WM_W800_Register Manual

V3.0

Beijing Lianshengde Microelectronics Co., Ltd. (winner micro)

Address: Floor 18, Yindu Building, No. 67, Fucheng Road, Haidian District, Beijing

Tel: +86-10-62161900

Company website: www.winnermicro.com

Document modification record

Version revision time		revision history	author	review
V0.1	2019-09-25	create	chengsheng	
V0.2	2020-05-13	Revise the remaining description errors;	Chengsheng	
V1.0	2020-7-20	Update the digital function interface	Ray	
v2.0	2020-8-20	Improve the high-speed interface function	Ray	
v2.1	2020-09-08	Add SD_ADC module description	chengsheng	
V3.0	2022-08-20	Update company information and modify bugs	Ray	

Table of contents

Figure catalog

table directory

1 **Introduction**

1.1 Purpose of writing

The W800 chip is an embedded Wi-Fi SoC chip launched by Lianshengde Microelectronics. The chip is highly integrated, requires few peripheral devices, and is cost-effective.

high. Applicable to various smart products in the IoT (smart home) field. Highly integrated Wi-Fi and Bluetooth 4.2 Combo function is its main function;

In addition, the chip integrates XT804 core, built-in QFlash, SDIO, SPI, UART, GPIO, I²C, PWM, I²S, 7816, LCD,

Touch Sensor and other interfaces support multiple hardware encryption and decryption algorithms. In addition, the chip MCU contains a security kernel, which supports code security permission setting

The whole system supports various security measures such as firmware encrypted storage, firmware signature, security debugging, and security upgrade to improve product security features.

This document mainly describes the internal structure of the W800 chip, the information of each functional module and the detailed register usage information; it is a guide for developers to develop drivers,

The main reference material for the application. There are open source implementations of various functions in the SDK provided by Lianshengde Microelectronics. Developers can refer to the corresponding driver

Programs, application examples to increase understanding of chip functions and register descriptions. This document does not describe the registers of the Wi-Fi/BT part.

1.2 References

For W800 chip packaging parameters, electrical characteristics, radio frequency parameters and other information, please refer to "W800 Chip Product Specifications";

The W800 chip integrates a ROM program, which provides functions such as downloading firmware, reading and writing MAC addresses, and reading and writing Wi-Fi parameters.

For more information, please refer to "WM_W800_ROM Function Brief";

The W800 chip has a built-in 2Mbytes QFlash memory as a storage space for codes and parameters. This document provides the basic operation of QFlash

for information. If you have requirements beyond the scope of this document, you need to refer to the QFlash manual;

W800 chip adopts Hangzhou Pingtouge XT804 core, 804-related function introduction, development materials, etc. can refer to the information released by Pingtouge company;

For more information, please refer to the Winnermicro website (http://www.winnermicro.com/).

2 features

ÿ Chip packaging

ÿ QFN32 package, 4mm x 4mm.

ÿ Chip integration

ÿ Integrated XT804 processor, up to 240MHz

ÿ Integrated 288KB SRAM

ÿ Integrated 2MB FLASH

ÿ Integrated 8-channel DMA controller, supports 16 hardware applications, and supports software linked list management

ÿ Integrated PA/LNA/TR-Switch

ÿ Integrated 32.768KHz clock oscillator

ÿ Integrated voltage detection circuit

ÿ Integrated LDO

ÿ Integrated power-on reset circuit

ÿ Chip interface

ÿ Integrate 1 SDIO2.0 Device controller, support SDIO 1-bit/4-bit/SPI three operation modes, working clock range 0~50MHz

ÿ Integrate 1 SDIO 2.0 HOST controller, support SDIO and SD card operation, working clock range 0~50MHz

ÿ Integrate a QSPI PSRAM interface, support PSRAM with a maximum capacity of 64MB, and a maximum operating clock frequency of 80MHz;

ÿ Integrate 5 UART interfaces, support RTS/CTS, baud rate range 1200bps~2Mbps

ÿ Integrate a high-speed SPI slave device interface, the working clock range is 0~50MHz

ÿ Integrate 1 SPI master/slave interface, the working clock of the master device is up to 20MHz, and the slave device supports a data transmission rate of up to 6Mbps

ÿ Integrate an I2C controller, support 100/400Kbps rate

ÿ Integrated PWM controller, supports 5-way PWM

Single output or 2-way PWM input. Maximum output frequency 20MHz, maximum input frequency 20MHz

ÿ Integrated duplex I2S controller, support 32KHz to 192KHz I2S interface codec

ÿ Integrate a 7816 interface, compatible with UART interface, support ISO-7816-3 T=0.T=1 mode; support

EVM2000 protocol

ÿ Support multiple hardware encryption and decryption modes, including

RSA/AES/RC4/DES/3DES/RC4/SHA1/MD5/CRC8/CRC16/CRC32/TRNG

ÿ Integrate one differential or two single-ended 12bit ADC interfaces;

ÿ Integrated 11-way Touch Sensor;

ÿ Support up to 17 GPIO ports, each IO port has

Rich reuse relationship. With input and output configuration options.

ÿ WIFI protocol and function

ÿ Support GB15629.11-2006, IEEE802.11 b/g/n;

ÿ Support WMM/WMM-PS/WPA/WPA2/WPS

ÿ Support WiFi Direct;

ÿ Support EDCA channel access mode;

ÿ Support 20/40M bandwidth working mode;

ÿ Support STBC, GreenField, Short-GI, support reverse transmission;

ÿ Support RIFS frame interval;

ÿ Support AMPDU, AMSDU;

ÿ Support 802.11n MCS 0~7, MCS32 physical layer transmission rate stalls, the maximum transmission rate is 150Mbps;

ÿ Support HT-immediate Compressed BlockAck, normal ACK, no ACK response mode;

ÿ Support CTS to self;

ÿ Support AP function; AP and STA can be used at the same time;

ÿ In the BSS network, multiple multicast networks are supported, and the encryption methods of each multicast network are different.

32 multicast networks and inbound STA encryption;

ÿ When the BSS network is used as an AP, the total number of supported stations and groups is 32;

ÿ Receive sensitivity:

ÿ 20MHz MCS7@-71dBm@10%PERÿ

ÿ 40MHz MCS7@-67dBm@10%PERÿ

ÿ 54Mbps@-73dBm@10%PERÿ

ÿ 11Mbps@-86dBm@8%PERÿ

ÿ 1Mbps@-96dBm@8%PERÿ

ÿ Support a variety of received frame filtering options;

ÿ Support monitoring function;

ÿ Bluetooth protocol and functions

ÿ Integrated Bluetooth baseband processor/coprocessor, support BT/BLE4.2 protocol

ÿ Support various rates of DR/EDR;

ÿ Support BLE 1Mbps rate;

ÿ Power supply and power consumption

ÿ 3.3V single power supply;

ÿ Support Wi-Fi power saving mode power management;

ÿ Support work, sleep, standby, shutdown working modes;

ÿ Standby power consumption is less than 15uA;

3 Overview

This chip is a SOC chip that supports multi-interface and multi-protocol wireless LAN 802.11n (1T1R). The SOC chip integrates emitter

RF Transceiver, CMOS PA Power Amplifier, Baseband Processor/Media Access Control, SDIO, SPI,

Low-power WLAN chip with UART, GPIO and other interfaces.

The W800 chip supports GB15629.11-2006, IEEE802.11 b/g/ n protocols, and supports STBC, Green Field,

Short-GI, Reverse Transfer, RIFS Frame Interval, AMPDU, AMSDU, T-immediate Compressed Block Ack,

Rich protocols and operations such as normal ACK, no ACK, CTS to self, etc.

The W800 chip integrates the RF transceiver front-end, A/D and D/A converters. It supports DSSS (Direct Sequence Spread Spectrum) as well as OFDM

(Orthogonal Frequency Division Multiplexing) modulation mode, with data descrambling capability, supports a variety of different data transmission rates. The analog front end of the transceiver

The equipped transceiver AGC function enables the system-on-a-chip to obtain the best performance. The W800 chip also includes a built-in enhanced signal monitor,

The influence of multipath effect can be largely eliminated.

In terms of security, the W800 chip not only supports the national standard WAPI encryption, but also supports the international standard WEP, TKIP, CCMP encryption,

These hardware components enable the data transmission system based on this chip to still obtain data close to that of non-encrypted communication during secure communication.

Data transfer performance.

In addition to supporting the energy-saving operation specified by IEEE802.11 and Wi-Fi protocol, the W800 chip also supports user-defined energy-saving solutions. chip

It supports four working modes: work, sleep, standby, and shutdown, so that the entire system can achieve low power consumption, and it is convenient for users to

Scenarios define different energy-saving solutions.

The W800 chip integrates a high-performance 32-bit embedded processor, a large amount of memory resources, and rich peripheral interfaces, which is convenient for users

It is easy to apply the chip to the secondary development of specific products.

The W800 chip supports the AP function, which can realize the establishment of 5 SSID networks at the same time, and realize the function of 5 independent APs. Supports building multiple

Multicast network function. It can realize the function of establishing a BSS network as an AP while joining other networks as a STA.

W800 chip supports WPS mode, so that users can realize encrypted complete network with one-button operation to ensure information security

sex.

The multi-function and high integration of W800 chip ensure that the WLAN system does not need too many off-chip circuits and external memory.

4 chip structure

4.1 Chip structure

The figure below describes the overall structure of W800 chip, the core part includes XT804 CPU, 288KB SRAM and 20KB ROM storage space.

The PMU part is used as the constant power supply module of the chip to provide power-on sequence management, start-up clock, real-time clock functions, etc. Provides a rich set of peripherals function and hardware encryption and decryption functions. The Wi-Fi part integrates MAC, BB and RF.

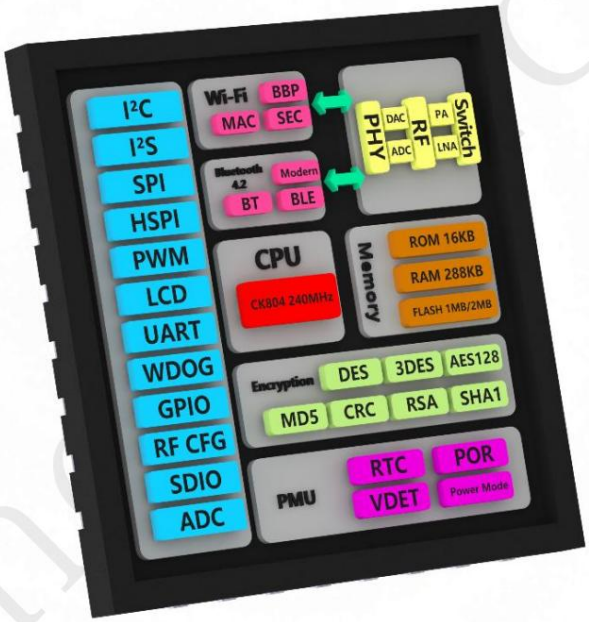


Figure 1 W800 chip structure diagram

4.2 Bus structure

The W800 chip consists of a two-level bus, as shown in the figure below

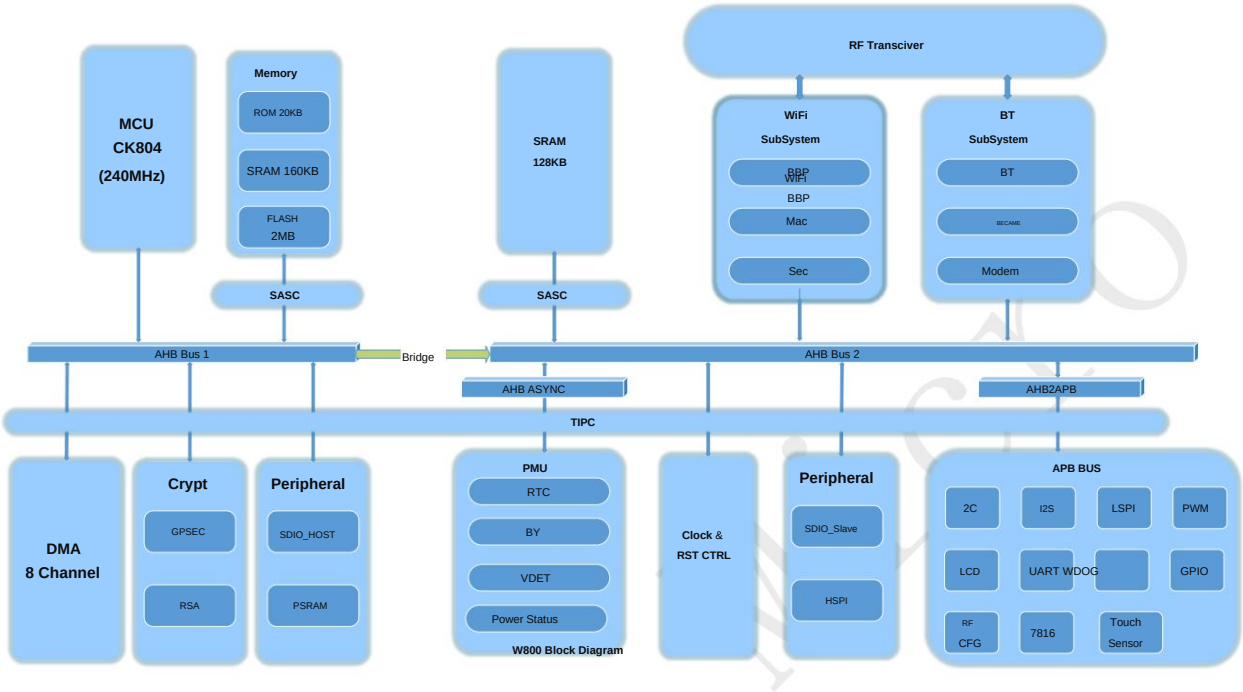


Figure 2 W800 bus structure diagram

(1) AHB-1 bus

This level of bus has four master devices - XT804, DMA, GPSEC and 5 slave devices.

XT804 is a 32-bit high-energy-efficiency embedded CPU core oriented to the control field. It adopts a 16/32-bit mixed coding instruction system and is designed

A streamlined and efficient 3-stage pipeline.

XT804 provides a variety of configurable features, including hardware floating point unit, on-chip cache, DSP acceleration unit, trusted protection technology,

On-chip tightly coupled IP, etc., users can configure according to application needs. In addition, XT804 provides multi-bus interface, supports system bus, pointer

Flexible configuration of command bus and data bus. XT804 has made a special acceleration for interrupt response, and the interrupt response delay only needs 13 cycles.

The bus clock works at the fastest frequency of 240MHz, and can be configured as 240/160/120/80/40MHz, or lower.

Table 1 AHB-1 bus master list

master device	Function
CPU	Complete chip register configuration, memory management and use, and complete 802.11MAC protocol. Highest Operating frequency 240MHz
DMA supports an independent 8-channel DMA module with a linked list structure, and supports 16 hardware DMA request sources on-chip.	
GPSEC	General encryption module, support DES/3DESSHA1/AES/MD5/RC4/CRC/PRDN. automatic completion The data blocks in the specified memory space are encrypted and written back.

Table 2 AHB-1 bus slave device list

Slave function	
ROM	ROM is used to store the initialization firmware after the CPU is powered on. Mainly complete the initial configuration of the chip register space and other work. After completing the above work, control the CPU The control is handed over to the firmware stored in FLASH.
AHB2AHB	Complete the conversion of CPU bus clock domain to BusMatrix2 bus clock domain master access. Require The clock domain must be of the same source, and the ratio of the CPU clock to the BusMatrix2 clock frequency is M: 1, M is an integer greater than or equal to 1.
FLASH stores firmware code and operating parameters.	
The 160KB of SRAM can be used to store instructions or data, and the firmware can use this memory as needed.	
RSA	Support up to 2048bit RSA encryption and decryption operations

GPSEC	general encryption/decryption module, supports SHA1/AES/MD5/RC4/CRC/TRNG. Autocomplete means Encrypt/decrypt data blocks in a given memory space and write them back.
SDIO_HOST	SDIO 2.0 standard SDIO HOST controller; can access SDIO interface peripherals through this interface. The SDIO interface clock is obtained by dividing the frequency of the bus clock, and the maximum support is 50MHz.
PSRAM_CTRL	PSRAM controller for QSPI interface. External PSRAM can be accessed through this controller. QSPI interface The port clock is obtained by dividing the frequency of the bus clock, and supports up to 80MHz clock.

(2) AHB-2 bus

This bus has 4 master devices and 3 slave devices, using a crossbar connection structure, which can realize different master devices to different slave devices Simultaneous access, thereby increasing the bandwidth. The bus clock works at the fastest frequency of 40MHz, and can be configured to be lower as required.

Table 3 AHB-2 bus master list

master device	Function
MAC	802.11MAC control protocol processing module. Operations on the bus mainly consist of sending reads data, receive write data, and send operations such as writing back of completion descriptors.
SEC	The security module completes the encryption, decryption and transfer of sent and received data. When sending, the data will be sent and the The MAC descriptor is moved to the specified location, and encryption is completed; when receiving, the received data and The MAC receive descriptor is moved to the specified location and decryption is completed.
AHB2AHB	Conversion of bus master access from the AHB-1 bus to the AHB-2 bus.
SDIO/HSPI	Connect the host to the chip through SDIO2.0 device controller or high-speed SPI slave device controller Accesses are converted to AHB bus signals and access to content memory and register space.

Each master device adopts a fixed priority, and the priority decreases from top to bottom.

Table 4 AHB-2 bus slave device list

slave device	Function
SRAM 128KB	Used to store uplink and downlink data cache, SDIO/SPI/UART interface uses this RAM as data cache
Configuration register configuration space, where the high-speed module configuration registers are uniformly addressed.	
APB All low-speed modules access space, and various low-speed modules are connected by APB bus.	
BT_CORE Bluetooth controller.	

4.3 Clock structure

W800 uses a 24/40MHz crystal as the SoC clock source, and a built-in DPLL outputs 480MHz to supply

Used by CPU, system bus, data bus and WiFi system; there is also a built-in 32.768KHZ RC oscillator for PMU

And LCD module use. An overview of the clock structure is shown in the figure below.

