0x559	RW	[7:0] KEY9	0x00
0x55a	RW	[7:0] KEY10	0x00
0x55b	RW	[7:0] KEY11	0x00
0x55c	RW	[7:0] KEY12	0x00
0x55d	RW	[7:0] KEY13	0x00
0x55e	RW	[7:0] KEY14	0x00
0x55f	RW	[7:0] KEY15	0x00

16 Public Key Engine

The TLSR8270 embeds Public Key Engine Standard Performance acceleration module and this section describes its function and use.

16.1 Calculation Model Overview

PKE (Public Key Engine) is specifically designed to accelerate large digital-to-analog operations in public key cryptographic operations. PKE SP-ECC is a version optimized for the elliptic curve algorithm. In this version, the following features are available.

- Support different bit width ECC (prime field): 192, 256 bits
- Support curve parameters: NIST p192, NIST p256, X25519, EdDSA

16.2 Function Description

16.2.1 Module Description

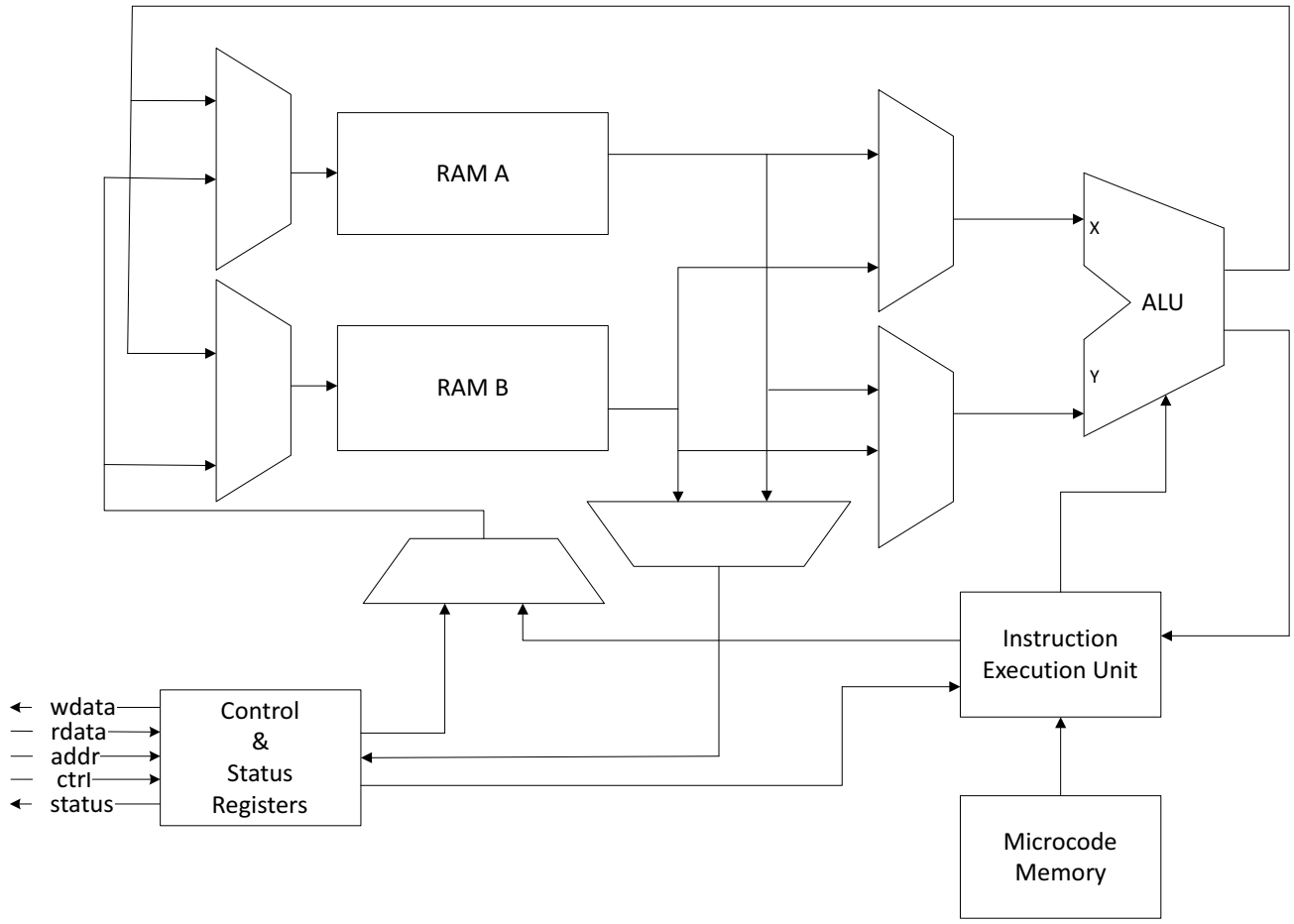
There are a large number of large digital-to-analog operations in public key cryptographic operations. PKE is designed to accelerate large digital-to-analog operations involved in RSA and Elliptic Curve Cryptography (ECC) operations in public key cryptography. Recently PKE can directly complete modular exponentiation in RSA and point multiplication in ECC. The CPU can query the operation of the PKE by polling or interrupting. The PKE includes one program memory unit (ROM), one instruction arithmetic unit (IEU), one 32-bit arithmetic unit (ALU), two pseudo-double-ended data RAMs, one register combination with interface module.

According to different register configurations, PKE can perform the following operations with different precisions:

- ECC (Prime field): 192 and 256 bits

In addition, the calculation of the PKE is finished in the form of Microcode and the Microcode is stored in the program storage unit. Therefore, different kind of public key cryptographic calculations can be implemented by pouring different microcode into the program storage unit. For instance, a high security public key algorithm instruction can be injected into a program storage unit in the PKE module in a SoC with high security requirements. Certainly these arithmetic instructions can be written to the ROM with a large program memory unit capacity. The CPU makes real-time calls according to different usage scenarios. The full microcode size is approximately 2 KB.

Figure 16-1 Block Diagram of PKE SP Module



16.2.2 Software Interface (Programming Model)

The interfaces of the PKE SP are all mapped into the 7 KB address space. The block of address mapping space mainly contains all the operands that the CPU can access. These operands contain modulus, power exponents, partial intermediate variables, and so on. In addition to this, the address map also contains control and status registers. The CPU can configure and monitor the PKE module through these control and status registers.

In the operations supported by PKE, the operands are also 192 bits at minimum. Therefore, it will encounter the problem of big-endian and little-endian when putting data into data RAM in the CPU or DMA. In the PKE module, words are arranged following an order of little-endian.

In PKE, the smallest operand is 32 bits (1 word), because the current ALU bit width input is 32 bits. If the operand is not word aligned, the high bit needs to be filled as 0.

After the PKE receives the start command, it starts the operation. During the operation, the host computer can query the current running state through the status register, or interrupt the current operation through the control register. In addition, the result of partial intermediate operations can be obtained by accessing the data RAM address.

The host computer can obtain the result of target operation finish by PKE through polling or interrupting. Data RAM supports word aligned and does not support byte alignment.

Table 16-1 Dual Port RAM Address Map

First Address of Operand	ECC		
	256 Bits	512 Bits	1024 Bits
A0	0x0400	0x0400	0x0400
A1	0x0424	0x0444	0x0484
A2	0x0448	0x0488	0x0508
A3	0x046C	0x04CC	0x058C
A4	0x0490	0x0510	0x0610
A5	0x04B4	0x0554	0x0694
A6	0x04D8	0x0598	0x0718
A7	0x04FC	0x05DC	0x079C
A8	0x0520	0x0620	0x0820
A9	0x0544	0x0664	0x08A4
B0	0x1000	0x1000	0x1000
B1	0x1024	0x1044	0x1084
B2	0x1048	0x1088	0x1108
B3	0x106C	0x10CC	0x118C
B4	0x1090	0x1110	0x1210
B5	0x10B4	0x1154	0x1294
B6	0x10D8	0x1198	0x1318
B7	0x10FC	0x11DC	0x139C
B8	0x1120	0x1220	0x1420
B9	0x1144	0x1264	0x14A4

The above table shows the address assignment of two RAMs in ECC mode. The operand registers are distributed in two blocks of data RAM, using the prefixes A and B to distinguish the two blocks of RAM. The addresses listed in the table are all CPU addressable addresses, RAM A has an address offset of 0x400, and RAM B has an address offset of 0x1000. The actual space used by RAM will be larger than the space listed in the table and some intermediate variable storage is not open to the CPU.

Data will be stored in the mode of little-endian in RAM.

16.3 Register Description

Table 16-2 Register Map

Address	R/W	Description	Default Value
0x2000	W1S	[0] Go Start signal. When write 1 to the byte, the PKE will start running in the next clock cycle. The operation of the PKE is based on the configuration of the control registers and data registers for that clock cycle written as 1. [7:1] RSVD	0x00
0x2001	R	[15:8] RSVD	0x00
0x2002	W1S	[16] Stop Stop signal. When write 1 to the byte, PKE will stop in the next clock cycle. [23:17] RSVD	0x00
0x2003	R	[31:24] RSVD	0x00
0x2004	R	[7:0] RSVD	0x00
0x2005	RW	[8] IRQEN Interrupt enable. When the bit is set as 1, the o_irq interface is valid. Regardless of whether the bit is set as 1, the STAT register is not affected by it. [15:9] RSVD	0x00
0x2006	RW	[23:16] Partial_Radix Select part of BASE_RADIX to determine the bit width that the operation really needs to use during the operation. The value of this field indicates the number of words, and the bit width of the operand is PARTICAL_RADIX*32 bits. For example, if BASE_RADIX = 2, PARTIAL_RADIX = 6, then the bit width of the operand is (6/ (256/32))*256 = 192. If the operations of ECC-192 need to be processed, BASE_RADIX and PARTIAL_RADIX should be configured as shown in this example. When using operands of other bit widths, configure BASE_RADIX and PARTIAL_RADIX according to the above formula.	0x00

Address	R/W	Description	Default Value
0x2007	RW	[31:27] RSVD [26:24] Base_Radix This field indicates the bit width cardinality at which the operation is performed. At the same time, the cardinality also represents the space required for the operand to be stored in the data RAM. For ECC point operations, the value of this field should be 2. 2: 256 bits Others: Reserved	0x02
0x2010	RW	[7:0] ADDR This field indicates the address of the next instruction to be executed by the PKE. This register can only be rewritten when the PKE is not working. Any write operation while the PKE is operating will be ignored. This field is also updated in real time when running the PKE and always pointing to the address of the instruction that will be executed next. Therefore, this register can also be combined with CTRL.STOP for debugging. It should be noted that the instructions are all word aligned. Therefore, the lowest 2 bits of the field are 0. When writing an instruction address to this field, it is limited to the address range of 0x00 ~ 0x2F. The written address will proceed "And" Operation with a mask, therefore ignoring the upper 6 bits.	0x00
0x2011	RW	[11:8] ADDR See above description for [7:0] [15:12] RSVD	0x00
0x2012	R	[23:16] RSVD	0x00
0x2013	R	[31:24] RSVD	0x00