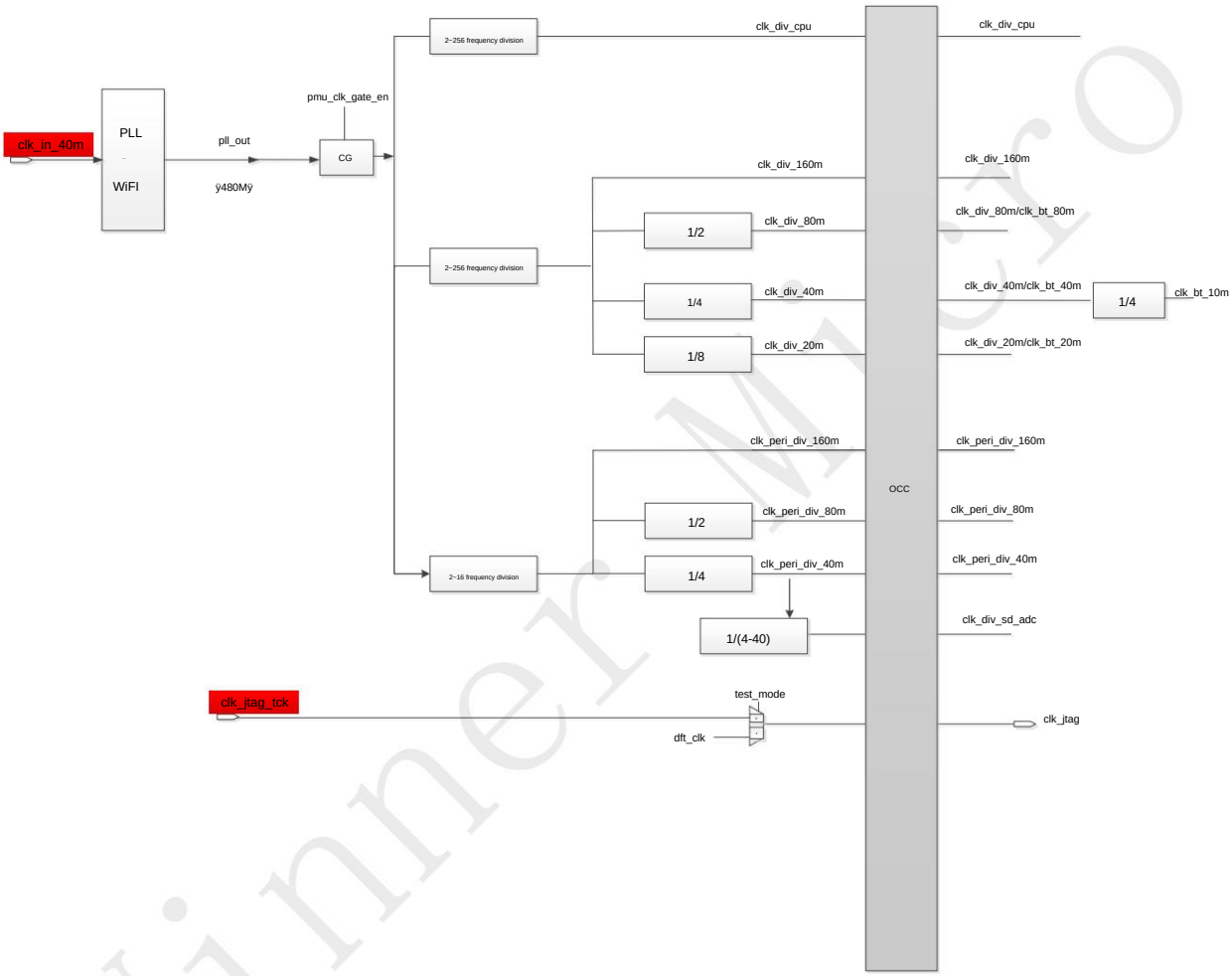
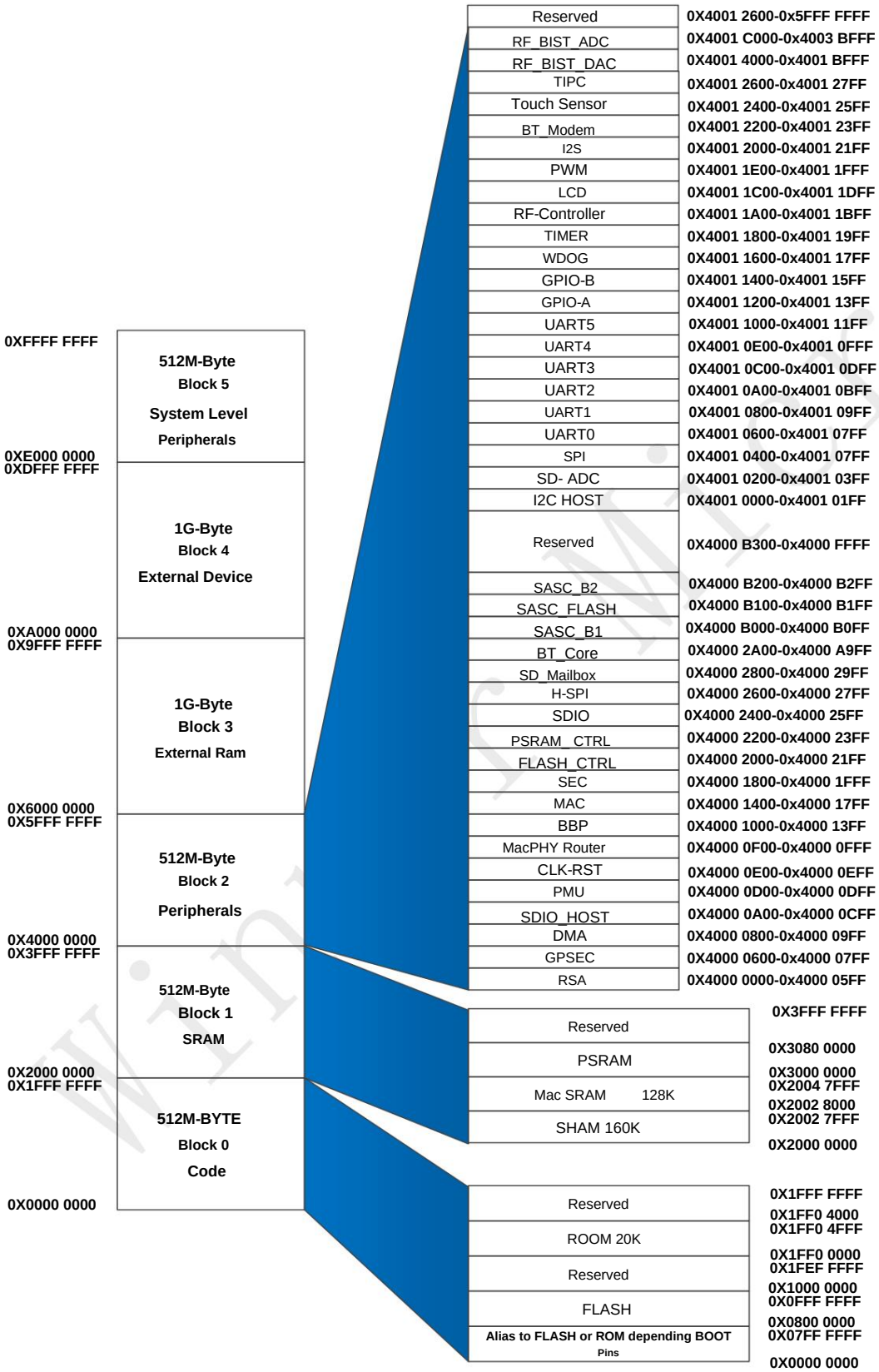


Figure 3 W800 clock structure

4.4 Address space



W800 Address Mapping

XT804 supports 4G storage space, which is divided into 6 blocks as shown in the above figure, which are code area, memory area, on-chip peripherals, and off-chip storage area, off-chip peripherals and system peripherals area. According to requirements, the w800 on-chip storage space is mapped to the first three areas as shown in Figure 3.

table 5 Detailed division of bus device address space

bus from equipment	BootMode=0	Address Space Subdivision	Remark
ROM 0x0000 0000 ~ 0x0004 FFF			Store the solidified firmware code
FLASH 0x0800 0000 ~ 0x0FFF	**FFFF**		stored as a dedicated instruction device.
SRAM 0x2000 0000 ~ 0x2002	7fff		Firmware memory and instruction storage Area
Mac RAM 0x2002 8000 ~ 0x2004	7fff		SDIO/H-SPI/UART data cache
PSRAM	0x3000 000~0x30800000		peripheral memory
CONFIG 0x4000 0000 ~ 0x4000	2FFF	0x4000 0000 ~ 0x4000 05FF	RSA configuration space
		0x4000 0600 ~ 0x4000 07FF	GPSEC configuration space
		0x4000 0800 ~ 0x4000 09FF	DMA configuration space
		0x4000 0A00 ~ 0x4000 0CFF	SDIO_HOST configuration empty between
		0x4000 0D00 ~ 0x4000 0DFF	PMU configuration space
		0x4000 0E00 ~ 0x4000 0EFF	Clock and Reset configuration

			space
		0x4000 0F00 ~ 0x4000 0FFF	MacPHY Router with setting space
		0x4000 1000 ~ 0x4000 13FF	BBP configuration space
		0x4000 1400 ~ 0x4000 17FF	MAC configuration space
		0x4000 1800 ~ 0x4000 1FFF	SEC configuration space
		0x4000 2000 ~ 0x4000 21FF	FLASH Controller configuration space
		0x4000 2200 ~ 0x4000 23FF	PSRAM_CTRL configuration space
		0x4000 2400 ~ 0x4000 25FF	SDIO Slave configuration empty between
		0x4000 2600 ~ 0x4000 27FF	H-SPI configuration space
		0x4000 2800 ~ 0x4000 29FF	SD Wrapper configuration space
		0x4000 2A00 ~ 0x4000 A9FF	BT Core configuration space
		0x4000 B000 ~ 0x4000 B0FF	SASC-B1 Level 1 bus memory security configuration module piece
		0x4000 B100 ~ 0x4000 B1FF	SASC-Flash Flash Security Configuration Module
		0x4000 B200 ~ 0x4000 B2FF	SASC-B2

			Secondary bus memory security configuration module piece
APB	0x4001 0000 ~ 0x 4001 C000	0x4001 0000 ~ 0x4001 01FF	I2C master
		0x4001 0200 ~ 0x4001 03FF	Sigma ADC
		0x4001 0400 ~ 0x4001 07FF	SPI master
		0x4001 0600 ~ 0x4001 07FF	UART0
		0x4001 0800 ~ 0x4001 09FF	UART1
		0x4001 0A00 ~ 0x4001 0BFF	UART2
		0x4001 0C00 ~ 0x4001 0DFF	UART3
		0x4001 0E00 ~ 0x4001 0FFF	UART4
		0x4001 1000 ~ 0x4001 11FF	UART5
		0x4001 1200 ~ 0x4001 13FF	GPIO-A
		0x4001 1400 ~ 0x4001 15FF	GPIO-B
		0x4001 1600 ~ 0x4001 17FF	WatchDog
		0x4001 1800 ~ 0x4001 19FF	Timer
		0x4001 1A00 ~ 0x4001 1BFF	RF_Controller
		0x4001 1C00 ~ 0x4001 1DFF	LCD
		0x4001 1E00 ~ 0x4001 1FFF	PWM

		0x4001 2000 ~ 0x4001 22FF	I2S
		0x4001 2200 ~ 0x4001 23FF	BT modem
		0x4001 2400 ~ 0x4001 25FF	Touch Sensor
		0x4001 2600 ~ 0x4001 25FF	TIPC Interface security settings
		0x4001 4000 ~ 0x4000 BFFF	RF_BIST DAC Transmit Memory
		0x4001 C000 ~ 0x4003 BFFF	RF_BIST ADC receive Memory
		0x4001 3C00 ~ 0x5FFF FFFF	RSV

4.4.1 SRAM

W800 has built-in 288KB SRAM. Among them, 160KB is mounted on the primary AHB bus, and 128KB is mounted on the secondary AHB bus. CPU

Devices on the primary bus can access all memory areas, but devices on the secondary bus can only access 128KB of memory on the secondary bus.

4.4.2 Flash

4.4.2.1 QFlash

W800 integrates 2MBytes QFlash inside. Through the integrated 32KB cache inside the chip, the XIP method is used to execute the program on QFlash

sequence. During the running of the program, the CPU first reads instructions from the Cache, and when the instruction cannot be obtained, it reads the instructions in 8Bytes per line

QFlash reads instructions and stores them in Cache. Therefore, when the continuous running code size is less than 32K, the CPU will not need to read from QFlash

Instructions are fetched, and the CPU can run at a higher frequency at this time. The above method is the operation method of reading instructions, and the RO segment of the entire Image will be

Operate in this way. This process requires no user intervention.

QFlash can also store data. When the user program needs to read and write data in QFlash, it needs to be done through the built-in QFlash controller.

Operation, QFlash provides corresponding address, instruction and other registers to help realize the operation that the user wants. For details, please refer to QFlash

The controller corresponds to the chapter.

Users need to pay attention to that when the program reads or writes data, it does not need to perform status judgment, wait and other operations, because the QFlash control

The device itself will make a judgment. When the QFlash controller returns, it indicates that the read or write has been completed.

4.4.2.2 SPI Flash

In addition to supporting 6PIN QFlash interface (built-in PIN, not packaged), W800 chip also supports low-speed SPI interface access. The SPI

The maximum operating frequency of the interface can reach 20MHz, and supports master-slave functions. For detailed description, refer to the corresponding chapter of the SPI interface.

4.4.3 PSRAM

W800 has a PSRAM controller with a built-in SPI/QSPI interface, supports external PSRAM device access with a maximum capacity of 64Mb, and provides a bus

Mode PSRAM read and write erase operations. The highest read and write speed is 80MHz. When the storage capacity needs to be expanded, the off-chip PSRAM can be used to expand

Fill code storage space or data storage space. PSRAM also supports program execution in XIP mode, and CPU Cache also supports caching

Data in PSRAM.

4.5 Start configuration

After the W800 chip is powered on, the CPU will start to execute the firmware in the ROM, and load the user Image at the specified address in the Flash.

The ROM firmware will read the BootMode (PA0) pin when it starts running, and judge to enter the boot state according to the signal of the pin:

Table 6 Start configuration

BootMode	start condition	boot mode
high		normal boot process
Low	Last <30ms, the quick test mode is invalid	normal boot process
	Continuous>=30ms	Enter function mode
Note: Test mode: chip test function, which cannot be operated by the user. Function mode: enter the basic functions realized by ROM, such as downloading firmware, programming MAC address, etc. For details, refer to		

"WM_W800_ROM Function Brief.pdf"

Typically, the BootMode pin should be used in production or debug phase. During the production phase, the user continuously pulls the BootMode pin

When it is low for more than 30ms, it enters the function mode and can quickly burn the Flash.

In the scenario of product rework or repair, if the chip does not enter the "highest security level" (for the description of the security level, please refer to

"WM_W800_ROM Function Brief"), you can enter the function mode through this pin, erase the old Image, write a new

Imageÿ

In the debugging stage, regardless of any faults in the firmware, you can enter the serial port by pulling the BootMode pin low for more than 30ms

Download function, burn new firmware.

5 Clock and reset module

5.1 Function overview

The clock and reset module completes the software control of the chip clock and reset system. Clock control includes clock frequency conversion, clock shutdown and adaptive

Should be gated; reset control includes soft reset control of the system and sub-modules.

5.2 Main features

- ÿ Support clock shutdown of each module
- ÿ Support partial module clock adaptive shutdown
- ÿ Support software reset of each module
- ÿ Support CPU frequency setting
- ÿ Support ADC/DAC loopback test
- ÿ Support I2S clock setting

5.3 Functional description

5.3.1 Clock gating

By configuring the clock gating enable register CLK_GATE_EN, you can control the clock shutdown of the specified function, so as to turn off a certain module function.

able purpose.

In order to provide the flexibility of the firmware to control the power consumption of the system, the clock and reset module provides the clock gating function of each module in the system. when off

When the clock of the corresponding module is disabled, the digital logic and clock tree of the module will stop working, which can reduce the dynamic power consumption of the system.

The specific switch of each module corresponds to the detailed description of the register SW_CLKG_EN.

5.3.2 Clock adaptive shutdown

According to the transition of some internal states, the chip adaptively turns off the clocks of some functional modules.

Please do not change the configuration, otherwise it may cause system abnormality when PMU function is configured.

5.3.3 Function reset

The chip provides the soft reset function of each subsystem, and the subsystem reset can be achieved by setting the corresponding BIT of SW_RST_CTRL to 0.

However, the reset status will not be cleared automatically, and the corresponding BIT bit of SW_RST_CTRL must be set to 1 to resume normal operation.

The soft reset function will not reset the CPU and WatchDog.

In this register, the reset operation of APB/BUS1/BUS2 (corresponding to APB bus, system bus and data bus) is not recommended, which will cause system

The system access device is abnormal.

5.3.4 Clock frequency division

The W800 system uses 40MHz/24MHz crystal as the system clock source, the system has a built-in DPLL, and fixedly outputs 480MHz clock as

The clock source of the whole system (as shown in the figure below).

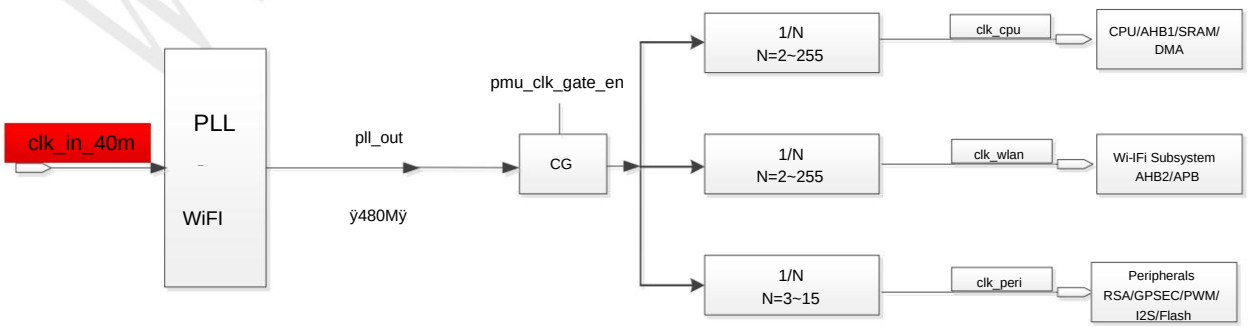


Figure 4 System clock frequency division relationship

The clock of the system bus is consistent with the CPU clock, and the clock of the data bus is fixed at 1/4 of the WLAN root clock.

The WLAN root clock is also the clock source of the entire WLAN system.

This module provides the function of setting CPU clock and WLAN root clock for firmware to adjust system performance and power consumption.

Setting the BIT[7:0] of the SYS_CLK_DIV register can adjust the CPU clock frequency division factor. The source clock of CPU clock frequency division is DPLL

output, fixed at 480MHz. The default value of the CPU clock division factor is 6, that is, the CPU default operating frequency is divided by 6 of 480MHz.

That is 80MHz. This parameter can be reconfigured when the required clock of the CPU needs to be adjusted.

The CLK_PERI clock provides the root clock of the operating clock of the encryption module in the SoC system, and the root clock of the operating clock of some interfaces, such as

Such as PWM interface, I2S interface, Flash interface clock. This clock is also divided by the 480MHz output from the DPLL. normal working condition

The lower frequency should be fixed at 3, and the root clock of CLK_PERI is 160MHz. It is obtained by dividing the frequency by 2 and dividing by 4 from the root clock of CLK_PERI

80MHz and 40MHz are provided for encryption module and interface module.

Setting BIT[15:8] of SYS_CLK_DIV register can adjust WLAN clock frequency division factor. The default frequency division factor is 3, that is, for DPLL

The 480MHz output frequency is divided by 3 to obtain a 160MHz clock, which is sent to the WLAN as the root node clock (the WLAN continues to divide the frequency to obtain a more

It is a detailed low frequency clock for use by WLAN systems.

Note: If you want the WLAN system to work normally, the WLAN root clock needs to be kept at 160MHz, otherwise the WLAN system will fail.

When the WLAN system is not required to work, the WLAN root clock can be lowered to reduce the dynamic power consumption of the system.

When changing the system clock configuration, you need to pay attention: the ratio of the system bus to the data bus needs to be maintained at M: 1, where M is an integer,

Minimum is 1. When changing the system clock configuration, it is also necessary to update the BIT [23:16] of the register SYS_CLK_DIV at the same time, and set the correct ratio

example coefficient. Otherwise, accessing the data bus will result in abnormal data.

[15:8] of SYS_CLK_SEL provides the frequency division factor for setting the working frequency of SAR_ADC, with 40M as the clock source for frequency division. Crossover system

The number is the assigned frequency division value.

BIT[4] of SYS_CLK_SEL is used to configure the clock frequency selection of the core operation of the RSA module, which can be 80MHz or 160MHz.

BIT[5] is to configure the clock frequency selection of the core operation of the GPSEC module, which can be 80MHz or 160MHz.

BIT[6] is to configure the clock frequency selection of the external bus of the FLASH module, which can be 40MHz or 80MHz.

When you need to reconfigure cpu_clk_divider, wlan_clk_divider, bus2_syncdn_factor, sdadc_fdiv, you need to set

Bit BIT[31] of SYS_CLK_DIV, the hardware automatically updates the above four parameters to the frequency divider, and then clears BIT[31].

I2S_CLK_CTRL provides the clock configuration function of the I2S module.

5.3.5 Debug function control

Users can enable and disable JTAG function by setting the value of DEBUG_CTRL (SYS_CLK_SEL-BIT[16]).