Address	R/W	Description	Default Value
0x2020	W1C	[0] Done When the bit is set to 1, it indicates that the operation ends. When this bit is set as 1 from external, the bit is cleared. In addition, this bit also acts as a clear bit for the external interrupt. When the bit is high as CTRL.IRQEN is active, the external interrupt signal is also pulled high. To write 1 from external, the external interrupt is also cleared. [7:1] RSVD	0x00
0x2021	R	[15:8] RSVD	0x00
0x2022	R	[23:16] RSVD	0x00
0x2023	R	[31:24] RSVD	0x00
0x2024	R	[3:0] STOP_LOG This field is used to indicate the reason when the PKE stopped. If the PKE stopped because the operation is complete, the value of this field is 0. If the value of this field is non-zero, then it proves that PKE operation is not completed, maybe encountering some exceptions and then external processing is required, the result is not available. 0: Normal stop 1: Received a termination request (CTRL.STOP is high) 2: No valid modulo inverse 3: Point is not on the curve (CTRL.CMD:PVER) 4: Invalid Microcode Others: Reserved [7:4] RSVD	0x00
0x2025	R	[15:8] RSVD	0x00
0x2026	R	[23:16] RSVD	0x00
0x2027	R	[31:24] RSVD	0x00

Address	R/W	Description	Default Value
0x2050	RW	[0] IAFF_R0 The input form of R0 is affine coordinate system enabled, this bit is only valid for ECC operations. When the bit is high, the input point is a point on the affine coordinate system. When the bit is low, the input point is a point on the Jacobian coordinate system. When it comes to the modular multiplication, if the bit is low, it will first convert the number on its scope to the Jacobian coordinate system before computing. [1] IMON_R0 The input form of R0 is Montgomery enabled. When the bit is high, data will be input in the form of Montgomery. When the bit is low, data will be input in the normal form. When it comes to the modular multiplication, if the bit is low, it will first convert the number on its scope to the Montgomery form before computing. [2] IAFF_R1 The input form of R1 is affine coordinate system enabled, this bit is only valid for ECC operations. When the bit is high, the input point is a point on the affine coordinate system. When the bit is low, the input point is a point on the Jacobian coordinate system. When it comes to the modular multiplication, if the bit is low, it will first convert the number on its scope to the Jacobian coordinate system before computing. [3] IMON_R1 The input form of R1 is Montgomery enabled. When the bit is high, data will be input in the form of Montgomery. When the bit is low, data will be input in the normal form. When it comes to the modular multiplication, if the bit is low, it will first convert the number on its scope to the Montgomery form before computing.	0x2a

Address	R/W	Description	Default Value
		[4] OAFF The output form is affine coordinate system enabled, this bit is only valid for ECC operations. When the bit is high, the output point is a point on the affine coordinate system. When the bit is low, the output point is a point on the Jacobian coordinate system. [5] OMON The output form is Montgomery enabled. When the bit is high, the output is in the form of Montgomery. When the bit is low, the output is in the normal form. [7:6] RSVD	
0x2051	RW	[9:8] ME_SCA_EN The secure modular exponentiation algorithm selects a signal that is valid only for modular exponentiation in RSA operations. 00: The secure modular exponentiation algorithm requires a public key and a private key. The exponentiation index is register B1 under the algorithm. 01: The secure modular exponentiation algorithm requires a private key. The exponentiation index is register B1 under the algorithm. 10: Montgomery stepwise modular exponentiation algorithm. 11: Non-secure modular exponentiation requires a public key. Under this algorithm, the exponentiation index is register A1. Among them, the decryption and signature of the RSA can only use the secure modular exponentiation algorithm. There are two different algorithms of selecting 01 or 10 according to whether using a public key. For RSA encryption and verification, non-secure modular exponentiation algorithm can be used. [15:10] RSVD	0x00
0x2052	R	[23:16] RSVD	0x00
0x2053	R	[31:24] RSVD	0x00

Address	R/W	Description	Default Value
0x2080	R	[3:0] MIR Secondary version number [7:4] MAR Main version number	0x00
0x2081	R	[15:8] RSVD	0x00
0x2082	R	[23:16] PROJECT Project number	0x00
0x2083	R	[31:24] PROJECT Project number	0x00
0x2400 ~ 0x2E10	RW	[31:0] DATA_A This field is used to store operational data.	-
0x3000 ~ 0x4A10	RW	[31:0] DATA_B This field is used to store operational data.	-

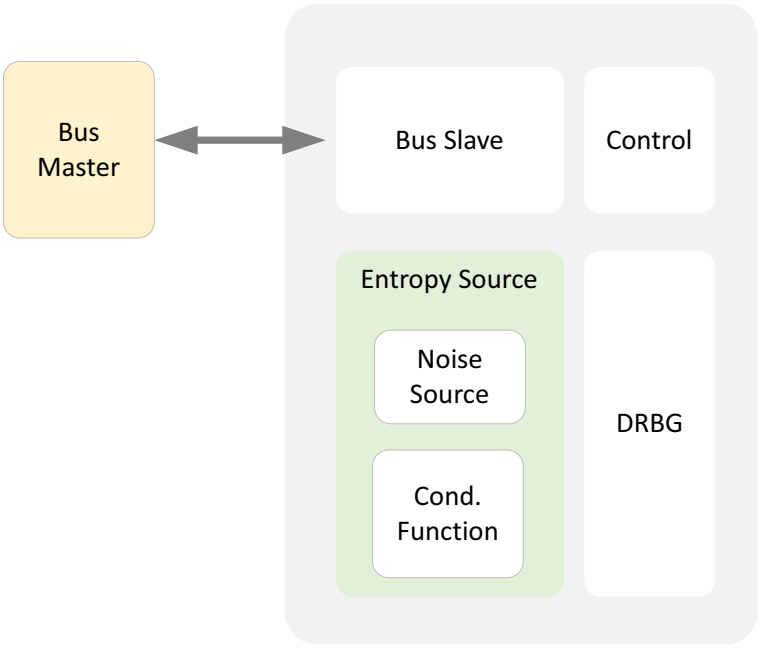
17 True Random Number Generator (TRNG)

17.1 Model Overview

This section describes the function and its use of the True Random Number Generator module.