5.4 Register Description

5.4.1 Register list

Table 7 Clock reset module register list

offset address	name	abbreviation	access	describe	reset value
0X0000 Software clock gating enable register SW_CLKG_EN			RW software configuration module whether to turn off the clock		0X0000_7FFF
0X0004 Software clock mask register SW_CLK_MASK RW				Whether the software configures whether the module is adaptively shut down bell	0X0000_007E
0X0008 Reserved					
0X000C Software reset control register SW_RST_CTRL			RW software configuration reset module		0X01FF_FFFF
0X0010 Clock frequency division configuration register		SYS_CLK_DIV	RW configuration clock divider ratio		0X0000_2212
0X0014 Debug Control Register		DEBUG_CTRL	RW configure ADC/DAC loopback test		0X0000_0000
0X0018	I2S Clock Control Register	I2S_CLK_CTRL	RW configure I2S clock		0X0000_0000
0X001C reset status register		RESET_STATUS RW		View CPU soft reset and Watchdog Reset state	0x0000_0000

5.4.2 Software Clock Gating Enable Register

Table 8 Software Clock Gating Enable Register

bit access		Instructions	reset value
[31:22] RO		reserve	
[21]	RW	soft_touch_gate_en Configure the clock gating of the touch_sensor module, the default touch_sensor module gating is enabled 0: touch_sensor module clock is off	1'b1

		1: touch_sensor clock is on	
[20]	RW	soft_bt_gate_en Configure the gating control of the BT./BLE module clock, the default BT/BLE module gating is enabled 0: BT/BLE module clock is off 1: BT/BLE clock is on	1'b1
[19]	RW	soft_qsram_gate_en Configure the gating control of the qspi_ram module clock, the default qspi_ram module gating is enabled 0: qspi_ram module clock is off 1: qspi_ram clock is on	1'b1
[18]	RW	soft_sdio_m_gate_en Configure the gate control of the sdio_master module clock, the default sdio_master module gate control is enabled 0: sdio_master module clock is off 1: sdio_master clock is on	1'b1
[17]	RW	soft_gpsec_gate_en Configure the gating control of the gpsec module clock, and the gating control of the gpsec module is enabled by default 0: gpsec module clock off 1: gpsec clock on	1'b1
[16]	RW	soft_rsa_gate_en Configure the gating of the RSA clock, the default RSA gating is enabled 0: RSA module clock is off 1: RSA clock is on	1'b1
[15]	RW	soft_i2s_gate_en	1'b1

		Configure the gating of the i2s clock, the default i2s gating is enabled 0: i2s clock off 1: i2s clock is on	
[14]	RW	soft_lcd_gate_en Configure the gating of the lcd clock, the default lcd gating is enabled 0: lcd clock off 1: LCD clock is on	1'b1
[13]	RW	Soft_pwm_gate_en Configure the gating of the pwm clock, the default pwm gating is enabled 0: pwm clock off 1: pwm clock on	1'b1
[12]	RW	soft_sd_adc_gate_en Configure the gating of sd_adc_ clock, the default sd_adc_ gating is enabled 0: sd_adc_clock off 1: sd_adc_clock on	1'b1
[11]	RW	soft_gpio_gate_en Configure the gating of the GPIO clock, the default GPIO gating is enabled 0: GPIO clock off 1: GPIO clock is on	1'b1
[10]	RW	soft_timer_gate_en Configure the gating of the timer clock, the default timer gating is enabled 0: timer clock off	1'b1

		1: timer clock is on	
[9]	RW	soft_rf_cfg_gate_en: internal use, do not modify Configure the gating of the rf_cfg clock, the default rf_cfg gating is enabled 1'b0: rf_cfg clock off 1'b1: rf_cfg clock on	1'b1
[8]	RW	soft_dma_gate_en Indicates whether the clock supplied to the dma clock domain is turned off 1'b0: dma clock off 1'b1: dma clock on	1'b1
[7]	RW	soft_ls_spi_gate_en Configure the gate control of the low-speed spi clock, the default low-speed spi gate is enabled 1'b0: low speed spi clock off 1'b1: low-speed spi clock on	1'b1
[6]	RW	soft_uart5_gate_en Configure the gate control of uart5, the default uart5 is open 0: uart5 off 1: uart5 open	1'b1
[5]	RW	soft_uart4_gate_en Configure the gate control of uart4, uart4 is enabled by default 0: uart4 off 1: uart4 open	1'b1
[4]	RW	soft_uart3_gate_en	1'b1

		Configure the gate control of uart3, uart3 is enabled by default 0: uart3 off 1: uart3 open	
[3]	RW	soft_uart2_gate_en Configure the gate control of uart2, uart2 is enabled by default 0: uart2 off 1: uart2 open	1'b1
[2]	RW	soft_uart1_gate_en Configure the gating of uart1 clock, the default uart1 gating is on 1'b0: uart1 clock off 1'b1: uart1 clock on	1'b1
[1]	RW	soft_uart0_gate_en Configure the gate control of uart0 clock, the default uart0 gate control is enabled 1'b0: uart0 clock off 1'b1: uart0 clock on	1'b1
[0]	RW	soft_i2c_gate_en Configure the gating of the i2c clock, the default i2c gating is enabled 1'b0: i2c clock off 1'b1: i2c clock on	1'b1

5.4.3 Software Clock Mask Register

Table 9 Software Clock Mask Register

bit access		Instructions	reset value
[6]	RW	soft_cpu_clk_gt_mask Indicates whether the clock supplied to the CPU clock domain (including CPU, bus1, ROM, SRAM) can be adaptive Shutdown (when the **CPU needs to enter the WFI state, do not set adaptive shutdown)** 1'b0: Allow adaptive shutdown and turn-on 1'b1: Adaptive power off and on not allowed	1'b1
[5 : 2] RW		**reserved for internal use, do not modify**	
[1]	RW	soft_sdioahb_clk_gt_mask Indicates whether the clock supplied to the sdio ahb clock domain can be adaptively shut down 1'b0: Allow adaptive shutdown and turn-on 1'b1: Adaptive power off and on not allowed	1'b1
[0]	RW	soft_pmu_clk_gt_mask After the clock output by pll, there is a gating unit, which is configured by this register, indicating whether it is allowed to be shut down by the PMU. 1'b0: Allows the PMU to turn off the gating unit, thereby turning off the clock 1'b1: The PMU is not allowed to shut down the gate unit	1'b0

5.4.4 Software reset control register

Table 10 Software reset control register

bit access		Instructions	reset value
[31]	RW	soft_touch_rst_n Software reset touch_sensor module 0: reset	1'b1

		1: Reset release	
[30]	RW	soft_rst_flash_n Software Reset Flash Controller Module 0: reset 1: Reset release	1'b1
[29]	RW	soft_rst_bt_n Software reset BT module 0: reset 1: Reset release	1'b1
[28]	RW	soft_rst_qspi_ram_n Software reset qspi_ram module 0: reset 1: Reset release	1'b1
[27]	RW	soft_rst_sdio_m_n Software reset sdio_master module 0: reset 1: Reset release	1'b1
[26]	RW	soft_rst_gpsec_n software reset gpsec module 1'b0: Reset 1'b1: Reset release	1'b1
[25]	RW	soft_rst_rsa_n	1'b1

		Software reset RSA module 1'b0: reset 1'b1: Reset release	
[24]	RW	soft_rst_i2s_n Software reset i2s module 1'b0: reset 1'b1: Reset release	1'b1
[23]	RW	soft_rst_lcd_n software reset lcd module 1'b0: reset 1'b1: Reset release	1'b1
[22]	RW	soft_rst_pwm_n Software reset pwm module 1'b0: reset 1'b1: Reset release	1'b1
[21]	RW	soft_rst_sar_adc_n Software reset sar_adc module 1'b0: reset 1'b1: Reset release	1'b1
[20]	RW	soft_rst_timer_n Software reset timer module 1'b0: Reset	1'b1

		1'b1: Reset release	
[19]	RW	soft_rst_gpio_n software reset gpio module 1'b0: reset 1'b1: Reset release	1'b1
[18]	RW	soft_rst_rf_cfg_n Software reset configures **the register module of** RF (internal use, do not modify) 1'b0: Reset 1'b1: Reset release	1'b1
[17]	RW	soft_rst_spis_n software reset high speed spi module 1'b0: reset 1'b1: Reset release	1'b1
[16]	RW	soft_rst_spim_n Software reset low speed spi module 1'b0: reset 1'b1: Reset release	1'b1
[15]	RW	soft_rst_uart5_n Software reset on-chip uart5 module 1'b0: Reset 1'b1: Reset release	1'b1
[14]	RW	soft_rst_uart4_n	1'b1

		Software reset on-chip uart4 module 1'b0: reset 1'b1: Reset release	
[13]	RW	soft_rst_uart3_n Software reset on-chip uart3 module 1'b0: reset 1'b1: Reset release	1'b1
[12]	RW	soft_rst_uart2_n Software reset on-chip uart2 module 1'b0: reset 1'b1: Reset release	1'b1
[11]	RW	soft_rst_uart1_n Software reset on-chip uart1 module 1'b0: reset 1'b1: Reset release	1'b1
[10]	RW	soft_rst_uart0_n Software reset on-chip uart0 module 1'b0: reset 1'b1: Reset release	1'b1
[9]	RW	soft_rst_i2c_n Software reset on-chip i2c module 1'b0: reset	1'b1

		1'b1: Reset release	
[8]	RW	soft_rst_bus2_n Software reset on-chip bus2 module 1'b0: reset 1'b1: Reset release	1'b1
[7]	RW	soft_rst_bus1_n Software reset on-chip bus1 module 1'b0: reset 1'b1: Reset release	1'b1
[6]	RW	soft_rst_apb_n Software reset abp bridge module 1'b0: reset 1'b1: Reset release	1'b1
[5]	RW	soft_rst_mem_mng_n Software reset mem_mng **module (internal use, please do not modify)** 1'b0: reset 1'b1: Reset release	1'b1
[4]	RW	soft_rst_dma_n software reset dma module 1'b0: reset 1'b1: Reset release	1'b1
[3]	RW	soft_rst_sdio_ahb_n	1'b1

		Software reset sdio ahb clock domain module 1'b0: reset 1'b1: Reset release	
[2]	RW	soft_rst_sec_n **Software reset security module (internal use, do not modify)** 1'b0: reset 1'b1: Reset release	1'b1
[1]	RW	soft_rst_mac_n Software reset mac **module (internal use, please do not modify)** 1'b0: reset 1'b1: Reset release	1'b1
[0]	RW	soft_rst_bbp_n Software reset bbp **module (internal use, please do not modify)** 1'b0: reset 1'b1: Reset release	1'b1

5.4.5 Clock Divider Configuration Register

Table 11 Clock frequency division configuration register

bit access		Instructions	reset value
[31]	RW	divide_freq_en When it is necessary to reconfigure cpu_clk_divider, wlan_clk_divider, bus2_syncdn_factor, When sdadc_fdiv, set this register, the hardware will automatically update the above four parameters to the frequency divider, and then clear this register	1'b0

		register. 1'b0: frequency division factor is in effect 1'b1: Require hardware to update frequency division parameters Note: When configuring the frequency division factor here, when Divide_freq_en is valid, all factors must already be valid	
[30:28]		reserve	
[27:24] RW		Peripheral_divider 160M clock frequency division factor: Use DPLL as the clock source for frequency division. The frequency division coefficient is the assigned frequency division value. The divider output should be 160MHz. DPLL output is 480MHz and should be configured as 3.	4'h3
[23: 16] RW		bus2_syncdn_factor The clock ratio relationship between bus1 and bus2 should be N:1 Among them, N is an integer. In the actual adjustment, it mainly depends on the ratio of the operating frequency of the CPU to the clock frequency of bus2. Since the default cpu uses 80MHz clock and bus2 uses 40MHz clock, then N=2	8'h2
[15 : 8] RW		wlan_clk_divider The clock from the PLL is divided and sent to the wlan system. This register is the frequency division factor, the factor>=2. The default frequency division factor is 3, that is, the 480MHz output of pll is divided by 3, and the 160MHz clock is obtained as the root The node clock is sent to wlan (wlan continues to divide the frequency to obtain a more detailed low-frequency clock); Note 1: If the WLAN system needs to work normally, this clock needs to be fixed at 160MHz; if the WLAN system is turned off, Then this clock can be reduced in frequency to save power consumption. This clock must not be configured higher than 160MHz. Note 2: The secondary bus clock and APB clock are divided by four;	8'h3

[7 : 0] RW		cpu_clk_divider The clock from the PLL is divided and sent to the CPU. This register is the frequency division factor, the factor>=2. The default frequency division factor is 6, that is, after the reset is released, the 480MHz clock output by the PLL is divided by 6 and sent to the CPU is the 80MHz clock. When you need to adjust the clock required by the CPU, you can reconfigure this parameter	8'h6

5.4.6 Debug Control Register

Table 12 Clock Selection Register

bit access		Instructions	reset value
[16]	RW	JTAG enable 1'b0: disable JTAG debug function 1'b1: Enable JTAG debug function	1'b0
[15ÿ8] RW		sd_adc_div sigma-delta ADC clock division factor: Use 40MHz as the clock source for frequency division. The frequency division coefficient is the assigned frequency division value. After configuring this register, you must configure Divide_freq_en in the register clk_divider to take effect;	8'd10
[7]	RW	RSV	1'b0
[6]	RW	qflash_clk_sel QSPI_FLASH clock selection 1: use 80MHz;	1'b0

		0: use 40MHz;	
[5]	RW	gpsec_sel GPSEC clock selection 1: use 160MHz; 0: use 80MHz;	1'b0
[4]	RW	rsa_sel RSA clock selection 1: use 160MHz; 0: use 80MHz;	1'b0
[3:0] RW		**Reserve, do not modify**	4'd0

5.4.7 I2S Clock Control Register

Table 13 I2S clock control register

bit access		Instructions	reset value
[31:18]		reserve	
[17: 8] RW		BCLKDIV BCLK divider: F_BCLK = F_I2SCLK / BCLKDIV Note: If EXTAL_EN is not selected and internal PLL is used then F_I2SCLK = F_CPU (corresponds to CPU frequency same). Assuming F_CPU = 160MHz, F_I2SCLK = external crystal oscillator frequency when WXTAL_EN is enabled, **BCLKDIV = round (F_I2SCLK/(Fs*W*F))** Where Fs is the sampling frequency of the audio data, W is the word width;	10'b0

		F = 1 when the data is mono; F = 2 when the data is stereo. For example, if the internal PLL is used and the data width is 24 bits, the format is stereo and the sampling frequency is 128KHz, BCLKDIV should be configured as (160 * 10e6 / 128 * 10e3 * * 2ÿ= 10'h1aÿ 24	
[7 : 2] RW		MCLKDIV If an external clock is selected, this MCLK divider is used to generate the appropriate MCLK frequency. F_mclk = F_I2SCLK / (2 * MCLKDIV)ÿ When MCLKDIV = 0, F_I2SCLK is an external clock; When MCLKDIV >= 1, F_mclk = F_I2SCLK; Note: F_mclk should be configured as 256 * fs, where fs is the sampling frequency.	6'b0
[1]	RW	MCLKEN MCLK enable switch 1'b0: Disable MCLK 1'b1: enable MCLK	1'b0
[0]	RW	EXTAL_EN External clock selection, choose to use internal I2S block clock or external clock 1'b0: internal clock 1'b1: external clock Note: When using an external clock, the external clock must be 2 * N * 256 fs, where fs is the sampling frequency, N must Must be an integer.	1'b0

5.4.8 Reset Status Register

Table 14 Reset Status Register

bit access		Instructions	reset value
[31:18]		reserve	
[17: 8]	WHERE	CPU soft reset status clear Write 1 to clear CPU soft reset status.	1'b0
[7 : 2] WO		Wdog soft reset status clear Write 1 to clear Wdog Reset Status.	1'b0
[1]	RO	CPU Soft Reset Status 1: The CPU generates a soft reset; 0: The CPU does not generate a soft reset;	1'b0
[0]	RO	Wdog reset state 1: Wdog generated Reset; 0: Wdpg does not generate a Reset	1'b0

6 DMA modules

6.1 Function overview

DMA is used to provide high-speed data transfers between peripherals and memory, and between memory and memory. Can operate without any CPU

Move data quickly through DMA without any problem. The CPU resource saved in this way does not affect the operation of other instructions by the CPU.

The DMA is mounted on the AHB bus, supports up to 8 channels, 16 hardware peripheral request sources, and supports linked list structure and register control.

6.2 Main features

- ÿ Amba2.0 standard bus interface, 8 DMA channels
- ÿ Support DMA operation based on memory linked list structure
- ÿ Support 16 hardware peripheral request sources
- ÿ Support 1, 4-burst operation mode
- ÿ Support byte, half-word, word as unit transfer operation
- ÿ Supports source and destination addresses unchanged or sequentially incremented or configurable to operate in a predefined address range
- ÿ Support memory-to-memory, memory-to-peripheral, and peripheral-to-memory data transfer methods

6.3 Functional description

6.3.1 DMA channels

W800 supports a total of 8 DMA channels, DMA channels do not interfere with each other and can run simultaneously. Requesting different data streams can choose different

DMA channel.

Each DMA channel is allocated in a different register address offset segment, you can directly select the address segment of the corresponding channel for configuration and use. Do not

The register configuration method of the same channel is exactly the same.

Table 15 DMA address allocation

DMA base address	0x4000 0800
DMA_CH0	Offset (0x10~0x38)
DMA_CH1	Offset (0x40~0x68)
DMA_CH2	Offset (0x70~0x98)
...	...
DMA_CH7	Offset (0x160~0x188)

6.3.2 DMA data flow

Eight DMA channels enable a unidirectional data transfer link between source and destination.

The source and destination addresses of DMA can be set to constant, increment or cycle after each DMA operation:

ÿ DMA_CTRL[2:1] controls how the source address changes after each DMA operation;

ÿ DMA_CTRL[4:3] controls how the destination address changes after each DMA operation.

DMA can set the handling unit of byte, half-word, and word, and the final quantity of data to be transported is an integer multiple of the handling unit.

DMA_CTRL[6:5] to set.

DMA can set how many units of data are transported each time through burst, and select 1 or 4 units to be transported at a time through DMA_CTRL[7].

1-bit data, if DMA_CTRL[6:5] is set to word and burst is set to 4, then 4 words of data will be moved each time.

DMA can set the number of Bytes each time DMA transfer is started, the maximum is 65535 Bytes, set by DMA_CTRL[23:8].

6.3.3 DMA Cycling Mode

DMA cycle address mode means that after setting the source and destination addresses of DMA, after the data transfer reaches the set cycle boundary, it will jump to Loop start address, and execute in this way until the set transmission byte is reached.

The source and destination addresses of the loop address mode need to be set with the SRC_WRAP_ADDR and DEST_WRAP_ADDR registers, and passed WRAP_SIZE to set the length of the loop.

6.3.4 DMA transfer mode

DMA supports 3 transfer modes:

ÿ memory to memory

Both the source address and the destination address are configured as the memory address to be transferred, and DMA_MODE[0] is set to 0 in software mode.

ÿ memory to peripherals

The source address is set to the memory address, the destination address is set to the peripheral address, DMA_MODE[0] is set to 1, hardware mode, DMA_MODE[5:2] selects the peripheral used.

ÿ Peripherals to memory

The source address is set to the peripheral address, the destination address is set to the memory address, DMA_MODE[0] is set to 1, hardware mode, DMA_MODE[5:2] selects the peripheral used.

6.3.5 DMA Peripheral Selection

When using the transfer method from peripherals to memory or memory to peripherals, except that the corresponding peripherals need to be set to DMA TX or RX, DMA_MODE[5:2] also needs to select the corresponding peripheral.

Note: Because there are 3 UART ports, when UART uses DMA, it is also necessary to select the corresponding port through UART_CH[1:0].

UARTÿ

6.3.6 DMA linked list mode

DMA supports linked list mode of operation. Through the linked list mode, when we DMA transfers the memory data of the current linked list, we can advance to the next

After the DMA finishes moving the current linked list, it judges that the next linked list is valid, and can directly move the data of the next linked list.

The efficiency of cooperation between DMA and CPU can be effectively improved by means of linked list.

Linked list operation mode: set the DMA to work in the linked list mode through the DMA_MODE[1] register, and then set the DESC_ADDR register to

It is the starting address of the linked list structure, and then enable DMA through the CHNL_CTRL register. When the DMA process completes the move of the current memory

Finally, the software notifies the DMA that there is still valid data in the linked list by setting the valid flag, and the DMA processes the data according to the valid flag of the linked list.

A data to be moved.

6.3.7 DMA Interrupts

DMA transfer completion or burst can generate an interrupt, and the INT_MASK register can mask the corresponding interrupt of the DMA channel.

When the DMA corresponding interrupt is generated, the status of the current interrupt can be queried through the INT_SRC register, indicating what is currently generating the interrupt.

The corresponding status bit needs software to write 1 to clear 0.