The random number generator module contains entropy source and post processing (DRBG). The entropy source is designed using RO. The top block diagram of the random number generator is shown below.

Figure 17-1 Module Boundary



17.2 Register Description

Table 17-1 Register Map

Address	R/W	Description	Default Value
0x4000	RW	[0] RBGEN Random bit generator enable. [4:1] ROSEN Each bit states enable for one RO SOURCE. From RO SOURCE1 to RO SOURCE4. [7:5] RSVD	0x1f
0x4001	R	[15:8] RSVD	0x00

Address	R/W	Description	Default Value
0x4002	RW	[16] RSVD [17] DIEN Data interrupt enable. 0: Data interrupt is disabled. 1: Data interrupt is enabled. [18] ERIEN Empty read interrupt enable. 0: Empty read interrupt is disabled. 1: Empty read interrupt is enabled. [23:19] RSVD	0x02
0x4003	RW	[24] IRQEN Global interrupt enable. Bit 16~19 active only when this bit is 1. 0: Disable interrupt function 1: Enable interrupt function [31:25] RSVD	0x01
0x4004	RW	[0] MSEL Mode select. 0: TRBG without post-processing. 1: TRBG with post-processing. [7:1] RSVD	0x00
0x4005	R	[15:8] RSVD	0x00
0x4006	R	[23:16] RSVD	0x00
0x4007	R	[31:24] RSVD	0x00
0x4008	W1C	[0] RSVD [1] DRDY Data ready. [2] ERERR Empty read error. [7:3] RSVD	0x00
0x4009	R	[15:8] RSVD	0x00
0x400a	R	[23:16] RSVD	0x00
0x400b	R	[31:24] RSVD	0x00

Address	R/W	Description	Default Value
0x400c	R	[7:0] RSVD	0x00
0x400d	R	[15:8] RSVD	0x00
0x400e	R	[23:16] RSVD	0x00
0x400f	R	[31:24] RSVD	0x00
0x4010	R	[3:0] MIR Sub version number. [7:4] MAR Main version number.	0x00
0x4011	R	[15:8] RSVD	0x00
0x4012	R	[23:16] PROJECT PROJECT number.	0x00
0x4013	R	[31:24] PROJECT PROJECT number.	0x00
0x4020	RW	[2:0] DFTV DRNG FIFO count threshold value. DRDY interrupt will be generated when actual TRBG FIFO count exceeds this threshold, e.g., if set to 5, an interrupt will be generated when the actual FIFO count transits from 4 to 5. [7:3] RSVD	0x07
0x4021	R	[15:8] RSVD	0x00
0x4022	RW	[18:16] TFTV TRNG FIFO count threshold value DRDY interrupt will be generated when actual TRBG FIFO count exceeds this threshold, e.g., if set to 5, an interrupt will be generated when the actual FIFO count transits from 4 to 5. [23:19] RSVD	0x07
0x4023	R	[31:24] RSVD	0x00
0x4024	R	[7:0] DFCNT DRBG FIFO count. Current number of random number in TRBG FIFO.	0x00

Address	R/W	Description	Default Value
0x4025	R	[8] DFE DRBG FIFO empty. [15:9] RSVD [18:16] TFCNT TRBG FIFO count. Current number of random number in TRBG FIFO.	0x01
0x4026	R	[23:19] TFCNT TRBG FIFO count. Current number of random number in TRBG FIFO.	0x00
0x4027	R	[24] TFE TRBG FIFO empty. [31:25] RSVD	0x01
0x4080	RW	[7:0] ROEN2 RO enable of RO SOURCE2. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 2.	0xff
0x4081	RW	[15:8] ROEN2 RO enable of RO SOURCE2. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 2.	0xff
0x4082	RW	[23:16] ROEN1 RO enable of RO SOURCE1. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 1.	0xff
0x4083	RW	[31:24] ROEN1 RO enable of RO SOURCE1. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 1.	0xff
0x4084	RW	[7:0] ROEN4 RO enable of RO SOURCE4. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 4.	0xff

Address	R/W	Description	Default Value
0x4085	RW	[15:8] ROEN4 RO enable of RO SOURCE4. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 4.	0xff
0x4086	RW	[23:16] ROEN3 RO enable of RO SOURCE3. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 3.	0xff
0x4087	RW	[31:24] ROEN3 RO enable of RO SOURCE3. Each bit controls one RO. In total, there are 16 ROs in RW RO SOURCE 3.	0xff

17.3 Interrupt Description

The RBG module has the following interrupt sources:

- CPU reads RBG_DR without data
- Data valid

The above interrupts can be set by RBG_CR. By default, the data valid interrupt is enabled.

When the RBGEN of RBG_CR is low, the interrupt signal will not be cleared. Therefore, before enabling RBGEN, it is necessary to ensure that there is no previous interrupt signal, otherwise it will affect the next interrupt.

17.3.1 CPU Reads RBG_DR Without Data

In order to prevent the CPU from reading the invalid data, the RBG can remind the CPU to read in such a situation when there is no valid random number. In order to avoid the CPU reading the empty data, it is recommended to read the RBG_FIFO_SR first every time to get the random number before the CPU gets data in the current FIFO to avoid invalid data.

The CPU can clear the interrupt by writing 1 to ERERR in RBG_SR. If the write is successful, the interrupt will be cleared. When the above situation occurs again, the interrupt will be valid again.

17.3.2 Data Valid

RBG provides two ways to output data. When the interrupt is enabled, the random number can be read by the way of interrupting. In this design, the data in the corresponding FIFO will only be pulled up after the threshold is reached, thus the CPU can obtain multiple data at once. The threshold can be set by RBG_FIFO_CR. The CPU can clear the interrupt by writing 1 to DRDY of RBG_SR. If the write is successful, the interrupt will be pulled down. The interrupt is pulled high again when the data in the FIFO reaches the threshold again.

It is important to note that the interrupt will only be pulled up when the amount of data in the FIFO reaches the threshold. Therefore, the data in the FIFO exceeds the threshold firstly and then RBG module pulls up the interrupt. When the CPU doesn't obtain data or have obtained data but the amount of data remaining in the FIFO is still larger than the threshold, then clear the interrupt. Although the data in the FIFO is still larger than the threshold, it will not be interrupted.

In addition, the CPU can use the RBG_FIFO_SR register to view the remaining data in the FIFO. It can also use this method to obtain a random number. Check the RBG_FIFO_SR register when the random number is needed and the number of random numbers indicated by the register can be fetched at one time. If the rate at which the CPU handles random numbers is slower than the rate at which RBG random numbers are generated, it is generally not recommended to use interrupt to obtain random numbers.

17.4 Usage Procedure

17.4.1 Normal Operation

Turn off the RBG module first after the CPU works normally, that is to set RBGEN of the RBG_CR to 0. Then it can be configured and write 1 to RBGEN after the configuration is complete to make it work normally.

The CPU can configure RBG module by configuring RBG_CR, RBG_FIFO_CR and other optional configuration registers. For detailed configuration instructions, please refer to the description in Section 17.2.

When writing 1 to RBGEN in RBG_CR, the modification of the value of the above register will not affect the RBG. Therefore, when configuring, set the RBGEN in the RBG_CR register after configuring other registers to enable the OSR_RBG module.

TRBG and DRBG can be switched by modifying RBG_RTCR during the operation to meet different usage environments.

17.4.2 Entropy Source

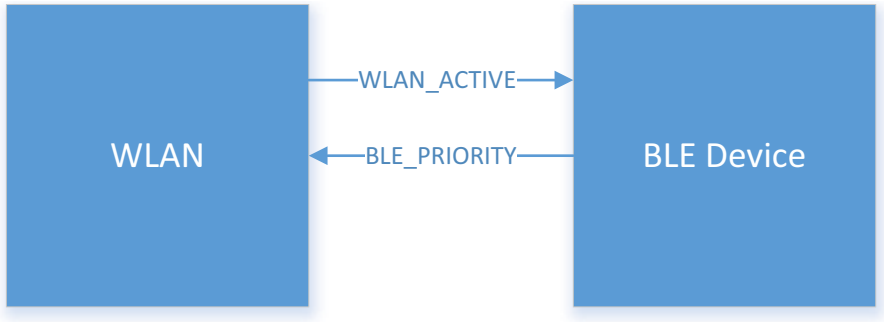
In this design, the random number generator module uses RO RNG as the entropy source. RO RNG contains modules such as random source and post-processing. RO RNG has four independent RO entropy sources. Each entropy source can choose to use its own RO CLK as the sampling clock or select the system clock as the sampling clock. The selection is determined by the input of I_rbg_sclk_sel, which is high for the system clock and low for the internal RO CLK. All RO enable signals are open at the same time and some of the ROs can be turned on or off by controlling the register.

18 PTA Interface

The TLSR8270 supports a Packet Traffic Arbitration (PTA) interface to facilitate co-existence with 802.11 WLAN. The TLSR8270F512ET48 supports a 2/3/4-wire PTA interface. Regarding the PTA's usage, the 2-wire PTA must use PB[3] + any other GPIO, the function of PB[3] is ble_priority; the 3-wire PTA must use PB[3]/ PB[4]/PB[5] and the 4-wire PTA must use PB[3]/PB[4]/PB[5] + any other GPIO, the function of PB[3]/PB[4]/ PB[5] is: PB[3]: ble_activity, PB[4]: ble_status, PB[5]: wlan_deny.

18.1 Two-Wire Signaling

Figure 18-1 Two-Wire Signaling



WLAN_ACTIVE:

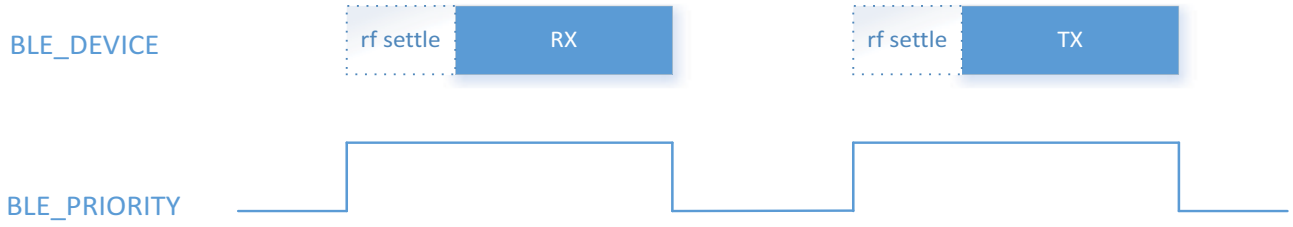
The WLAN_AVTIVE signal is asserted by WLAN controller when 802.11b/g packets are actively being transmitted or received. The BLE device avoids transmitting low-priority packets that are likely to cause interference with the 802.11b/g activity.

BLE_PRIORITY:

The BLE_PRIORITY signal should be asserted by the BLE device during high-priority transmit or receive activity. When this signal is asserted, WLAN device defers (or aborts) some or all of its transmissions.