6.4 Register Description

6.4.1 Register list

Table 16 DMA register list

offset address	name	abbreviation	access	describe	reset value
0X0000 interrupt	mask register	INT_MASK	RW	Set the DMA interrupt that needs to be masked	0X0000_FFFF
0X0004 interrupt	status register	INT_SRC	RW	indicates current DMA interrupt status	0X0000_0000
0X0008	DMA channel selection register	DMA_CH		Which UART to choose when RW UART peripheral	0X0000_0000

0X000C Reserved					
DMA CHNL0 registers					
0X0010	DMA Source Address Register	SRC_ADDR	Source address for RW DMA transfers		0X0000_0000
0X0014	DMA Destination Address Register	DEST_ADDR	Destination address for RW DMA transfer		0X0000_0000
0X0018	DMA cycle source start address register SRC_WRAP_ADDR RW DMA transfer source address in cycle mode 0X0000_0000				
0X001C	DMA cycle destination start address register device	DEST_WRAP_ADD R	DMA Transfer Destination in RW Cycle Mode site		0X0000_0000
0X0020	DMA Cycle Length Register	WRAP_SIZE	DMA cycle boundary in RW cycle mode 0X0000_0000		
0X0024	DMA Channel Control Register	CHNL_CTRL	RW Current channel DMA start and stop 0X0000_0000		
0X0028	DMA Mode Select Register	DMA_MODE	RW sets how DMA works		0X0000_0000
0X002C	DMA Flow Control Register	DMA_CTRL	RW Set DMA transfer data flow		0X0000_0000
0X0030	DMA Transfer Bytes Register	DMA_STATUS	RO Get the number of bytes currently transferred 0X0000_0000		
0X0034	DMA linked list entry address register DESC_ADDR		RW DMA linked list address entry address setting 0X0000_0000		
0X0038	DMA current destination address register CUR_DEST_ADDR RO Address of current DMA operation				0X0000_0000
DMA CHNL1 registers					
0X0040 - 0X0068	Same DMA CHNL0 registers				
DMA CHNL2 registers					
0X0070 - 0X0098	Same DMA CHNL0 registers				
DMA CHNL3 registers					
0X00A0 - same DMA CHNL0 registers					

0X00C8	
DMA CHNL4 registers	
0X00D0 - 0X00F8	Same DMA CHNL0 registers
DMA CHNL5 registers	
0X0100 - 0X0128	Same DMA CHNL0 registers
DMA CHNL6 registers	
0X0130 - 0X0158	Same DMA CHNL0 registers
DMA CHNL7 registers	
0X0160 - 0X0188	Same DMA CHNL0 registers

6.4.2 Interrupt mask register

Table 17 DMA interrupt mask register

bit access		Instructions	reset value
[31:16]		reserve	
[15]	RW	channel7 transfer_done interrupt mask, active high.	1'b1
[14]	RW	channel7 burst_done interrupt mask, active high.	1'b1
[13]	RW	channel6 transfer_done interrupt mask, active high.	1'b1
[12]	RW	channel6 burst_done interrupt mask, active high.	1'b1

[11]	RW	channel5 transfer_done interrupt mask, active high.	1'b1
[10]	RW	channel5 burst_done interrupt mask, active high.	1'b1
[9]	RW	channel4 transfer_done interrupt mask, active high.	1'b1
[8]	RW	channel4 burst_done interrupt mask, active high.	1'b1
[7]	RW	channel3 transfer_done interrupt mask, active high.	1'b1
[6]	RW	channel3 burst_done interrupt mask, active high.	1'b1
[5]	RW	channel2 transfer_done interrupt mask, active high.	1'b1
[4]	RW	channel2 burst_done interrupt mask, active high.	1'b1
[3]	RW	channel1 transfer_done interrupt mask, active high.	1'b1
[2]	RW	channel1 burst_done interrupt mask, active high.	1'b1
[1]	RW	channel0 transfer_done interrupt mask, active high.	1'b1
[0]	RW	channel0 burst_done interrupt mask, active high.	1'b1

6.4.3 Interrupt Status Register

Table 18 DMA interrupt status register

bit access		Instructions	reset value
[31:16]		reserve	
[15]	RW	channel7 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0
[14]	RW	channel7 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0
[13]	RW	channel6 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0
[12]	RW	channel6 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0
[11]	RW	channel5 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0

[10]	RW	channel5 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0
[9]	RW	channel4 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0
[8]	RW	channel4 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0
[7]	RW	channel3 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0
[6]	RW	channel3 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0
[5]	RW	channel2 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0
[4]	RW	channel2 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0
[3]	RW	channel1 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0
[2]	RW	channel1 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0
[1]	RW	channel0 transfer_done interrupt status, write 1 to clear 0. An interrupt is generated when the DMA transfer is complete.	1'b0
[0]	RW	channel0 burst_done interrupt status, write 1 to clear 0. DMA burst completion generates an interrupt.	1'b0

6.4.4 UART select register

Table 19 UART selection register

bit access		Instructions	reset value
[31: 24]		reserve	
[23:8] RW		dma req clear: Writing 1 to each bit clears the corresponding dma req request. Self-cleaning. For example, writing 1 to bit 23 will clear the 15th corresponding dma request in dma_sel; Writing 1 to bit 8 will clear the 0th dma request-uart_rx_req in dma_sel;	16'd0
[2 : 0] RW		Uart dma channel selection: 3'd0: uart0 module dma channel access dma	3'h0

		3'd1: uart1 module dma channel access dma 3'd2: uart2/7816 module dma channel access dma 3'd3: uart3 module dma channel access dma 3'd4: uart4 module dma channel access dma 3'd5: uart5 module dma channel access dma	

6.4.5 DMA source address register

Table 20 DMA source address register

bit access		Instructions	reset value
[31: 0] RW		In acyclic mode, the source address of DMA transfer, peripheral address or memory address	32'h0

6.4.6 DMA Destination Address Register

Table 21 DMA destination address register

bit access		Instructions	reset value
[31: 0] RW		In acyclic mode, the destination address of DMA transfer, peripheral address or memory address	32'h0

6.4.7 DMA cycle source start address register

Table 22 DMA cycle source start address register

bit access		Instructions	reset value
[31: 0] RW		In loop mode, the starting address of the source address of DMA transfer, peripheral address or memory address	32'h0

6.4.8 DMA Cycle Destination Start Address Register

Table 23 DMA cycle destination start address register

bit access		Instructions	reset value
[31: 0] RW		In loop mode, the starting address of the destination address of DMA transfer, peripheral address or memory address	32'h0

6.4.9 DMA Cycle Length Register

Table 24 DMA cycle length register

bit access		Instructions	reset value
[31:16] RW		In cycle mode, the DMA destination address cycle length. DMA increments the transfer data sequentially from the start address, and when the number of transfer data bytes reaches the set value, it will jump Go to the cycle start address, and continue to move data from the start address	16'h0
[15: 0] RW		In cycle mode, DMA source address cycle length.	16'h0

6.4.10 DMA channel control register

Table 25 DMA channel control register

bit access		Instructions	reset value
[31: 2]		reserve	
[1]	RW	dma_stop Stop dma operation, active high. DMA will stop after finishing the current burst operation and clear chnl_on at the same time. The software should be based on A chnl_on of 0 ensures that dma has been completely stopped.	1'b0
[0]	RW	chnl_on	1'b0

		Start current channel DMA conversion, active high. Automatically cleared to 0 after conversion or setting stops after Dma is completed.	

6.4.11 DMA mode selection register

Table 26 DMA mode selection register

bit access		Instructions	reset value
[31: 7]		reserve	
[6]	RW	chain_link_en Valid in linked list mode, indicating whether dma will continue to read and process subsequent links after processing the first linked list surface. If it is 1, update the next_desc_addr in the linked list, and continue to read the next linked list until the linked list Among them, vld is 0; if it is 0, it will stop after processing the current linked list.	1'b0
[5 : 2] RW		dma_sel Selection of 16 dma_reqs. 4'd0ÿuart rx dma req 4'd1ÿuart tx dma req 4'd2ÿpwm_cap0_req 4'd3ÿpwm_cap1_req 4'd4ÿLS_SPI rx dma req 4'd5ÿLS_SPI tx dma req 4'd6ÿSD_ADC chnl0 req 4'd7ÿSD_ADC chnl1 req	4'h0

		4'd8ÿSD_ADC chnl2 req 4'd9ÿSD_ADC chnl3 req 4'd10: I2S RX req 4'd11: I2S TX req 4'd12: SDIO_HOST req	
[1]	RW	chain_mode 1'b0: use normal mode 1'b1: use linked list mode	1'b0
[0]	RW	dma_mode 1'b0: software mode. 1'b1: hardware method.	1'b0

6.4.12 DMA data flow control register

Table 27 DMA data flow control register

bit access		Instructions	reset value
[31:24]		reserve	
[23: 8] RW		total_byte The total number of bytes to be operated. It needs to be consistent with the data_size configuration, that is, if it is a word operation, then It should be configured as an integer multiple of 4; if it is half-word operation, it should be configured as an integer multiple of 2.	16'h0
[7]	RW	burst_size Set how many units of data the DMA moves each time 1'b0: burst is 1	1'b0

		1'b1: burst is 4 When the last burst size exceeds the number of remaining transfers, use the burst size as the size of the remaining data Small.	
[6 : 5] RW		data_size Set up handling units for DMA 2'h0ÿchange 2'h1ÿhalf_word 2'h2ÿword 2'h3: reserved	2'h0
[4 : 3] RW		dest_addr_inc 2'h0: The destination address remains unchanged after each operation; 2'h1: The destination address is automatically accumulated after each operation. 2'h2: Reserved 2'h3: Loop operation, the destination address is automatically accumulated after each operation, and jumps to the start of the loop when it reaches the defined loop boundary start address.	2'h0
[2 : 1] RW		src_addr_inc 2'h0: The source address remains unchanged after each operation; 2'h1: The source address is automatically accumulated after each operation. 2'h2: Reserved 2'h3: Loop operation, the source address is automatically accumulated after each operation, and jumps to the beginning of the loop when it reaches the defined loop boundary address.	2'h0
[0]	RW	auto_reload	1'b0

		After the current DMA transfer is completed, the next DMA transfer will be performed automatically according to the current DMA configuration.	

6.4.13 DMA Transfer Bytes Register

Table 28 DMA Transfer Bytes Register

bit access		Instructions	reset value
[31:16]		reserve	
[15: 0] RW		transfer_cnt The number of bytes currently transferred. Every time dma is restarted (chnl_on is set to 1), it is cleared to 0 and starts counting again.	16'h0

6.4.14 DMA linked list entry address register

Table 29 DMA linked list entry address register

bit access		Instructions	reset value
[31: 0] RW		desc_addr When the linked list is enabled, it is used as the entry address of the linked list. After each transmission of a linked list is completed, the base of the next linked list The address is updated to this register.	32'h0

6.4.15 DMA Current Destination Address Register

Table 30 DMA Current Destination Address Register

bit access		Instructions	reset value
[31: 0] RO		current_dest_addr Current DMA operation destination address.	32'h0

		When the software stops dma, you can know the destination address that dma will operate by checking this register.	

7 General hardware encryption module

7.1 Function overview

The encryption module automatically completes the encryption of the source address space data of the specified length, and automatically writes the encrypted data back to the specified destination address space after completion.

between; support SHA1/MD5/RC4/DES/3DES/AES/CRC/TRNG.

7.2 Main features

- ÿ Support SHA1/MD5/RC4/DES/3DES/AES/CRC/TRNG encryption algorithm
- ÿ DES/3DES supports both ECB and CBC modes
- ÿ AES supports three modes: ECB, CBC and CTR
- ÿ CRC supports CRC8, CRC16_MODBUS, CRC16_CCITT and CRC32 four modes
- ÿ CRC supports input/output reverse
- ÿ SHA1/MD5/CRC supports continuous multi-packet encryption
- ÿ Built-in true random number generator, also supports seed seed to generate pseudo-random numbers

7.3 Functional description

7.3.1 SHA1 encryption

The hardware SHA1 calculation can be performed on continuous multi-packet data, and the calculation result is stored in the register, and the encrypted result of the previous packet can be used as the next packet

initial value.

7.3.2 MD5 encryption

The hardware MD5 calculation can be performed on continuous multi-packet data, and the calculation result is stored in the register. The encryption result of the previous packet can be used as the initial packet of the next packet.

initial value.

7.3.3 RC4 encryption

Supports RC4 encryption and decryption.

7.3.4 DES encryption

Support DES encryption and decryption, support ECB and CBC two modes.

7.3.5 3DES encryption

Support 3DES encryption and decryption, support ECB and CBC two modes.

7.3.6 AES encryption

Support AES encryption and decryption, support ECB, CBC and CTR three modes.

7.3.7 CRC encryption

The hardware CRC calculation can be performed on continuous multi-packet data, and the calculation result is stored in the register. The encryption result of the previous package can be used as the initial packet of the next package.

Initial value, supports CRC8, CRC16_MODBUS, CRC16_CCITT and CRC32 four modes, supports input/output reverse.

The calculation formula of CRC32 is as follows:

1ÿCRC-32: 0x04C11DB7

X32+X26+X23+X22+X16+X12+X11+X10+X8+X7+X5+X4+X2+X+1

Commonly used in ZIP, RAR, IEEE 802 LAN/FDDI, IEEE 1394, PPP-FCS and other protocols.

2. CRC-16: supports two kinds of polynomials

2.1: 0X1021

X16 + X12 + X5 + 1

Commonly used in ISO HDLC, ITU X.25, V.34/V.41/V.42, PPP-FCS CCITT and other protocols.

2.2: 0X8005

X16 + X15 + X2 + 1

Commonly used in USB, ANSI X3.28, SIA DC-07 and other protocols.

3ÿCRC-8ÿ0X207

x8+x2+x1+1

7.3.8 TRNG random number generator

A true random number generator module is integrated in the W800 system. Divided into analog module and digital post-processing module. The analog module outputs a random clock ad_trng_clks and random number ad_trng_dout, the digital post-processing module is used to eliminate the deviation and autocorrelation of random numbers. related control Control registers are in the GPSEC register list.

The basic operation process is as follows:

1. Enable TRNG_EN, set TRNG_SEL to 1, make GPSEC register 0x48 display the output value of TRNG. At this time mode

   The analog module starts to output random clock and random signal. The signal obtained by the first 8 clock samples is used as the initial state of the LFSR to initialize the LFSR Chain, the data obtained by each random clock sample is post-processed by XOR CHAIN and LFSR registers, shifted and stored in the register in TRNG_RANDOM.

2. Software can read random value through GPSEC register 0x48. Digital post-processing when the register TRNG_DIG_BYPASS is set to 1

   The module stops working, and directly stores the output value of the analog module into the result register TRNG_RANDOM.

7.4 Register description

7.4.1 Register list

Table 31 Encryption module register list

offset address	name	abbreviation	access	describe	reset value
0X0000 source address register		SRC_ADDR	RW RC4/SHA1/AES/DES/3DES/CRC/MD 5 multiplexing source address		0X0000_0000
0X0004 destination address register		DEST_ADDR	RW RC4/AES/DES/3DES multiplexing destination address		0X0000_0000
0X0008 configuration register		GPSEC_CFG	RW general hardware encryption module configuration register		0X0000_0000
0X000C Control register		GPSEC_CTRL	RW general hardware encryption module control register		0X0000_0000
0X0010 Secret key 0 low register KEY00			RW Key0 lower 32 bits first input key (RC4/AES/DES/3DES), multiplexing CRC There		0X0000_0000
0X0014 Key 0 high register KEY01			RW Key0 high 32-bit first input key ÿRC4/AES/DES/3DESÿ		0X0000_0000
0X0018 Secret key 1 low register KEY10			RW Key1 lower 32-bit second input key ÿRC4/AES//3DESÿ		0X0000_0000
0X001C Key 1 high register KEY11			RW Key1 high 32-bit second input key ÿRC4/AES//3DESÿ		0X0000_0000
0X0020 Secret key 2 low register		KEY20	RW Key2 The third input key of the lower 32 bits (3DES), multiplexing iv1 lower 32-bit input initial Vector (AES)		0X0000_0000

0X0024 Secret key 2 high register		KEY21	RW Key2	High 32-bit third input key (3DES), multiplexing iv1 high 32-bit input initial Vector (AES)	0X0000_0000
0X0028	Initial vector 0 low register device	IV00	RW IV0	Low 32-bit input initial vector ÿAES/DES/3DESÿ	0X0000_0000
0X002C	Initial vector 0 high register device	IV01	RW IV0	high 32-bit input initialization vector ÿAES/DES/3DESÿ	0X0000_0000
0X0030 status register		GPSEC_STS	RW	general hardware encryption module status register	0X0000_0000
0X0034 Digest 0 Register		SHA1-DIGEST0	RW	sha1-digest0/MD5-digest0	0X6745_2301
0X0038 Summary 1 Register		SHA1 DIGEST1	RW	sha1-digest1/MD5-digest1	0XEFCD_AB89
0X003C Summary 2 Register		SHA1-DIGEST2	RW	sha1-digest2/MD5-digest2	0X98BA_DCFE
0X0040 Summary 3 Register		SHA1-DIGEST3	RW	sha1-digest3/MD5-digest3	0X1032_5476
0X0044 Summary 4 Register		SHA1-DIGEST4	RW	sha1-digest4/CRC	0XC3D2_E1F0
0X0048 RNG_result		RNG_RESULT	RW	RNG output	0X0000_0000
0X004C	Key 3 low register Key30		RW Key3	The lower 32 bits of the third input key (of RC4 256bit mode)	0X0000_0000
0X0050	Key 3 low register Key31		RW Key3	High 32-bit third input key (RC4's 256bit mode)	0X0000_0000
0X0054 TRNG configuration		PRINCIPAL_CROSS	RW	True random number generator configuration selection	0X40

7.4.2 Configuration registers

Table 32 Encryption module configuration register

bit access		Instructions	reset value
[31]	RW	RC4_128_256 0: Flag RC4 encryption/decryption is carried out according to the block length of 128bit; 1: Flag RC4 encryption/decryption is carried out according to the block length of 256bit.	1'b0
[30]	RW	RNG start 1'b0: do not start RNG 1'b1: start RNG	1'b0
[29]	RW	RNG Load_seed Hardware automatically clears to 0 1'b0: The random number generator will use zero as the seed by default to generate a random number with the corresponding number of digits 1'b1: Start generating random numbers after the seed loading is complete	1'b0
[28]	RW	RNG switch Controls the number of bits to generate random numbers, 1'b0: 16 bits 1'b1: 32 bits	1'b0
[27]	RW	des_soft_reset des The hardware automatically clears 0 after the soft reset is completed 1'b0: do not generate a soft reset and do not change the current state 1'b1: The encryption algorithm is reset to the initial state by software	1'b0
[26]	RW	aes_soft_reset After the aes soft reset is completed, the hardware automatically clears to 0 1'b0: do not generate a soft reset and do not change the current state	1'b0

		1'b1: The encryption algorithm is reset to the initial state by software	
[25]	RW	rc4_soft_reset rc4 hardware automatically clears 0 after soft reset is completed 1'b0: do not generate a soft reset and do not change the current state 1'b1: The encryption algorithm is reset to the initial state by software	1'b0
[24]	RW	crc_datarev 1'b0: CRC input data is not reversed 1'b1: CRC input data reversed	1'b0
[23]	RW	crc_chksrev 1'b0: CRC output result is not inverted 1'b1: CRC output result inverted	1'b0
[22:21] RW		sub_mode Algorithm type sub-pattern selection: 0: ECB mode of DES/AES encryption algorithm, CRC8 mode of reusable CRC algorithm 1: CBC of DES/AES encryption algorithm, CRC16_0 mode of CRC algorithm can be reused 2: CTR mode of AES encryption algorithm, CRC16_1 mode of CRC algorithm can be reused 3: MAC mode of AES encryption algorithm, CRC32 of CRC algorithm can be reused	2'b0
[20]	RW	encrypt_decrypt Encryption or decryption mode selection for RC4/AES/DES/3DES algorithm: 1'b0: encrypted 1'b1: Decryption	1'b0
[19]	RW	gpsec_int_mask	1'b0

		1'b0: Do not mask encryption/decryption completion interrupt 1'b1: Shield encryption/decryption completion interrupt	
[18:16] RW		cypher_mode Cryptographic Algorithm Type 3'b000ÿRSV 3'b001ÿRC4 3'b010: SHA1 3'b011ÿAES 3'b100ÿDES 3'b101ÿ3DES 3'b110ÿCRC 3'b111ÿMD5	3'b0
[15: 0] RW		total_byte The total number of bytes required for encryption and decryption operations.	16'h0

7.4.3 TRNG Control Register

Table 33 TRNG module control register

bit access		Instructions	reset value
[31: 7]		reserve	
[6]	RW	TRNG_INT_MASK TRNG interrupt mask	1'b1

		0: TRNG module reports interruption; 1: TRNG module does not report interrupt;	
[5:3] RW		TRNG_CP TRNG module control signal	3'd0
[2]	RW	TRNG_DIG_BYPASS TRNG digital post-processing module bypass: 0: TRNG module performs post-processing; 1: The TRNG module does not perform post-processing;	1'b0
[1]	RW	TRNG_SEL: RNG output selection signal. 0: Register 0x48 displays the output result of the pseudo-random module; 1: Register 0x48 displays the output result of TRNG;	1'b0
[0]	RW	TRNG_EN: TRNG module enable signal. High effective. 0: TRNG module stops; 1: TRNG module starts working;	1'b0

7.4.4 Control registers

Table 34 Encryption module control register

bit access		Instructions	reset value
[31: 2]		reserve	

[1]	RW	sec_stop Stop the current encryption and decryption operations 1'b0: invalid 1'b1: encryption/decryption stopped	1'b0
[0]	RW	sec_strt Start encryption and decryption, and after the number of bytes in the encryption and decryption operation is completed, the hardware will automatically clear to 0 1'b0: do not start encryption/decryption 1'b1: start encryption/decryption	1'b0

7.4.5 Status Register

Table 35 Encryption Module Status Register

bit access		Instructions	reset value
[31: 17]		reserve	
[16]	RW	int_flag Software write 1 to clear 1'b0: Do not generate encryption/decryption completion interrupt 1'b1: generate encryption/decryption completion interrupt	1'b0
[15: 0] RO		transfer_cnt The number of bytes currently encrypted. It is cleared to 0 each time the encryption and decryption are restarted, and restarts counting.	16'h0

8 RSA encryption module

8.1 Function overview

RSA operation hardware coprocessor, providing Montgomery (FIOS algorithm) modular multiplication operation function. Cooperate with RSA software library to realize RSA algorithm Law. Supports 128-bit to 2048-bit modular multiplication.

8.2 Main features

ÿ Support 128-bit to 2048-bit modular multiplication (the length of the modular multiplication is an integer multiple of 32 bits)

ÿ Support D*D; X*Y; D*Y; X*X and other 4 modular multiplication modes

8.3 Functional description

8.3.1 Modular multiplication function

RSA operation hardware coprocessor, providing Montgomery (FIOS algorithm) modular multiplication operation function. Implement RSA together with the RSA software library algorithm. Supports 128-bit to 2048-bit modular multiplication.

8.4 Register Description

8.4.1 Register list

Table 36 RSA register list

offset address name		abbreviated	access	describe	reset value
0X0000~0X00FC	Data X register	XBUF	RW Data	X Register	
0X0100~0X01FC	Data Y register	YBUF	RW Data	Y Register	
0X0200~0X02FC	data M register	MBUF	RW Data	M Register	
0X0300~0X03FC	Data D register	DBUF	RW Data	D Register	

0X0400	RSA Control Register RSACON	RW		RSA Control Register	0X0000_0000
0X0404	Parameter MC register	RSAMC	WO	parameter MC register	0X0000_0000
0X0408	Parameter N Register	RSAN	RW	parameter N register	0X0000_0000

8.4.2 Data X register

XBUF corresponds to the buffer of data X (2048bit), and the corresponding haddr value is 0000h~00fch. The corresponding rules are as follows:

Table 37 RSA Data X Register

000h	004h	008h	.....	00f8h	00fch
X[31:0]	X[63:32]	X[95:64]	.....	X[2015:1984]	X[2047:2016]

8.4.3 Data Y Register

YBUF corresponds to the buffer of data Y (2048bit), and the corresponding haddr value is 0100h~01fch. The corresponding rules are as follows:

Table 38 RSA Data Y Register

0100h	0104h	0108h	.....	01f8h	01fch
AND[31:0]	Y[63:32]	Y[95:64]	.....	AND[2015:1984]	AND[2047:2016]

8.4.4 Data M Register

MBUF corresponds to the buffer of data M (2048bit), and the corresponding haddr value is 0200h~02fch. The corresponding rules are as follows:

Table 39 RSA data M register

0200h	0204h	0208h	.....	02f8h	02fch
M[31:0]	M[63:32]	M[95:64]	.....	M[2015:1984]	M[2047:2016]

8.4.5 Data D register

DBUF corresponds to the buffer of data D (2048bit), and the corresponding haddr value is 0300h~03fch. The corresponding rules are as follows:

Table 40 RSA Data D Register

0300h	0304h	0308h	.....	03f8h	03fch
D[31:0]	D[63:32] D[95:64]			D[2015:1984]	D[2047:2016]

8.4.6 RSA Control Register

RSACON, RSA control register, the actual physical space is a 32bit register.

Table 41 RSA Control Register

bit access		Instructions	reset value
[31: 6]		reserve	
[5]	RW	Modulo multiplication enable control bit. The software writes "1" to start the modular multiplication operation, and the hardware automatically clears "0" after the operation is completed. 1'b0	
[4]	RW	Provide soft reset function, high effective. The software writes "1" for soft reset, after the reset is completed, the hardware automatically clears "0"ÿ 1. Set parameters MC and N to 0. 2. After starting the modular multiplication (bit5 is set to 1), setting this bit to "1" will terminate the current operation (when bit0 changes to High, indicating that the execution of the soft reset command is completed and the operation is terminated), but the internal data buffer (X, Y, M, D) Part of the calculation results that have been completed in will be retained.	1'b0
[2 : 3] RW		Modular multiplication mode selection. 2'b00ÿX = D*D mod M 2'b01ÿD = X*Y mod M 2'b10ÿX = D*Y mod M 2'b11ÿD = X*X mod M	2'b0
[1]	RW	reserve	1'b0

[0]	RW	Modular multiplication operation completed flag, high effective. Set to "1" by hardware and clear to "0" by software. Writing "1" by software is invalid. 1'b0	

8.4.7 Parameter MC register

Table 42 RSA parameter MC register

bit access		Instructions	reset value
[31:0] WO		RSAMC corresponds to parameter MC (32bit). The reset value is all 0. The read value is all 0s.	32'h0

8.4.8 Parameter N Register

RSAN corresponds to parameter N (7bit). The N value is the modulo length divided by 32. That is, if you call the 1024bit modular multiplication operation, you need to set N =32. When writing to this register, the lower 7 bits are valid data, and when reading, the upper 25 bits are 0. The reset value is all 0.

Table 43 RSA parameter N register

bit access		Instructions	reset value
[31: 7]		reserve	
[6 : 0] RW		RSAN corresponds to parameter N (7bit). The N value is the modulo length divided by 32.	7'h0

9 GPIO modules

9.1 Function overview

The GPIO controller realizes the configuration of GPIO attributes by software, enabling users to operate GPIO conveniently.

Each GPIO can be individually configured by software, set it as an input port, output port, set its suspension, pull-up, pull-down state,

Set its rising edge, falling edge, double edge, high level, low level interrupt trigger mode.

9.2 Main Features

- ÿ Support GPIO software configuration
- ÿ Support GPIO interrupt configuration
- ÿ Up to 48 GPIO available

9.3 Functional description

The GPIO provided in W800 is divided into two groups, one group is GPIOA, the other group is GPIOB, the starting addresses of GPIOA and GPIOB registers are different.

Same, but same function.

When the user wants to use a specific IO as a software-controlled GPIO, set the corresponding position in the GPIO multiplexing selection register to 0.

Can.

The GPIO direction control register is used to control the direction of the GPIO, 1 means that the corresponding GPIO is used as an output pin, and 0 means that the corresponding GPIO

as an input pin.

The GPIO pull-up and pull-down control register is used to control the pull-up and pull-down function of the corresponding IO.

The GPIO pull-up control register is active low, setting it to 0 means enabling the pull-up function of the corresponding IO, and setting it to 1 means turning off the pull-up function.

For the attributes of IO, please refer to the IO multiplexing table.

The GPIO pull-down control register is active high, setting it to 1 means enabling the pull-down function of the corresponding IO, and setting it to 0 means turning off the pull-down function.

For the attributes of IO, please refer to the IO multiplexing table.

When the GPIO data register is set to the input state, it indicates the level of the input IO, and when it is set to the output state, it can be specified by writing 1 or 0

IO output level. This register is controlled by the GPIO data enable register, only when the GPIO data enable register is set to 1

At this time, the GPIO data register can be read and written.

The GPIO module provides input signal detection function. By configuring the registers related to GPIO interrupts, high and low level detection and up and down edges can be realized

jump detection. When the input signal corresponding to the IO meets the pre-set conditions, such as high level trigger or rising edge trigger, etc., it will trigger

GPIO interrupt is reported to MCU for processing. The MCU needs to clear the corresponding interrupt status to avoid false triggering of the interrupt.

9.4 Register Description

9.4.1 Register list

Table 44 GPIOA register list

offset address	name	abbreviated access		describe	reset value
0X0000	GPIO data register	GPIO_DATA	RW	Read and write GPIO current data	0X180B
0X0004	GPIO Data Enable Register	GPIO_DATA_EN	RW	configures the enable bit of GPIO_DATA	0XFFFF
0X0008	GPIO direction control register	GPIO_DIR	RW	configures GPIO direction	0X0000
0X000C	GPIO pull-up control register	GPIO_PULL_EN	RW	Configure GPIO pull-up	0XFFFF
0X0010	GPIO Multiplexing Select Register	GPIO_AF_SEL	RW	Configure GPIO alternate function enable bit	0XFFFF

0X0014	GPIO multiplexing selection register 1	GPIO_SF_S1	RW	GPIO alternate function selection bit high address bit	0X0000
0X0018	GPIO multiplexing selection register 0	GPIO_AF_S0	RW	GPIO alternate function selection bit low address bit	0X0000
0X001C	GPIO pull-down control register	GPIO_DN_ENA	RW	Configure GPIO pull-down	0X0000
0X0020	GPIO interrupt trigger configuration register GPIO_IS		RW	configures the interrupt trigger mode of GPIO	0X0000
0X0024	GPIO interrupt edge trigger mode configuration register memory	GPIO_IBE	RW	Configure GPIO interrupt edge trigger mode	0X0000
0X0028	GPIO interrupt upper and lower edge trigger configuration register memory	GPIO_IEV	RW	configure GPIO interrupt upper and lower edge trigger or high and low level trigger	0X0000
0X002C	GPIO interrupt enable configuration register GPIO_IE		RW	configure GPIO interrupt enable	0X0000
0X0030	GPIO Raw Interrupt Status Register	GPIO_RIS	RO	query GPIO raw interrupt status (before MASK)	0X0000
0X0034	GPIO Masked Interrupt Status Register GPIO_MIS		RO	Query the interrupt status after GPIO masking (MASK back)	0X0000
0X0038	GPIO Interrupt Clear Control Register GPIO_IC		WO	controls GPIO interrupt clearing	0X0000

Table 45 GPIOB register list

offset address	name	abbreviated access		describe	reset value
0X0000	GPIO data register	GPIO_DATA	RW	Read and write GPIO current data	0X0000_7304
0X0004	GPIO Data Enable Register	GPIO_DATA_EN	RW	Configure the enable bit of GPIO_DATA	0X7FFF_FFFF
0X0008	GPIO direction control register	GPIO_DIR	RW	configures GPIO direction	0X0000_0000
0X000C	GPIO pull-up control register	GPIO_PULL_EN	RW	Configure GPIO pull-up	0XFFFF_FFFF

0X0010	GPIO Multiplexing Select Register	GPIO_AF_SEL	RW	Configure GPIO alternate function enable bit	0XFFFF_FFFF
0X0014	GPIO multiplexing selection register 1	GPIO_SF_S1	RW	GPIO multiplexing function selection bit high address bit	0X0000_0000
0X0018	GPIO multiplexing selection register 0	GPIO_AF_S0	RW	GPIO alternate function selection bit low address bit	0X0000_0000
0X001C	GPIO pull-down control register	GPIO_DN_ENA	RW	Configure GPIO pull-down	0X0000_0000
0X0020	GPIO interrupt trigger configuration register	GPIO_IS	RW	Configure GPIO interrupt trigger mode	0X0000_0000
0X0024	GPIO interrupt edge trigger mode configuration register memory	GPIO_IBE	RW	Configure GPIO interrupt edge trigger mode	0X0000_0000
0X0028	GPIO interrupt upper and lower edge trigger configuration register memory	GPIO_IEV	RW	configure GPIO interrupt up and down edge trigger or High and low level trigger	0X0000_0000
0X002C	GPIO interrupt enable configuration register	GPIO_IE	RW	configure GPIO interrupt enable	0X0000_0000
0X0030	GPIO Raw Interrupt Status Register	GPIO_RIS	RO	query GPIO raw interrupt status (MASK forward)	0X0000_0000
0X0034	GPIO Masked Interrupt Status Register	GPIO_MIS	RO	Query the interrupt status after GPIO masking (after MASK)	0X0000_0000
0X0038	GPIO Interrupt Clear Control Register	GPIO_IC	WO	controls GPIO interrupt clearing	0X0000_0000

9.4.2 GPIO Data Register

Table 46 GPIOA data register

bit access		Instructions	reset value
[15: 0]	RW	GPIO current data, each BIT corresponds to the corresponding GPIO line	16'h180b

Table 47 GPIOB data register

bit access		Instructions	reset value
[31: 0]	RW	GPIO current data, each BIT corresponds to the corresponding GPIO line	32'h7304

9.4.3 GPIO Data Enable Register

Table 48 GPIOA data enable register

bit access		Instructions	reset value
[15: 0]	RW	Corresponding to the BIT enable bit of GPIO_DATA, only when the corresponding BIT is 1, the operation of the corresponding bit of GPIO_DATA It is effective only if the operation is performed, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO_DATA[x] cannot be read or written [x] = 1, GPIO_DATA[x] can be read and written	16'hffff

Table 49 GPIOB data enable register

bit access		Instructions	reset value
[31: 0]	RW	Corresponding to the BIT enable bit of GPIO_DATA, only when the corresponding BIT is 1, the operation of the corresponding bit of GPIO_DATA It is effective only if the operation is performed, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO_DATA[x] cannot be read or written [x] = 1, GPIO_DATA[x] can be read and written	32'h7fff_ffff

9.4.4 GPIO direction control register

Table 50 GPIOA direction control register

bit access		Instructions	reset value
[15: 0]	RW	GPIO direction control, each BIT corresponds to the corresponding GPIO line, 1'bx:	16'h0

		[x] = 0, GPIO[x] is input [x] = 1, GPIO[x] is output	

Table 51 GPIOB direction control register

bit access		Instructions	reset value
[31: 0]	RW	GPIO direction control, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] is input [x] = 1, GPIO[x] is output	32'h0

9.4.5 GPIO pull-down control register

Table 52 GPIOA pull-up control register

bit access		Instructions	reset value
[15: 0]	RW	GPIO pull-up control, each BIT corresponds to the corresponding GPIO line, 1'bx: Note: This register is active low [x] = 0, GPIO[x] has pull-up [x] = 1, GPIO[x] has no pull-up	16'hffff

bit access		Instructions	reset value
[15: 0]	RW	GPIO pull-down control, each BIT corresponds to the corresponding GPIO line, 1'bx: Note: This register is active high [x] = 1, GPIO[x] has pull-down [x] = 0, GPIO[x] no pull-down	16'h0000

Table 53 GPIOB pull-down control register

bit access		Instructions	reset value
[31: 0]	RW	GPIO pull-up control, each BIT corresponds to the corresponding GPIO line, 1'bx: Note: This register is active low [x] = 0, GPIO[x] has pull-up [x] = 1, GPIO[x] has no pull-up	32'hffff_ffff

bit access		Instructions	reset value
[31: 0]	RW	GPIO pull-down control, each BIT corresponds to the corresponding GPIO line, 1'bx: Note: This register is active high [x] = 1, GPIO[x] has pull-down [x] = 0, GPIO[x] no pull-down	32'h0000_0000

9.4.6 GPIO multiplexing selection register

Table 54 GPIOA multiplexing selection register

bit access		Instructions	reset value
[15: 0]	RW	GPIO multiplexing function enable bit, each BIT corresponds to whether the corresponding GPIO multiplexing function is enabled, 1'bx:	16'hffff

		[x] = 0, GPIO[x] multiplexing function is disabled [x] = 1, GPIO[x] multiplexing function is enabled When [x] = 1, the alternate function depends on the corresponding BIT of the two registers GPIO_AF_S1 and GPIO_AF_S0 status. S1.[x] = 0, S0.[x] = 0, alternate function 1 (opt1) S1.[x] = 0, S0.[x] = 1, alternate function 2 (opt2) S1.[x] = 1, S0.[x] = 0, alternate function 3 (opt3) S1.[x] = 1, S0.[x] = 1, alternate function 4 (opt4) [x] = 0, if GPIO_DIR[x] = 0, and GPIO_PULL_EN[x] = 1, the GPIO is multiplexed as opt6 Analog IO function For the IO multiplexing function, please refer to the chip pin multiplexing relationship	

Table 55 GPIOB multiplexing selection register

bit access		Instructions	reset value
[31: 0]	RW	GPIO multiplexing function enable bit, each BIT corresponds to whether the corresponding GPIO multiplexing function is enabled, 1'bx: [x] = 0, GPIO[x] multiplexing function is disabled [x] = 1, GPIO[x] multiplexing function is enabled When [x] = 1, the alternate function depends on the corresponding BIT of the two registers GPIO_AF_S1 and GPIO_AF_S0 status. S1.[x] = 0, S0.[x] = 0, alternate function 1 (opt1)	32'hffff_ffff

		S1.[x] = 0, S0.[x] = 1, alternate function 2 (opt2) S1.[x] = 1, S0.[x] = 0, alternate function 3 (opt3) S1.[x] = 1, S0.[x] = 1, alternate function 4 (opt4) [x] = 0, if GPIO_DIR[x] = 0, and GPIO_PULL_EN[x] = 1, the GPIO is multiplexed as opt6 Analog IO function For the IO multiplexing function, please refer to the chip pin multiplexing relationship	

9.4.7 GPIO multiplexing selection register 1

Table 56 GPIOA multiplexing selection register 1

bit access		Instructions	reset value
[15: 0]	RW	GPIO multiplexing function selection bit high address bit, together with GPIO_AF_S0 determine the multiplexing function For the IO multiplexing function, please refer to the chip pin multiplexing relationship	16'h0

Table 57 GPIOB multiplexing selection register 1

bit access		Instructions	reset value
[31: 0]	RW	GPIO multiplexing function selection bit high address bit, together with GPIO_AF_S0 determine the multiplexing function For the IO multiplexing function, please refer to the chip pin multiplexing relationship	32'h0

9.4.8 GPIO multiplexing selection register 0

Table 58 GPIOA multiplexing selection register 0

bit access		Instructions	reset value
[15: 0]	RW	GPIO multiplexing function selection bit low address bit, together with GPIO_AF_S1 determine the multiplexing function How to configure see GPIO_AF_SEL register description	16'h0

Table 59 GPIOB multiplexing selection register 0

bit access		Instructions	reset value
[31: 0]	RW	GPIO multiplexing function selection bit low address bit, together with GPIO_AF_S1 determine the multiplexing function How to configure see GPIO_AF_SEL register description	32'h0

9.4.9 GPIO interrupt trigger configuration register

Table 60 GPIOA interrupt trigger mode configuration register

bit access		Instructions	reset value
[15: 0]	RW	GPIO interrupt trigger mode, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] interrupt is edge triggered [x] = 1, GPIO[x] interrupt is level triggered	16'h0

Table 61 GPIOB interrupt trigger mode configuration register

bit access		Instructions	reset value
[31: 0]	RW	GPIO interrupt trigger mode, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] interrupt is edge triggered	32'h0

		[x] = 1, GPIO[x] interrupt is level triggered	

9.4.10 GPIO interrupt edge trigger mode configuration register

Table 62 GPIOA interrupt edge trigger mode configuration register

bit access		Instructions	reset value
[15: 0]	RW	GPIO interrupt edge trigger mode, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] edge trigger interrupt mode is determined by GPIO_IEV [x] = 1, both edges of GPIO[x] trigger interrupt	16'h0

Table 63 GPIOB interrupt edge trigger mode configuration register

bit access		Instructions	reset value
[31: 0]	RW	GPIO interrupt edge trigger mode, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] edge trigger interrupt mode is determined by GPIO_IEV [x] = 1, both edges of GPIO[x] trigger interrupt	32'h0

9.4.11 GPIO interrupt upper and lower edge trigger configuration register

Table 64 GPIOA interrupt upper and lower edge trigger configuration register

bit access		Instructions	reset value
[15: 0]	RW	GPIO interrupt upper and lower edge trigger or high and low level trigger selection, each BIT corresponds to the corresponding GPIO line, 1'bxÿ [x] = 0, GPIO[x] interrupt is triggered by low level or falling edge [x] = 1, GPIO[x] interrupt is triggered by high level or rising edge	16'h0

Table 65 GPIOB interrupt upper and lower edge trigger configuration register

bit access		Instructions	reset value
[31: 0]	RW	GPIO interrupt upper and lower edge trigger or high and low level trigger selection, each BIT corresponds to the corresponding GPIO line, 1'bxÿ [x] = 0, GPIO[x] interrupt is triggered by low level or falling edge [x] = 1, GPIO[x] interrupt is triggered by high level or rising edge	32'h0

9.4.12 GPIO Interrupt Enable Configuration Register

Table 66 GPIOA interrupt enable configuration register

bit access		Instructions	reset value
[15: 0]	RW	GPIO interrupt enable control, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] interrupt disabled [x] = 1, GPIO[x] interrupt enable	16'h0

Table 67 GPIOB interrupt enable configuration register

bit access		Instructions	reset value
[31: 0]	RW	GPIO interrupt enable control, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] interrupt disabled [x] = 1, GPIO[x] interrupt enable	32'h0

9.4.13 GPIO Raw Interrupt Status Register

Table 68 GPIOA Raw Interrupt Status Register

bit access		Instructions	reset value
[15: 0]	RW	GPIO raw interrupt status (before MASK), each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] no interrupt generated [x] = 1, GPIO[x] has an interrupt	16'h0

Table 69 GPIOB Raw Interrupt Status Register

bit access		Instructions	reset value
[31: 0]	RW	GPIO raw interrupt status (before MASK), each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] no interrupt generated [x] = 1, GPIO[x] has an interrupt	32'h0

9.4.14 GPIO Masked Interrupt Status Register

Table 70 GPIOA Masked Interrupt Status Register

bit access		Instructions	reset value
[15: 0]	RW	GPIO masked interrupt status (after MASK), each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] no interrupt generated (after MASK) [x] = 1, GPIO[x] interrupt generated (after MASK)	16'h0

Table 71 GPIOB Masked Interrupt Status Register

bit access		Instructions	reset value
[31: 0]	RW	GPIO masked interrupt status (after MASK), each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] no interrupt generated (after MASK) [x] = 1, GPIO[x] interrupt generated (after MASK)	32'h0

9.4.15 GPIO Interrupt Clear Control Register

Table 72 GPIOA Interrupt Clear Control Register

bit access		Instructions	reset value
[15: 0]	RW	GPIO interrupt clear control, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, no action [x] = 1, clear GPIO[x] interrupt status	16'h0

Table 73 GPIOB Interrupt Clear Control Register

bit access		Instructions	reset value
[31: 0]	RW	GPIO interrupt clear control, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, no action [x] = 1, clear GPIO[x] interrupt status	32'h0

10 High Speed SPI Device Controller

10.1 Function overview

Compatible with the general SPI physical layer protocol, by agreeing on the data format for interaction with the host, the host can perform high-speed access to the device, the highest support

The working frequency is 50MHZ.

10.2 Main Features

- ÿ Compatible with general SPI protocol
- ÿ Selectable level interrupt signal
- ÿ Support up to 50Mbps rate
- ÿ Simple frame format, full hardware analysis and DMA

10.3 Functional description

10.3.1 Introduction to SPI protocol

SPI works in master-slave mode, usually there is one master device and one or more slave devices, at least 4 wires are required, in fact 3 wires are also possible (single

when transferring). They are SDI (data input), SDO (data output), SCLK (clock), CS (chip select).