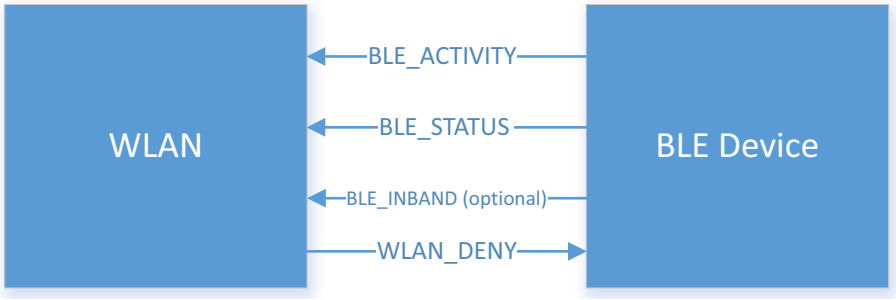
The WLAN_ACTIVE signal is judged by the software.

Figure 18-2 Example of Two-Wire PTA Timing Diagram



18.2 Three-Wire or Four-Wire Signaling

Figure 18-3 Three-Wire or Four-Wire Signaling



BLE_ACTIVITY:

The BLE device should assert BLE_ACTIVITY for the duration of a "transaction". This usually corresponds to a transmit-receive or receive-transmit pair. This signal is asserted the time t1 before rf settle operation of first BLE RX/TX packet.

BLE_STATUS:

At the same time as asserting BLE_ACTIVITY, the BLE device should assert BLE_STATUS if the transaction is considered to be high priority. After the time t2 the signal should be changed to indicate whether or not the BLE device is transmitting (asserted) or receiving (de-asserted). This signal must be updated prior to transmission or reception to indicate any change of direction.

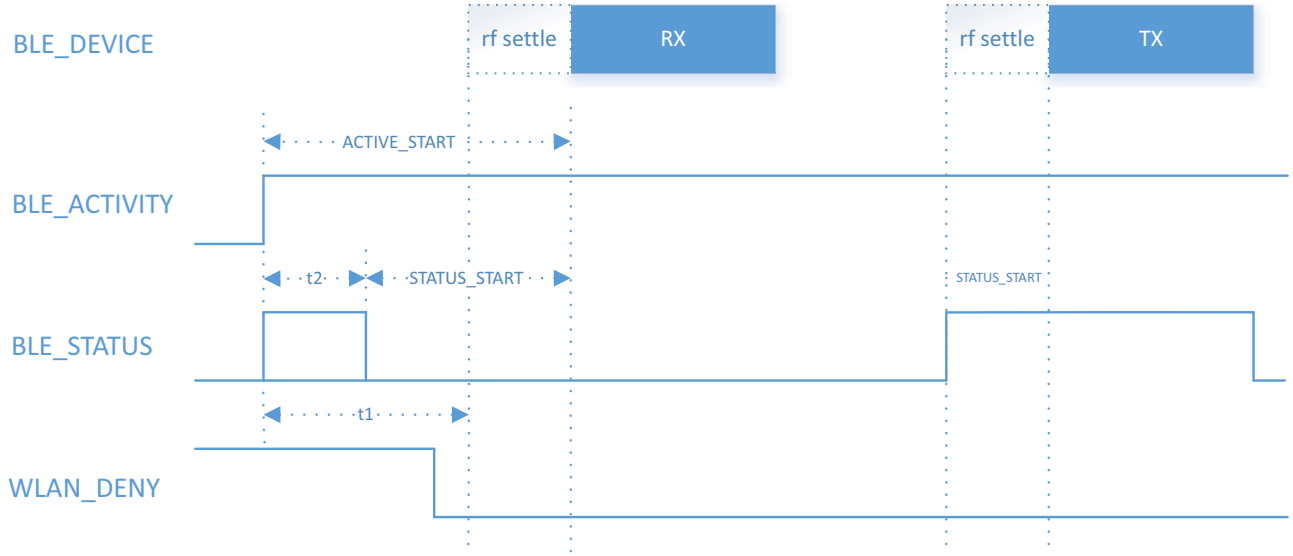
BLE_INBAND (optional):

This signal is optional and is only of benefit if there is sufficient isolation between the radios to support simultaneous operation on non-overlapping frequencies. The BLE device asserts BLE_INBAND (asserted by software) if any of the channels used in the transaction overlap the 802.11b/g frequencies.

WLAN_DENY:

The WLAN controller drives WLAN_DENY to indicate whether the requested BLE transaction is allowed or denied (which should be effective within the time t1 after asserting BLE_ACTIVITY) to determine the activity direction. If the signal is asserted, the BLE device does not start the transaction.

Figure 18-4 Example of Four-Wire PTA Timing Diagram



The two registers below are used to configure t1/t2:

Table 18-1 Register Configuration for t1/t2

Address	Name	R/W	Description	Default Value
0xf12	r_t_coex_t1	RW	[7:0]: Corresponds to t1 in Figure 18-4 above. Specifies the time after assertion of BLE_ACTIVITY signal at which the WLAN_DENY should be stable and is sampled by BLE device to determine whether to launch transaction The value of the register should be t1 - 1 (Unit: μs)	0x31
0xf13	r_t_coex_t2	RW	[7:0]: Corresponds to t2 in Figure 18-4 above. Specifies the time after assertion of the BLE_ACTIVITY signal at which the BLE_STATUS signal is changed from transaction priority to packet direction The value of the register should be t2 - 1 (Unit: μs)	0x13

19 Key Electrical Specifications

19.1 Absolute Maximum Ratings

Table 19-1 Absolute Maximum Ratings

Item	Sym.	Min	Max	Unit	Conditions
Supply voltage	VDD	-0.3	3.6	V	All AVDD, DVDD and VDD_IO pin must have the same voltage
Voltage on input pin	V_{In}	-0.3	VDD + 0.3	V	-
Output voltage	V_{Out}	0	VDD	V	-
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.*