(1) SDI – Serial Data In, serial data input

(2) SDO – Serial Data Out, serial data output

(3) SCLK – Serial Clock, clock signal, generated by the master device

(4) CS – Chip Select, slave device enable signal, controlled by the master device.

Among them, CS is the control signal whether the slave chip is selected by the master chip, that is to say, only when the chip select signal is a predetermined enable signal (high

Potential or low potential), the master chip is only valid for the operation of the slave chip. This makes it possible to connect multiple SPI devices on the same bus.

In addition to the above 4 signal lines, HSPI also adds an additional INT line, which generates a drop when the slave device has data to upload

The interrupt along the edge realizes the active reporting of data.

SPI communication is completed through data exchange, the data is transmitted bit by bit, the clock pulse is provided by SCLK, and SDI and SDO are based on

This pulse completes the data transfer. The data output is through the SDO line, and the data changes on the rising or falling edge of the clock, and changes on the following falling edge or

rising edge is read. Complete one-bit data transmission, input also uses the same principle. Therefore, at least 8 clock signal changes (rising edge

and the falling edge once), to complete the transmission of 8-bit data.

The SCLK signal line is controlled by the master device, and the slave device cannot control the signal line. In an SPI-based device, there is at least one master device.

10.3.2 SPI working process

The HSPI inside the chip works with the wrapper controller, and the wrapper controller integrates DMA, which is realized by DMA

Data exchange between HSPI internal FIFO and on-chip buffer. This operation is implemented by hardware, and the software does not need to care about data sending and receiving

process, you only need to configure the sending and receiving data linked list, and operate the corresponding registers of the wrapper controller.

For a detailed introduction to the wrapper controller, please refer to the relevant chapters.

10.4 Register Description

10.4.1 Register list for internal operation of HSPI chip

Table 74 HSPI internal access registers

offset address	name	abbreviation	access	describe	reset value

0X0000	HSPI FIFO clear register CLEAR_FIFO		RW	Clears the contents of the Tx and Rx FIFOs while Circuitry that resets the system clock domain synchronously	0X0000_0000
0X0004	HSPI Configuration Register	SPI_CFG	RW	configures the transmission mode of SPI and the big and small end configuration place	0X0000_0000
0X0008	HSPI mode configuration register MODE_CFG		RW	configures the ahb master to access the bus burst length	0X0000_0000
0X000C	HSPI Interrupt Configuration Register	SPI_INT_CPU_ MASK	RW	configuration interrupt is enabled	0X0000_0003
0X0010	HSPI Interrupt Status Register	SPI_INT_CPU_ TTS	RW	Get and clear interrupt status	0X0000_0000
0X0018	HSPI data upload length register RX_DAT_LEN		RW	configures the length of data that can be uploaded	0X0000_0000

10.4.1.1 HSPI FIFO flush register

Table 75 HSPI FIFO flush register

bit	access	Instructions	reset value
[31: 1]	RO	reserve	
[0]	RW	Clear FIFOs, clears the contents of Tx and Rx FIFO, and simultaneously resets the circuit of the system clock domain (this except the registers in the list) 0: Do not clear FIFO 1: clear valid Set by software, cleared by hardware Note: If you want to reset the whole circuit, you need to use the asynchronous reset pin of this module: rst_n	1'b0

10.4.1.2 HSPI Configuration Register

Table 76 HSPI configuration register

bit access		Instructions	reset value
[31: 4] RO		reserve	
[3]	RW	Bigendian, spi interface supports big endian selection of data. 0: Support small segment data transmission 1: Support big-endian data transmission	1'b0
[2]	RW	spi_tx_always_drive 0: The spi output is only valid when the chip selection is valid, and it is high impedance at other times 1: spi output is always valid	1'b0
[1]	RW	SPI CPHA 0: Transmission mode A 1: Transmission mode B	1'b0
[0]	RW	SPI CPOL, polarity of SCK at IDLE 0: 0 when SCK is IDLE 1: 1 when SCK is IDLE	1'b0

10.4.1.3 HSPI Mode Configuration Register

Table 77 HSPI Mode Configuration Register

bit access		Instructions	reset value
[31: 1] RO		reserve	

[0]	RW	Burst len, the burst length when the ahb master accesses the bus 0: burst len is 1 word 1: burst len is 4 words It is recommended to set it as a burst transmission of 4 words, so that when the frequency of the spi interface is high, it can ensure continuous flow	1'b0

10.4.1.4 HSPI Interrupt Configuration Register

Table 78 HSPI Interrupt Configuration Register

bit access		Instructions	reset value
[31: 2] RO		reserve	
[1]	RW	IntEnRxOverrun, RxOverrun interrupt enable 0: Rx FIFO overflow interrupt enable 1: Rx FIFO overflow interrupt disabled	1'b1
[0]	RW	IntEnTxUnderrun, TxUnderrun interrupt enable 0: Tx FIFO underflow interrupt enable 1: Tx FIFO underflow interrupt disabled	1'b1

10.4.1.5 HSPI Interrupt Status Register

Table 79 HSPI Interrupt Status Register

bit access		Instructions	reset value
[31: 2] RO		reserve	
[1]	RW	RxOverrun 0ÿRx FIFO overflow	1'b0

		1ÿRx FIFO overflow Write 1 to clear	
[0]	RW	TxUnderrun 0ÿTx FIFO underflow 1ÿTx FIFO underflow Write 1 to clear	1'b0

10.4.1.6 HSPI data upload length register

Table 80 HSPI data upload length register

bit access		Instructions	reset value
[31:16] RO		reserve	
[15: 0] RW		Rx_dat_len Indicates the length of data that can be uploaded, in bytes The uploaded length is an integer multiple of words, if the uploaded length is less than a whole word, it will be rounded up.	16'h0

10.4.2 Host-side access to HSPI controller register list

The host side accesses the SPI interface registers through a fixed SPI command format. The command length is fixed at one byte, and the data length is fixed at two byte.

Table 81 HSPI Interface Configuration Register (Master Access)

offset address	name	abbreviation	access	describe	reset value

0X02 Get data length register		RX_DAT_LEN	RO	When uploading data, the spi master is used to obtain the slave The length of the data read by the device	0X0000
0X03 Send data flag register		TX_BUFF_AVAIL	RO	When the master sends data to the slave, it is used to judge whether it is possible Download data or command	0X0000
0X04 Reserved		RSV	RO		
0X05 interrupt configuration register		SPI_INT_HOST_MASK RW Whether to mask interrupt			0X0000
0X06 interrupt status register		SPI_INT_HOST_STTS	RO	Interrupt status register, the spi master queries this bit Confirm whether there is data uploaded	0X0000
0X07 Reserved		RSV	RO		
0X00 data port 0		DAT_PORT0	RW	The Spi master communicates with the slave device through this register port Send data, send the previous data frame using this port	
0X10 data port 1		DAT_PORT1	RW	The Spi master communicates with the slave device through this register port Send data, send the last data frame to use the port	
0X01 command port 0		DN_CMD_PORT0	WHERE	The Spi master sends the slave device through this register Command data, send the previous command data to use the port	
0X11 command port 1		DN_CMD_PORT1	WHERE	The Spi master sends the slave device through this register Command data, send the last frame of command data to make use this port	

10.4.2.1 HSPI Get Data Length Register

Table 82 HSPI get data length register

bit	access	Instructions	reset value
[15: 0] RO		The spi host read-only register is mainly used to know how much data is read from the device when uploading data However, in this module, the upload length is an integer multiple of words. If the upload length value is not a whole word, the host reads Round up when counting, that is, read more redundant bytes	16'h0

10.4.2.2 HSPI sent data flag register

Table 83 HSPI sent data flag register

bit	access	Instructions	reset value
[15: 2] RO		reserve	
[1]	RO	tx_cmdbuff_avail Flag to send cmd buff is available, if available, the host can send cmd. 0: send buff is not available 1: send buff available	1'b0
[0]	RO	tx_buff_avail Flag sending buff is available, if available, the host can send data. 0: send buff is not available 1: send buff available	1'b0

10.4.2.3 HSPI Interrupt Configuration Register

Table 84 HSPI Interrupt Configuration Register

bit	access	Instructions	reset value
[15: 1] RO		reserve	
[0]	RO	IntMaskup_dat_cmd_rdy interrupt mask 0: Interrupts are not masked, interrupts can be generated 1: Interrupts are masked Note: It is recommended to use the host's own internal interrupt mask, which can improve efficiency.	1'b0

10.4.2.4 HSPI Interrupt Status Register

Table 85 HSPI Interrupt Status Register

bit	access	Instructions	reset value
[15: 1] RO		reserve	
[0]	RO	up_dat_cmd_rdy Status register for generating interrupts to the SPI master 0: Data or command is not ready 1: Data or command is ready legible	1'b0

10.4.2.5 HSPI data port 0

Table 86 HSPI data port 0

bit	access	Instructions	reset value
	RW	The SPI host performs data transmission through the register port and the device, and writes the data to the register to send the data. Data, read from this register, you can upload data. If the frame being transmitted requires multiple transmissions to complete If successful, the last transmission uses the register port DAT_PORT1, and the others use DAT_PORT0.	

10.4.2.6 HSPI Data Port 1

Table 87 HSPI data port 1

bit	access	Instructions	reset value
	RW	The SPI host performs data transmission through the register port and the device, and writes the data to the register to send the data. Data, read from this register, you can upload data. If the frame being transmitted requires multiple transmissions to complete If successful, the last transmission uses the register port DAT_PORT1, and the others use DAT_PORT0.	

10.4.2.7 HSPI Command Port 0

Table 88 HSPI command port 0

bit	access	Instructions	reset value
	RW	The SPI host interacts with the device through the register port, and writes to the register to issue commands. make. If the command being transmitted requires multiple transfers to complete, the last transfer uses the register Port DN_CMD_PORT1, others use DN_CMD_PORT0. Note: This window is only used to issue commands for driver and firmware negotiation.	

10.4.2.8 HSPI Command Port 1

Table 89 HSPI command port 1

bit	access	Instructions	reset value
	RW	The SPI host interacts with the device through the register port, and writes to the register to issue commands. make. If the command being transmitted requires multiple transfers to complete, the last transfer uses the register Port DN_CMD_PORT1, others use DN_CMD_PORT0. Note: This window is only used to issue commands for driver and firmware negotiation.	

10.4.3 High Speed SPI Device Controller Interface Timing

Mainly describe the SPI read and write timing, and how the main SPI interacts with HSPI for data.

10.4.3.1 Data format

The data format is divided into two parts: the command field and the data field, as shown in the figure below. The fixed length of the command field is 8bit, and the length of the data field depends on the access object
Different, different lengths, see below for details.

The highest bit of the command field is the read and write flag bit, and the remaining 7 bits are the address.

- ÿ 0 means read data from the next 7bit address
- ÿ 1 means write data to the next 7bit address

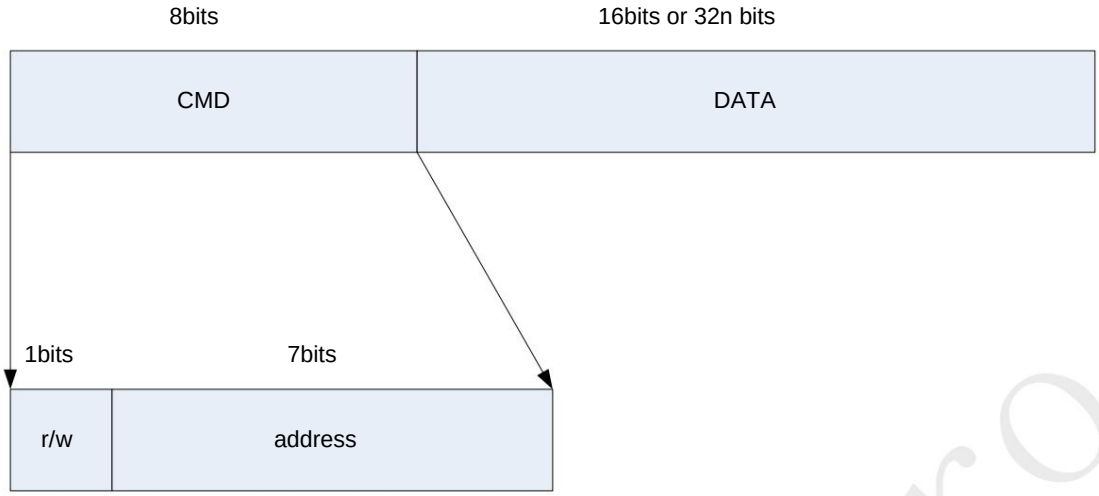


Figure 5 SPI send and receive data format of upper computer

The data field of this module only supports two lengths, the upper computer SPI access interface configuration register (Table 2), the length of the data field is 16bit;

Transmit data through ports (data port 0, data port 1, command port 0 and command port 1), and the length of the data field is an integer of 32 bits

times;

The following figure is the timing diagram of the read and write interface configuration register. The slave device is configured in little endian mode by default.

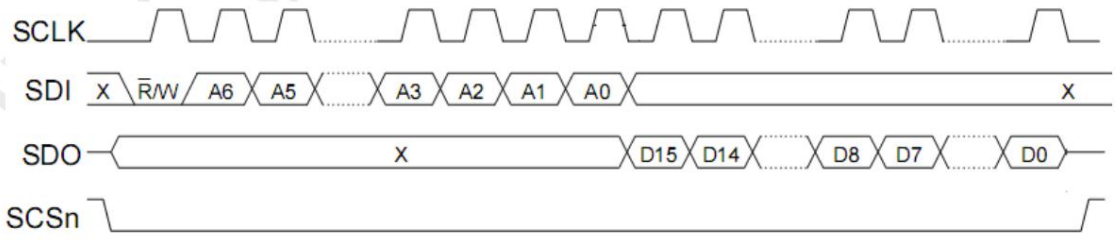


Figure 6 HSPI register read operation (big endian mode)

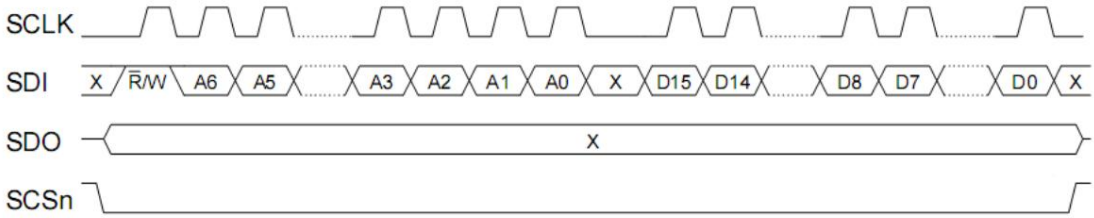


Figure 7 HSPI register write operation (big endian mode)

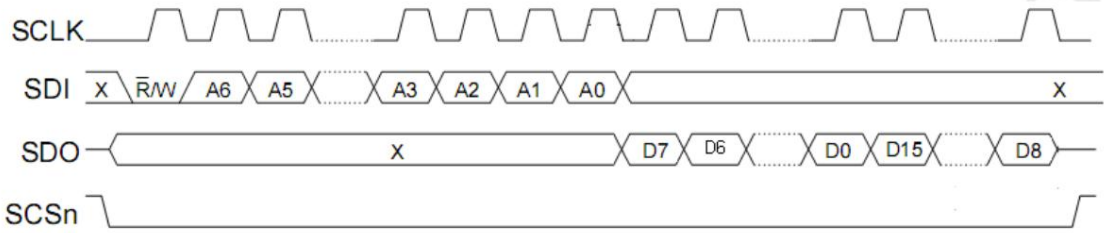


Figure 8 Register read operation (little endian mode)

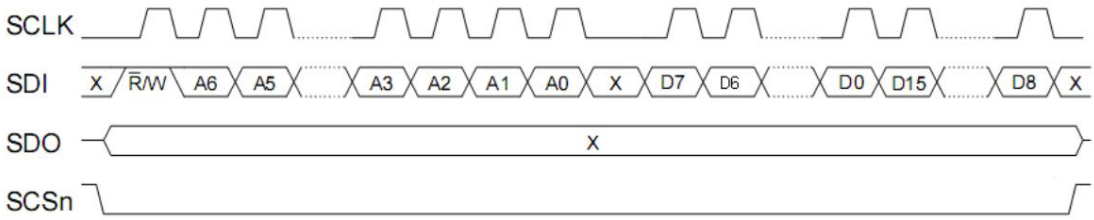


Figure 9 Register write operation (little endian mode)

The following figure is the sequence diagram of reading and writing data, the length of the data field is an integer multiple of 32bit, and the figure only transmits the length of one word.

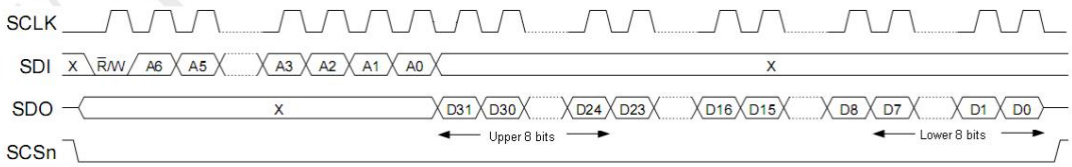


Figure 10 Port read operation (big endian mode)

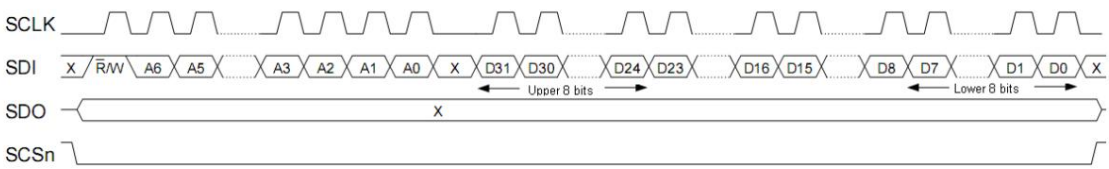


Figure 11 Port write operation (big endian mode)

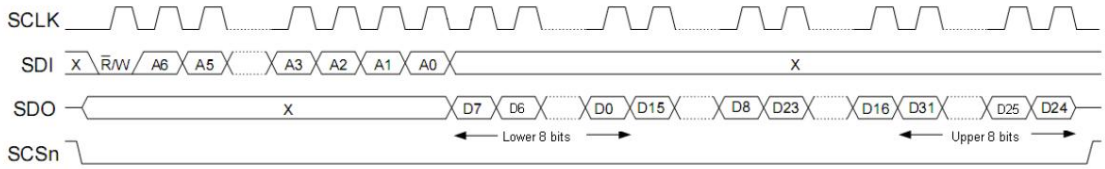


Figure 12 Port read operation (little endian mode)

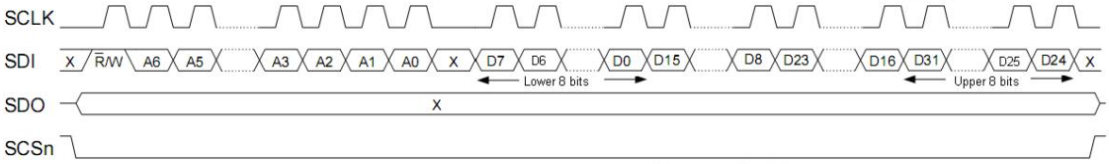


Figure 13 Port write operation (little endian mode)

Note: There is no waiting time between the command and the data, that is, after the command field is transmitted, the data transmission can be followed immediately, and no idle clock is required

or free time. A time delay is fine, but no idle clocks can appear.

10.4.3.2 Timing

This module supports half-duplex, and the supported timing is divided into 4 types according to the clock phase and sampling point. The following timings just give the clock

phase and sampling relationship. It should be noted that the chip supports by default (CPOL=0, CPHA=0).

Note: There is no waiting time between the command and the data, that is, after the command field is transmitted, the data transmission can be followed immediately, and no idle clock is required

or free time. A time delay is fine, but no idle clocks can appear.