19.2 Recommended Operating Conditions

Table 19-2 Recommended Operating Conditions

Item	Sym.	Min	Typ	Max	Unit	Conditions
Power-supply voltage	VDD	1.8	3.3	3.6	V	All AVDD, DVDD and VDD_IO pin must have the same voltage.
Supply rise time (from 1.6 V to 1.8 V)	t_R	-	-	10	ms	-
Operating temperature range	T_{Opr}	-40	-	85	°C	-

19.3 DC Characteristics

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

Table 19-3 DC Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
RX current	I_{Rx}	-	4.6	-	mA	Whole chip with DCDC
		-	9.1	-	mA	Whole chip with LDO
TX current	I_{Tx}	-	4.9	-	mA	Whole chip @ 0 dBm with DCDC
		-	9.5	-	mA	Whole chip @ 0 dBm with LDO
Deep sleep with 16 KB SRAM retention	I_{Deep1}	-	0.8	-	μA	Without 32K RC @ 0.6 V[a]
Deep sleep with 32 KB SRAM retention		-	1.0	-	μA	
Deep sleep without SRAM retention	I_{Deep2}	-	0.4	-	μA	
Deep sleep with 16 KB SRAM retention	I_{Deep3}	-	1.3	-	μA	With 32K RC @ 0.6 V[b]
Deep sleep with 32 KB SRAM retention		-	1.5	-	μA	
Deep sleep without SRAM retention	I_{Deep4}	-	0.8	-	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.

19.4 AC Characteristics

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

Table 19-4 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 19-5 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 BLE 125 kbps, ±250 kHz deviation BLE 500 kbps, ±250 kHz deviation 2.4G proprietary 500 kbps, ±125 kHz deviation 2.4G proprietary 250 kbps, ±62.5 kHz deviation					
BLE 1 Mbps RF_RX Performance (±250 kHz Deviation)							
Sensitivity	1 Mbps	-	-	-96	-	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	-

Item		Sym.	Min	Typ	Max	Unit	Conditions
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	
Image rejection		-	-	-26	-	dB	Wanted signal at -67 dBm
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	-
BLE 125 kbps RF_RX Performance (±250 kHz Deviation)							
Sensitivity	125 kbps	-	-	-100	-	dBm	-
Frequency offset tolerance		-	-200	-	+100	kHz	-
Co-channel rejection		-	-	4	-	dB	Wanted signal at -67 dBm
In-band blocking rejection (equal modulation interference)	+1/-1 MHz offset	-	-	-6/-2	-	dB	Wanted signal at -67 dBm
	+2/-2 MHz offset	-	-	-42/-37	-	dB	
	≥ 3 MHz offset	-	-	-42	-	dB	
Image rejection		-	-	-37	-	dB	Wanted signal at -67 dBm
BLE 125 kbps RF_TX Performance							

Item		Sym.	Min	Typ	Max	Unit	Conditions
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 500 kbps RF_RX Performance (±250 kHz Deviation)							
Sensitivity	500 kbps	-	-	-98	-	dBm	-
Frequency offset tolerance		-	-200	-	+100	kHz	-
Co-channel rejection		-	-	6	-	dB	Wanted signal at -67 dBm
In-band blocking rejection (equal modulation interference)	+1/-1 MHz offset	-	-	-6/-1	-	dB	Wanted signal at -67 dBm
	+2/-2 MHz offset	-	-	-42/-35	-	dB	
	≥ 3 MHz offset	-	-	-42	-	dB	
Image rejection		-	-	-35	-	dB	Wanted signal at -67 dBm
BLE 500 kbps 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	-

Table 19-6 USB Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
USB output signal cross-over voltage	V_{Crs}	1.3	-	2.0	V	-

Table 19-7 RSSI Characteristics

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

Table 19-8 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	-
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 19-9 RC Oscillator Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
24 MHz RC Oscillator						

Item	Sym.	Min	Typ	Max	Unit	Conditions
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 19-10 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	-

19.5 SPI Characteristics

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

Table 19-11 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

Item	Sym.	Min	Typ	Max	Unit	Conditions
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.

19.6 I2C Characteristics

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

Table 19-12 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	-

19.7 Flash Characteristics

T = -40 ~ +85°C unless otherwise stated.

Table 19-13 Flash Memory Characteristics

Item	Sym.	Min	Typ	Max	Unit	Conditions
Retention period	-	20	-	-	year	-
Number of erase cycles	-	100k	-	-	cycle	-
VDD for programming	-	1.65	-	2.0	V	Note this refers to the SoC supply
Sector size	-	-	4	-	KB	-
Page programming time	TPP	-	1.6	6	ms	-
Sector erase time	TSE	-	150	500	ms	-

Item	Sym.	Min	Typ	Max	Unit	Conditions
Block erase time (32 KB/64 KB)	TBE	-	0.5/0.8	2.0/3.0	s	-
Program current	I_P	-	-	10	mA	-
Erase current	I_E	-	-	10	mA	-

19.8 ESD Characteristics

Table 19-14 HBM/CDM Results

Model	Pin Combinations	Value	V Class
HBM	IO vs VSS(+)	+2 kV	JESD22-A114F Class-2: 2000 V - <4000 V
	IO vs VSS(-)	-2 kV	
	IO vs VDD(+)	+2 kV	
	IO vs VDD(-)	-2 kV	
	IO vs IO(+)	+2 kV	
	IO vs IO(-)	-2 kV	
	VDD vs VSS(+)	+2 kV	
	VDD vs VSS(-)	-2 kV	
	VDD vs VDD(+)	+2 kV	
	VDD vs VDD(-)	-2 kV	
CDM	ALL Pin(+)	+500 V	JEDEC22-C101F Class C2: 500 V - <1000 V
	ALL Pin(-)	-500 V	

Table 19-15 Latch-Up I-Test Result

Mode	Spec	Value	Pass/Fail
Positive	+100 mA	+100 mA	Pass
Negative	-100 mA	-100 mA	Pass

Table 19-16 Latch-Up V_{supply} Over Voltage Test Result

Part Number	Voltage	Mode	Spec	Value	Pass/Fail
TLSR8270F512ET48	1.2 V	Positive	$1.5V_{max}$	1.98 V	Pass
	1.4 V			2.31 V	
	1.8 V			2.97 V	
	3.3 V			5.445 V	
	5 V		MSV	6.25 V	

19.9 Storage Condition

The TLSR8270 series is applicable to Moisture Sensitivity Level 3 (based on JEDEC Standard).

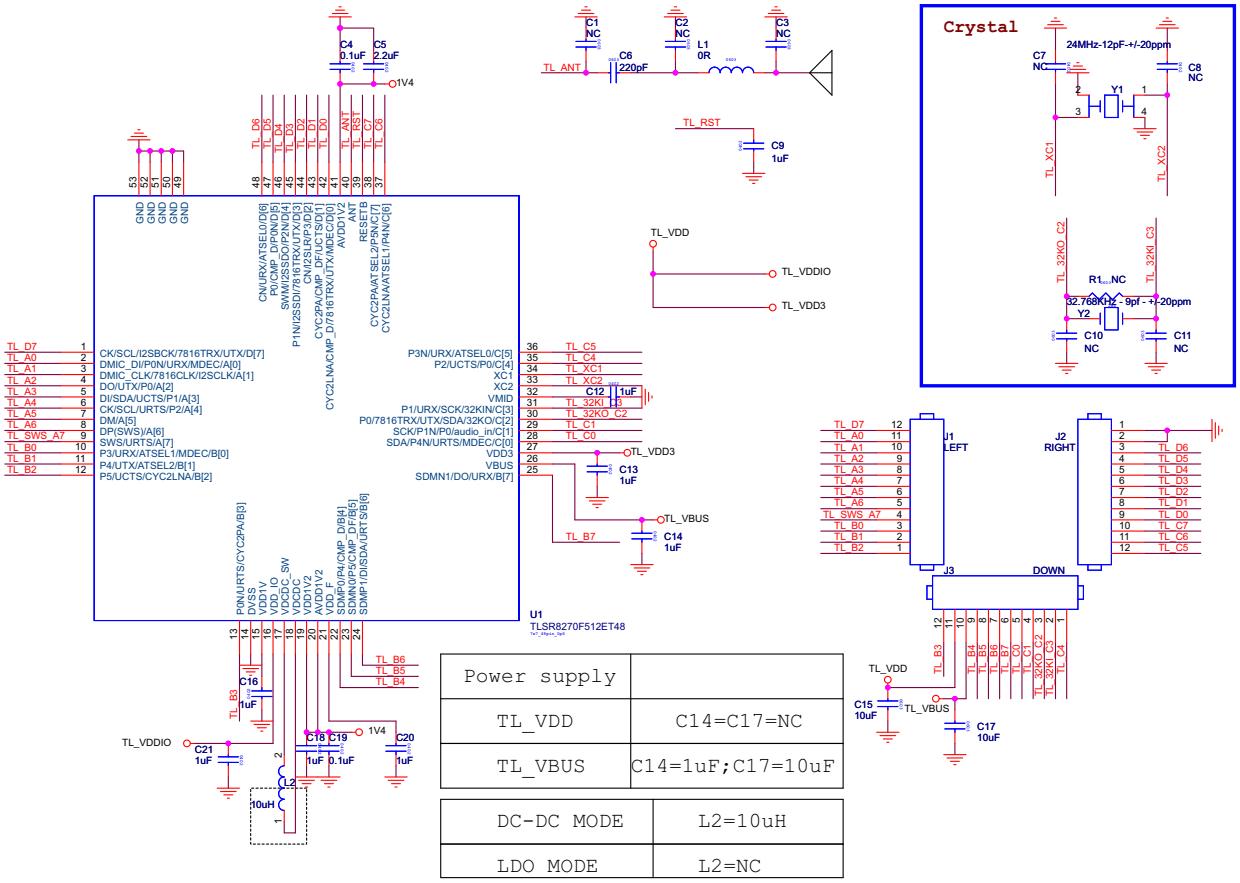
1. Calculated shelf life in sealed moisture barrier bag (MBB): 12 months at <40°C and <90% relative humidity (RH)
2. Peak package body temperature: 260°C
3. After bag is opened, devices that will be subjected to reflow solder or other high temperature process must be
   - Mounted within: 168 hours of 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.

20 Reference Design

20.1 Schematic of TLSR8270F512ET48

Figure 20-1 Schematic of TLSR8270F512ET48



Power supply	
TL_VDD	C14=C17=NC
TL_VBUS	C14=1uF;C17=10uF

DC-DC MODE	L2=10uH
LDO MODE	L2=NC

20.2 BOM (Bill of Material) of TLSR8270F512ET48

Table 20-1 BOM Table of TLSR8270F512ET48

Quantity	Reference	Value	Description	PCB Footprint
7	C1, C2, C3, C7, C8, C10, C11	NC	Not mounted	0402
2	C4, C19	0.1 μF	Capacitance, X5R, ±10%	0402
1	C5	2.2 μF	Capacitance, X5R, ±10%	0402
1	C6	220 pF	Capacitance, X7R, ±10%	0402