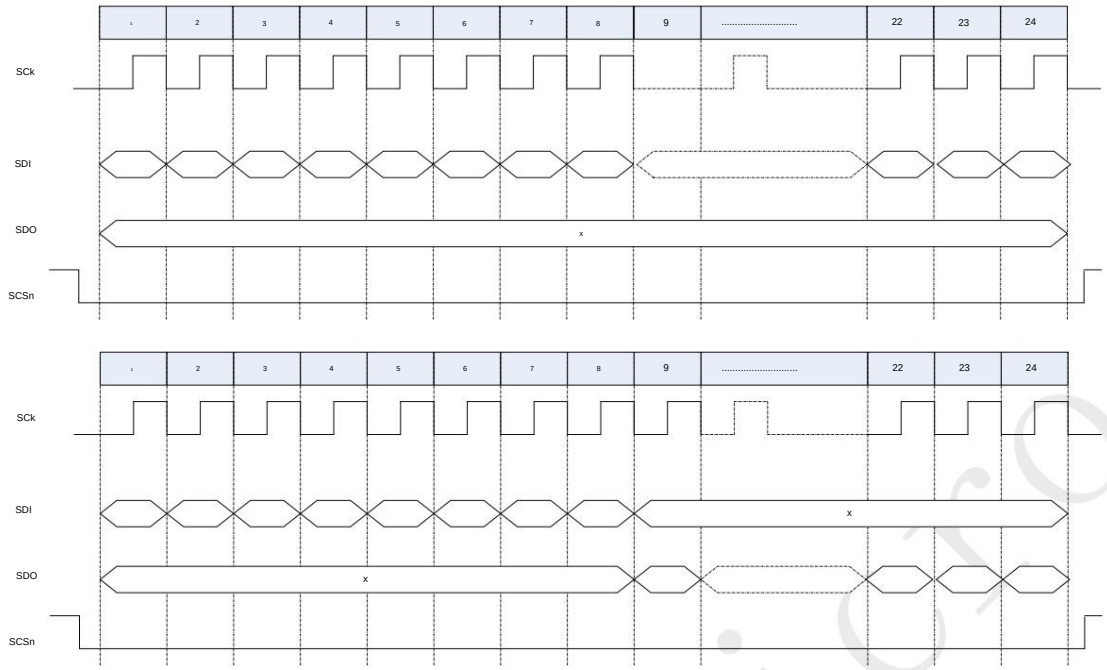


Figure 14 CPOL=0, CPHA=0

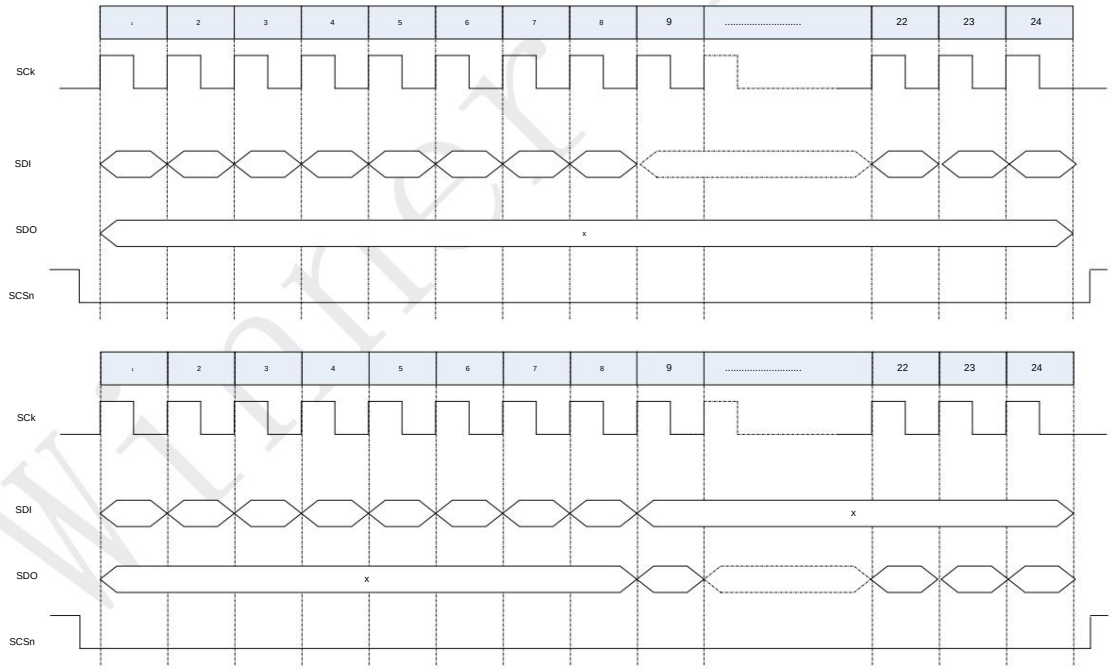


Figure 15 CPOL=0, CPHA=1

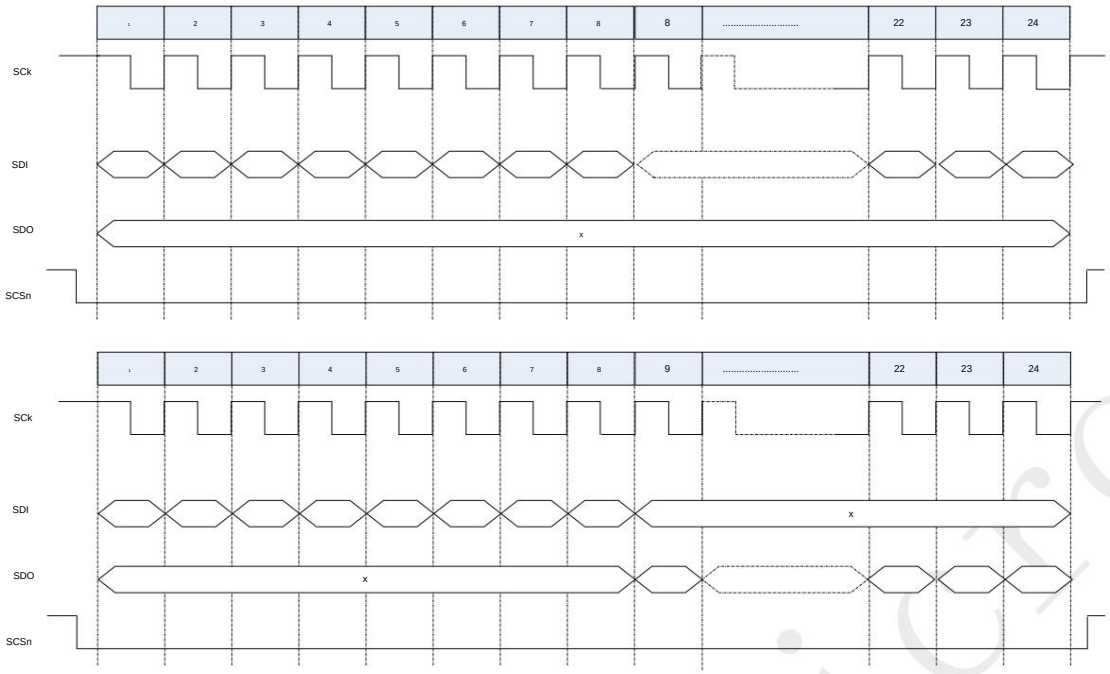


Figure 16 CPOL=1, CPHA=0

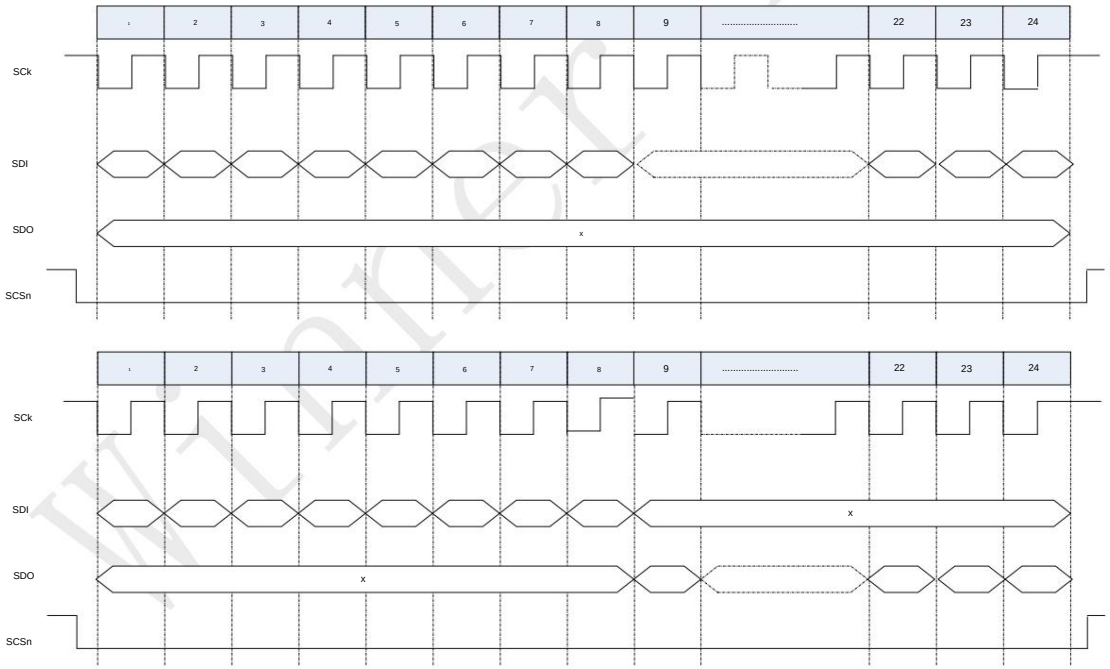


Figure 17 CPOL=1, CPHA=1

10.4.3.3 Interrupts

The interrupt signal is sent from the slave device to the master device, triggered by the SPI_INT pin, and is active at low level.

spi_int mainly notifies the spi host that data or commands need to be uploaded, and the interface registers that the spi host cares about when processing interrupts are:

ÿ SPI_INT_HOST_MASK

ÿ SPI_INT_HOST_STTS

ÿ RX_DAT_LEN

Note: Each uploaded frame corresponds to an interrupt. Only after the transmission of the current frame that needs to be uploaded is completed, if there are still frames to be uploaded, at this time, the

Generate a new interrupt. The figure below is one way to handle interrupts.

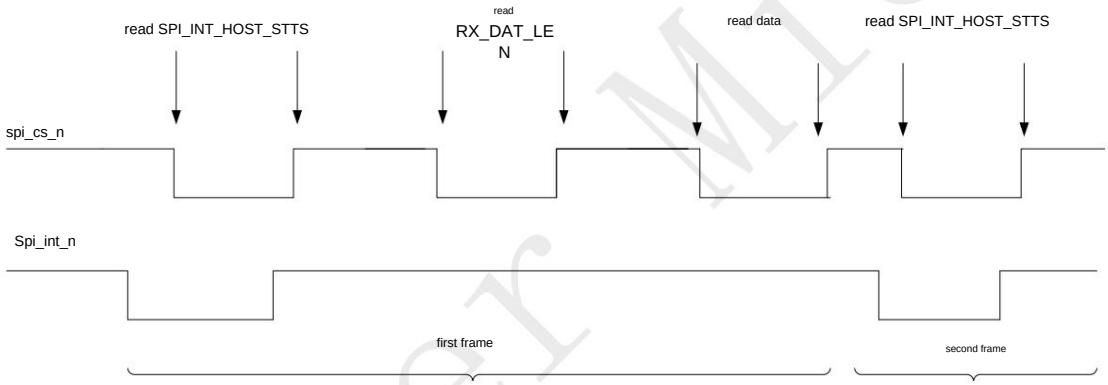


Figure 18 Main SPI processing interrupt flow

10.4.3.4 Main SPI send and receive data workflow

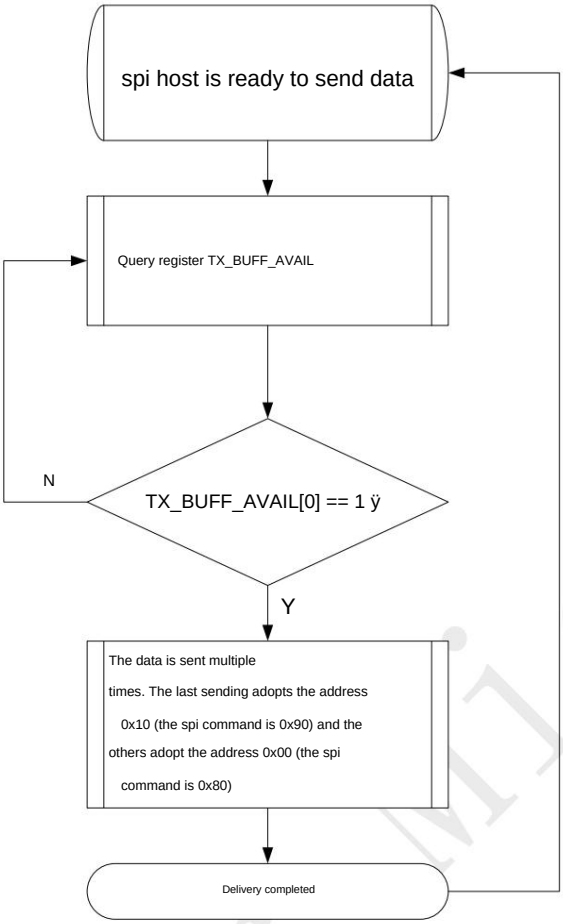


Figure 19 Flow chart of downlink data

Note: The length of the sent data must be in words, if it is not a whole word, fill it with 0.

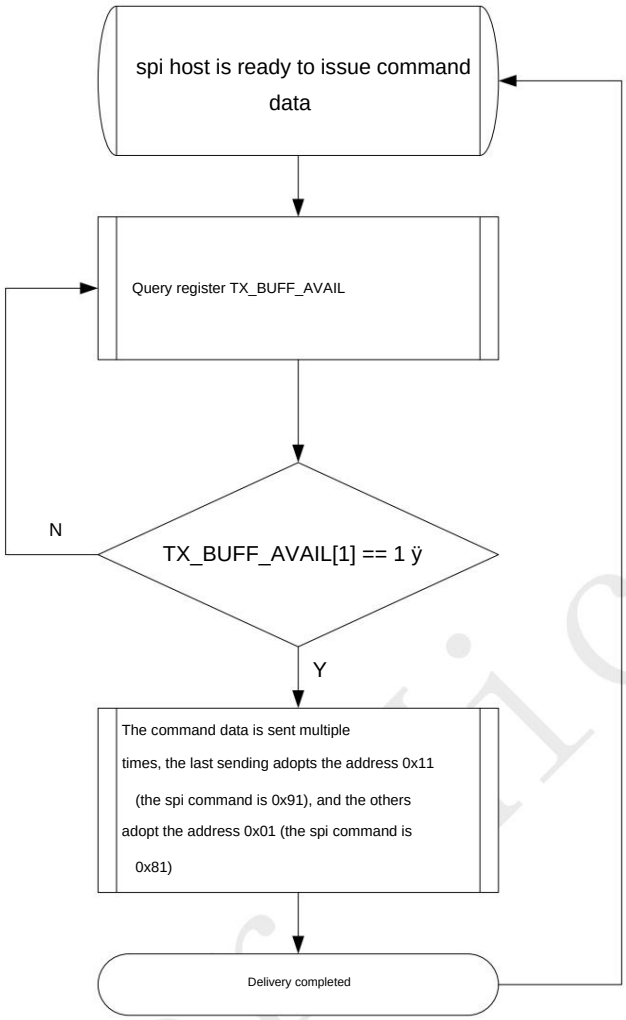


Figure 20 Downlink command flow chart

Note: The length of the issued command must be in words, if it is not a whole word, fill it with 0.

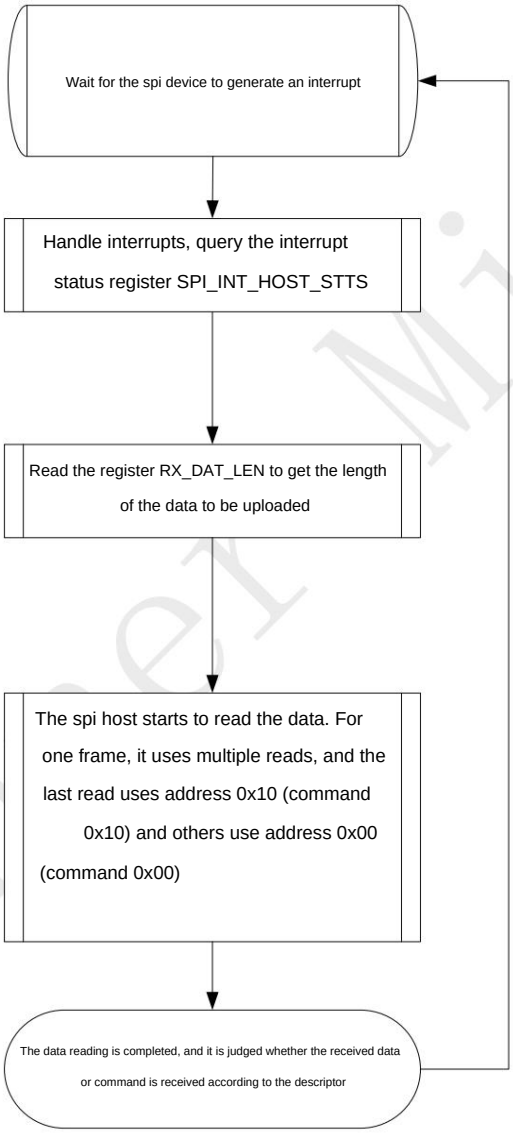


Figure 21 Uplink data (command) flow chart

The flow of uplink data is the same as that of uplink commands.

It should be noted here that the length of the upstream data must be in words, if the effective length is not a whole word, the excess data at the end can be thrown away

Lose.

It should be noted that there are two channels between the master and the slave, data and command, to exchange data, and users can choose a channel to use according to their needs.

Use or both. The maximum data length of one interaction of the command channel is 256 bytes, and the maximum data length of one interaction of the data channel is

1500 bytes. The data length limit is controlled by the slave device. If the length exceeds the limit, the data structure of the slave device will be destroyed.

11 SDIO device controller

11.1 Function overview

W800 integrates the SDIO device interface, as a slave device, to complete the data interaction with the host. Internally integrated 1024byte asynchronous FIFO, to complete the data interaction between the host and the chip.

11.2 Main Features

- ÿ Compatible with SDIO Card Specification 2.0
- ÿ Support host rate 0~50MHz
- ÿ Support up to 1024 bytes Block
- ÿ Support 1-bit SD and 4-bit SD modes

11.3 Functional description

11.3.1 SDIO bus

The SDIO bus is similar to the USB bus, and the SDIO bus also has two ends, one end is the host end, and the other end is the device end, using HOST The design of DEVICE is to simplify the design of DEVICE, and all communication is started by issuing commands from the HOST. exist As long as the DEVICE side can parse the HOST command, it can communicate with the HOST, and the SDIO HOST can connect to multiple DEVICEÿ

In the SDIO bus definition, the DAT1 signal line is multiplexed as an interrupt line. In the 1BIT mode of SDIO, DAT0 is used to transmit data, DAT1 Used as a break line. In the 4BIT mode of SDIO, DAT0 -DAT3 are used to transmit data, and DAT1 is multiplexed as an interrupt line.

11.3.2 SDIO commands

On the SDIO bus, the HOST end initiates the request, and then the DEVICE end responds to the request, where the request and response will contain data information:

ÿ Command: The command used to start the transmission is sent from the HOST to the DEVICE, and the command is sent through the CMD message

transmitted by the number line;

ÿ Response: The response is returned by DEVICE, as the response of Command. It is also transmitted through the CMD line;

ÿ Data: Data is transmitted bidirectionally. It can be set to 1-wire mode or 4-wire mode. Data is passed through DAT0-

DAT3 signal line transmission.

Each operation of SDIO is initiated by HOST on the CMD line. For some CMDs, DEVICE needs to return a Response.

Others don't.

For the read command, first HOST will send a command to DEVICE, then DEVICE will return a handshake signal, at this time, when HOST

After receiving the response handshake signal, the data will be placed on the 4-bit data line, and the CRC check code will be followed while transmitting the data. When the whole

After the read transmission is completed, HOST will send another command to notify DEVICE that the operation is complete, and DEVICE will return a response at the same time.

For writing commands, first HOST will send commands to DEVICE, then DEVICE will return a handshake signal, at this time, when HOST

After receiving the response handshake signal, the data will be placed on the 4-bit data line, and the CRC check code will be followed while transmitting the data. When the whole

After the first write transmission is completed, HOST will send another command to notify DEVICE that the operation is complete, and DEVICE will return a response at the same time.

11.3.3 SDIO Internal Storage

The SDIO device has a fixed memory map inside, including a general information area (CIA) and a special function area (function unique area).

The registers in the CIA include I/O port functions, interrupt generation, and port work information, which can be defined for the CIA by reading and writing function 0

Registers for related operations. CIA contains CCCR, FBR and CIS three aspects of information. Among them, CCCR defines the common

Common control register, the host can check the SDIO card and operate the port by operating CCCR, the address of CCCR is 0X00-

0XFF. FBR defines the supported port function 1 to port function 7 operations, including the requirements and functions of each port, power control, etc.,

The address of FBR is 0Xn00-0Xnff (where n is the function port number). CIS defines some information structures of the card, the address is 0X1000-

0X17FFF, CIS has a public CIS and each function port's own CIS, where the initial address of the public CIS is in the CIS Pointer of CCCR

In the domain, the CIS of each port function is in the CIS Pointer domain of each functional port FBR.

The storage map of the CIA is shown in the figure below.

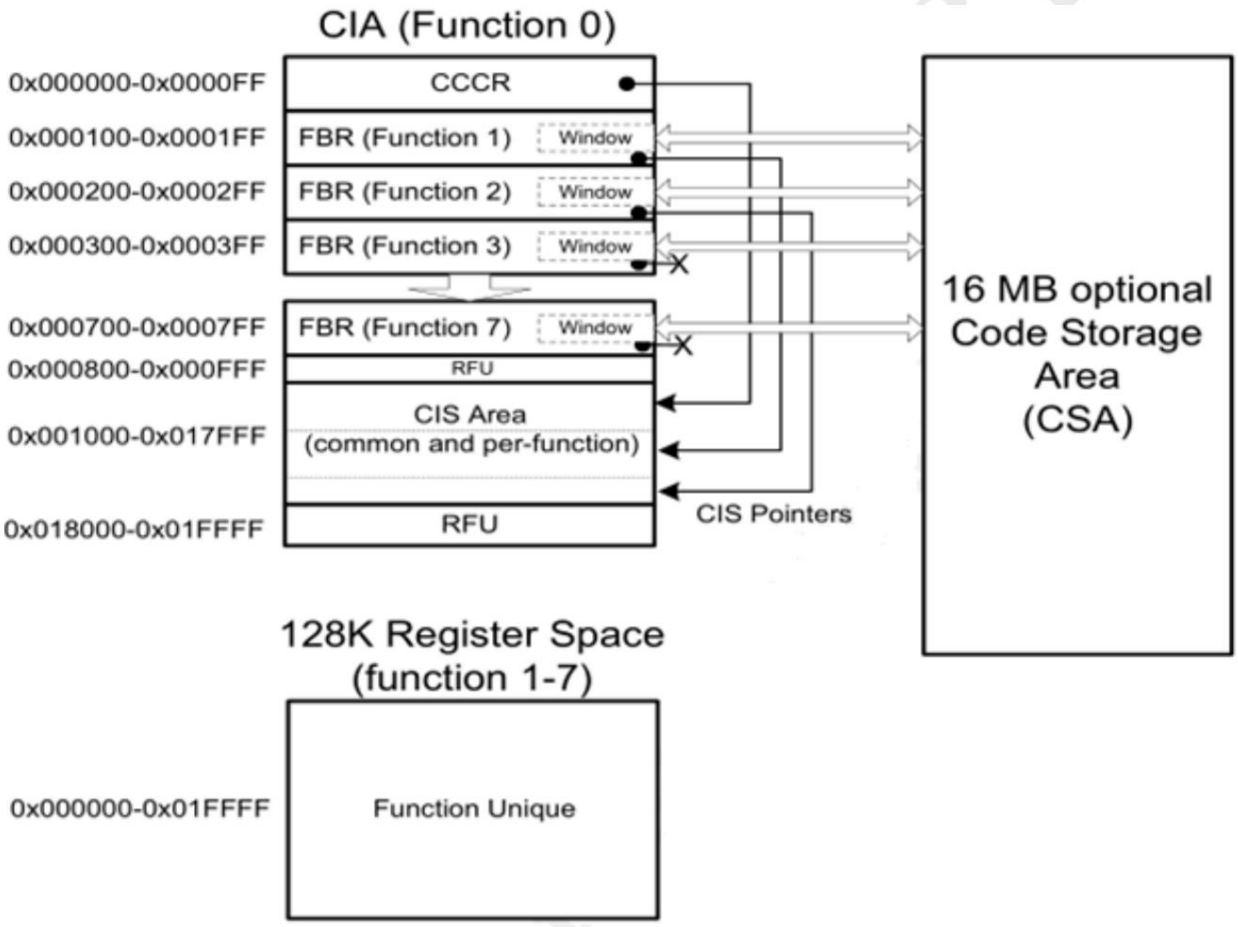


Figure 22 SDIO internal memory mapping

The description of each register in CIA refers to the following. For an in-depth look at the CIA, see the SDIO Protocol Specification.

11.4 Register Description

11.4.1 Register List

11.4.2 SDIO Fn0 Register

The Fn0 register is a register specified by the SDIO protocol, and its address range is: 0x00000~0x1FFFF, a total of 128K. starting address is

0x00000ÿ

The Fn0 register is accessed by the SDIO host through the CMD52 command, the offset address is the access address, and the function number is 0.

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00	CCCR/SDIO Revision	SDIO bit 3	SDIO bit 2	SDIO bit 1	SDIO bit 0	CCCR bit 3	CCCR bit 2	CCCR bit 1	CCCR bit 0
0x01	SD Specification Revision	RFU	RFU	RFU	RFU	SD bit 3	SD bit 2	SD bit 1	SD bit 0
0x02	I/O Enable	IOE7	IOE6	IOE5	IOE4	IOE3	IOE2	IOE1	RFU
0x03	I/O Ready	IOR7	IOR6	IOR5	IOR4	IOR3	IOR2	IOR1	RFU
0x04	Int Enable	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IENM
0x05	Int Pending	INT7	INT6	INT5	INT4	INT3	INT2	INT1	RFU
0x06	I/O Abort	RFU	RFU	RFU	RFU	RES	AS2	AS1	AS0
0x07	Bus Interface Control	CD Disable	SCSI	ECSI	RFU	RFU	RFU	Bus Width 1	Bus Width 0
0x08	Card Capability	4BLS	LSC	E4MI	S4MI	SBS	SRW	SMB	SDC
0x09-0x0B	Common CIS Pointer	Pointer to card's *common* Card Information Structure (CIS)							
0x0C	Bus Suspend	RFU	RFU	RFU	RFU	RFU	RFU	BR	BS
0x 0D	Function Select	DF	RFU	RFU	RFU	FS3	FS2	FS1	FS0
0x 0E	Exec Flags	EX7	EX6	EX5	EX4	EX3	EX2	EX1	EXM
0x 0F	Ready Flags	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RFM
0x10-0x11	FN0 Block Size	I/O block size for Function 0							
0x 12	Power Control	Reserved for Future Use (RFU)						EMPC	SMPC
0x 13	High-Speed	RFU	RFU	RFU	RFU	RFU	RFU	EHS	SHS
0x14-0xEF	RFU	Reserved for Future Use (RFU)							
0xF0-0xFF	Reserved for Vendors	Area Reserved for Vendor Unique Registers							

Figure 23 CCCR register storage structure

Address	7	6	5	4	3	2	1	0
0x100	Function 1 CSA enable	Function 1 supports CSA	RFU	RFU	Function 1 Standard SDIO Function interface code			
0x101	Function 1 Extended standard SDIO Function interface code							
0x102	RFU	RFU	RFU	RFU	RFU	RFU	EPS	SPS
0x103-0x108	Reserved for Future Use (RFU)							
0x109-0x10B	Pointer to Function 1 Card Information Structure (CIS)							
0x10C-0x10E	Pointer to Function 1 Code Storage Area (CSA)							
0x10F	Data access window to Function 1 Code Storage Area (CSA)							
0x110-0x111	I/O block size for Function 1							
0x112-0x1FF	Reserved for Future Use							
0x200-0x7FF	Function 2 to 7 Function Basic Information Registers (FBR)							
0x800-0xFFF	Reserved for Future Use							

Figure 24 FBR1 register structure

Address	7	6	5	4	3	2	1	0
0x0001000 - 0x017FFF	Card Common Card Information Structure (CIS) area for card common and all functions							
0x018000- 0x01FFFF	Reserved for Future Use							

Figure 25 CIS storage space structure

11.4.2.1 SDIO CCCR Register and FBR1 Register List

Table 90 SDIO CCCR register and FBR1 register list

offset address	abbreviated name access			describe	reset value
0X00	CCCR/SDIO Revision	SDIOx RO		[3:0], indicates the supported CCCR/FBR format 4'h0ÿ CCCR/FBR Version 1.00 4'h1ÿCCCR/FBR Version 1.10 4'h2ÿCCCR/FBR Version 1.20 4'h3-4'hFÿRsv Represented by CIA Register[3:0]	4'h2

		CCCRx RO		[7:4], indicates the supported SDIO protocol version 4'h0ÿ SDIO Version 1.00 4'h1ÿ SDIO Version 1.10 4'h2ÿ SDIO Version 1.20 (unreleased) 4'h3ÿ SDIO Version 2.00 4'h4-4'hFÿ Rsv Represented by CIA Register[7:4]	4'h3
0X01	SD pecificatio n Revision	SDx	RO [3:0],	indicates the supported SD protocol version 4'h0ÿSD Physical Version 1.01 (March 2000) 4'h1ÿSD Physical Version 1.10 (October 2004) 4'h2ÿSD Physical Version 2.00 (May 2006) 4'h3-4'hFÿ Rsv Represented by CIA Register[11:8]	4'h2
			RO	RFU	4'h0
0X02	I/O Enable IOEx		RO	RFU	1'b0
			RW [7:1],	Function enable, bit1-bit7 correspond to 7 functions respectively, corresponding to If the SD host sets the corresponding bit to 1, then the corresponding function is enabled, otherwise the corresponding function doesn't work. Note: CIS0, CIS1 and CSA are placed in Fn1, even if Fn1 is not enabled, The SD host can also read and write these three areas (CIS0, CIS1 cannot Write).	7'b0
0X03	I/O Ready IORx		RO	RFU	1'b0

			RO	[7:1], IOR has a total of 7 bits, corresponding to the status of 7 functions, if the corresponding bit If it is 1, it means that the function can work. In this design, HC8051 configures the function of program register The ready bit is 1, so this register bit1=1, so that the flag Fn1 can be normal Work. Note: The read and write operations to CIS0, CIS1, CSA are independent of IOR1, that is, even if IOR1=0, you can also access the contents of these three storage spaces.	7'b0
0X04 Int Enable	IENM	RW	[0], interrupt enable signal	0: Interrupts from the card cannot be sent to the SD host 1: Any function interrupt can be sent to the host	1'b0
		IENx	RW	[7:1], interrupt enable for functionx. IEN1=0, the interrupt from Fn1 will not be sent to the host. IEN1=1, then allow the Fn1 interrupt to be sent to the host	7'b0
0X05 Int	Pending		RO	[0],RFU	1'b0
		YOU	RO	[7:1], the interrupt of functionx is pending. INT1=0, no Fn1 interrupt is pending INT1=1, Fn1 interrupt is pending. Note: If IEN1 and IENM are not 1, the host will not receive the pending interrupt	7'b0
0X06 I/O Abort	ASx		WO	[2:0], cancel IO read or write, thereby releasing the bus. To cancel the Fn1 operation, use CMD52 to write to 3'b1. The command is under SPI not support.	3'b0
		RES	WO	[3], soft reset signal	1'b0

				1: Reset the circuit of the SD clock domain, this bit is automatically cleared after being set, no dedicated The door is cleared. This reset signal will not affect the protocol selection of the current card (SD or SPI mode) without affecting CD Disable. Only use CMD52 to operate.	
			RO	RFU	4'b0
0X07 Bus	Interface control	Bus Width	RW [1:0], data line width 2'b00: 1bit data line mode 2'b10: 4bit data line mode Reset or power on, it will become 2'b00		2'b00
			RO	[4:2],RFU	3'b000
		ECSI RW [5], enable continuous SPI interrupt.		If SCSI is 1, this register is used to allow SDIO card in SPI mode Next, an interrupt is given at any time, and the state of the CS line does not need to be concerned at this time.	1'b0
		SCSI	RO [6], supports continuous SPI interrupt.	If it is 1, it means that the SDIO card supports SPI mode, at any time given interrupt without caring about the state of CS. When progtam reg[2] is 1, this register is set.	1'b1
		CD Disabl and	RW [7], connect or disconnect the 10-90K pull-up resistor on CD/DAT[3] (pin1).	0: Connect a pull-up resistor 1: Disconnect the pull-up resistor After power-on, this register is cleared to 0, that is, a pull-up resistor is connected. The state of this register is not It will be affected by reset command in SD protocol.	1'b0
0X08 Card		SDC	RO [0], support to execute CMD52 command during data transmission. 1'b1 not supported in SPI mode		

	Capability			the register. When progtam reg[3] is 1, this register is 1	
		SMB	RO	[1], means that the SDIO card supports the Block transfer mode required by CMD53. When program_reg[4] is 1, this register is 1	1'b1
		SRW	RO	[2], indicating that SDIO supports read wait - Read Wait Control (RWC) operation. When program_reg[5] is 1, this register is 1	1'b1
		SBS	RO	[3] , means the SDIO card supports suspend/resume. If 0, the register is not supported (0x0C-0x0F) If 1, all functions except Fn0 will be suspended according to SD host requirements or restore When program_reg[6] is 1, this register is 1	1'b1
		S4MI	RO	[4], indicating that the SDIO card supports data transfer to the host in 4bit multi-block data transmission mode Generate an interrupt. 0: Does not support interrupts between block transfers, in this case, as long as IENx=1, SDIO can still initiate an interrupt to the host in other interrupt cycles 1: Support generating interrupts between block transfers When program_reg[7] is 1, this register is 1	1'b1
		E4MI RW [5], interrupt enable.		In the 4bit multi-block mode, it is allowed to generate data to the host during two block data transmissions. Student interruption. 0: not allowed 1: allow	1'b0

				Power-on reset or reset command will clear this register to 0	
		LSC	RO	[6], 0: means the SDIO card is a full speed device 1: Indicates that the SDIO card is a low-speed device When program_reg[8] is 1, this register is 1	1'b0
		4BLS	RO	[7], 0: Indicates that SDIO is a low-speed mode device or does not support 4bit mode 1: Indicates that SDIO is a low-speed mode device and supports 4bit mode When program_reg[9] is 1, this register is 1	1'b1
0X09- 0X0B	Common CIS pointer RO			[23:0], point to the start address pointer of SDIO card common CIS (CIS0). CIS0 package Contains information about the entire card. Its access space is: 0x001000-0x017FFF. Pointers are stored in little-endian format (LSB).	24'h00 1000
0X0C Bus	Suspend	BS	RO	[0] , bus status. 0: The currently selected function does not use the data bus 1: The currently selected function (use FSx or use the function in the IO command number) is executing a command that will transmit data on the data line This register is used by the host to determine which function is currently using the data bus Wire. If the SDIO card does not support the suspend resume function, this register is 0. Any operation to access CIA cannot be suspended, this register is always 1, even if BR sends register is 1.	1'b0

				In SPI mode, it is read-only and is 0.	
		BR	RW	Bus release request/status. This register is used to request the selected function (with FSx or CMD53 general function number is selected) to release the data bus and hang up related operations. If the host sets this register to 1, the selected function will temporarily stop at the According to the data transmission on the line, and suspend the command of the current data operation. BR register save Hold at 1 until the release process is complete. Once the function is suspended, the device clears the Zero BS, BR to inform the host. The host can monitor the status of pending requests by reading the BR Execution status, if BR is 1, the pending request is still executing. The host can actively Write 0 to BR to cancel the pending request in progress. In SPI mode, it is read-only and is 0.	4'h0
			RO	[7:2],RFU	6'b0
0X0D Function	Select	FSx	RW	[3:0], used to select function[0-7] in suspend/resume operation. two ways Write FSx: Perform IO write operation to CCCR A newly initiated IO command will cause FSx to be set to the function in the command numberÿ If the function is currently suspended, write the function's number, when reading FSx, the data transfer operation of this function will be resumed do. The returned value will be the number of the currently selected function. Note: When reading FSx, if BS=0, the value of FSx is undefined.	4'b0

				4'b0000ÿ Transaction of function 0 (CIA) 4'b0001-4'b0111ÿ Transaction to functions 1-7 4'b1000ÿ Transaction of memory in combo card 4'b1001-4'b1111ÿ Not defined, reserved for future use	
			RO	[3:1],RFU	3'b000
		DF	RO [7], restore data flag. Writing the function number to FSx will restore the selected	Data transfer in function. Once data transfer resumes, the DF register will indicate Whether there is more data to transfer. 0: No more data to transfer after function is resumed. 1: There is more data to transfer after function is resumed. DF is used to control the interrupt period in 4bit mode. If it is 1, in function After recovery, there is more data to be transferred, in which case the interrupt cycle is canceled. If 0, function resumes after data transfer ends (in case of busy), In this case, after recovery, there is no data transfer, so the host can The start of an interrupt cycle is detected after function resumes.	1'b0
0X0E Exec Flags EXx			RO [7:0], execution flag bit. The host determines all functions through these bits[7-	1] The status of the executed command. These registers can inform the host that a function is The command is executed, so no new commands can be issued for this function. In SPI mode, it is read-only and is 0.	8:00 a.m
0X0F Ready	Flags	RFx	RO [7:0], read flag bits. The host can know the function[7-	1] Read and write busy status. If a function is executing a write transaction, the corresponding	8:00 a.m

				Clearing the RFx bit indicates that the function is busy and not ready to receive more data according to. If a function is performing a read operation, the corresponding RFx bit is cleared If it is zero, it indicates that the read data is invalid, and if it is 1, it indicates that the read data can be transmitted. SPI is invalid, read-only, and is 0	
0X10- 0X11	FN0 Block Size		RW [15:0],	Block size corresponding to Fn0 during transmission. maximum 2048Byte, minimum 1Byte. The storage method is small segment format (LSB)	16:00
0X12 Power	Control	SMPC RO [0],	supports host power control.	0: The total SDIO current is less than 200mA, even if all functions are valid (IOEx=1). EMPC, SPS, EPS are all 0. 1: The total current of SDIO can exceed 200mA. EMPC, SPS, EPS are valid.	1'b1
		EMPC RO [1],	host power control enable.	0: The total current of SDIO card is less than 200mA. SDIO card automatic switching function(s) Go to low current mode or not allow some functions to be enabled, and it ignores the EPS value, so that the current of the card is less than or equal to 200mA. 1: The total current of SDIO card can exceed 200mA, and SPS and EPS are valid. the host Use SPS, EPS and IOEx in FBR according to its ability to provide current Capable of higher current function.	1'b0
			RO	[7:2],RFU	
0X13 High	Speed	SHS	RO	[0], means the SDIO card supports high speed 0: does not support high speed 1: Support high speed	1'b1

		EHS	RW[1], high speed enable 0: SDIO card works at the default speed, the highest frequency is 25MHZ 1: SDIO card can work in high speed mode, the highest frequency is 50MHZ		1'b0
			RO	[7-2]ÿRFU	
0X14- 0XEF	RFU		RO	Reserved for Future Use (RFU)	8'b0
0XF0- 0XFF	Reserved for Vendors		RO	Area Reserved for Vendor Unique Registers	8'b0
0X100	I/O Device Interface Code		RO	[3:0], flags what kind of device Fn1 is. Programmable by Register CIA[15:12]. 4'h0 No SDIO standard interface supported by this function 4'h1 This function supports the SDIO Standard UART 4'h2 This function supports the SDIO Type-A for Bluetooth standard interface 4'h3 This function supports the SDIO Type-B for Bluetooth standard interface 4'h4 This function supports the SDIO GPS standard interface 4'h5 This function supports the SDIO Camera standard interface 4'h6 This function supports the SDIO PHS standard interface	4'h7

			4'h7 This function supports the SDIO WLAN interface 4'h8 This function supports the Embedded SDIO-ATA standard interface	
	RFU	RO	[5:4],RFU	2'b00
	Function supports CSA	RO	[6], 0: Fn1 does not support CSA 1: Fn1 support and CSA Can be programmed through register CIA[16]	1'b0
	Function CSA enable	RW [7],	0: Do not allow access to CSA 1: Allow access to CSA	1'b0
0X101 Extended standard I/O device type code		RO	[7:0], extension of I/O Device Interface Code can be programmed by register CIA[24:17]	8'b0
0X102 SPS		RO	[0], indicates whether Fn1 has power consumption selection 0: no power selection 1: There are two power consumption options, which can be selected through EPS can be programmed by register CIA[25]	1'b0
	EPS	RW [1], power selection	0: Fn1 works in high current mode 1: Fn1 works in low current mode	1'b0

		RO	[7ÿ2]ÿRFU	6'b0
0X103- 0X108		RO	RFU	0
0X109- 0X10B	Address pointer to function CIS1	RO	[16:0], the CIS address pointer of Fn1, namely CIS1, instructs the host to access the CIS of Fn1 initial address. The storage method is LSB segment format.	17'h02 000
		RO	[23ÿ17],RFU	7'b0
0X10C · 0X10E	Address pointer to function CSA	RW [23:0], pointing to the 24bit address pointer of CSA, the host accesses through the CSA access window After asking CSA, the pointer will automatically increase by 1. Addresses are stored in small segment format (LSB)		24'h00 0000
0X10F	Data access window to CSA	RW [7:0], read and write window to CSA. When writing to this address, the corresponding data will be Write the address indicated by the CSA 24bit address pointer through this window. During read operation, read data from the address indicated by the 24bit CSA address pointer, through This window is sent to the host.		8'b00
0X110- 0X111	Function1 IO Block Size	RW [15:0], 16bit register, used to set the IO block size. biggest The block size is 2048Byte, the minimum is 1. The data is stored in little-endian mode (LSB).		16'b0
0X100 0- 0X101 0	CIS0	RO The host accesses the CIS address space of Fn0, that is, the host accesses through this address space segment CIS0. This SDIO card supports up to 17 bytes of CIS0		
0X200 0-	CIS1	The RO host accesses the CIS address space of Fn1, that is, the host accesses through this address space segment CIS1. This SDIO card supports CIS1 byte data of 55~308.		

0X2133				
RFU			RFU	

11.4.3 SDIO Fn1 register

The Fn1 register is the address space allocated to function1 by the SDIO protocol, and its address range is: 0x00000~0x1FFFF, a total of 128K.

Since the AHB bus address width inside the chip is 32 bits, SDIO cannot use 17-bit addresses to directly access the inside of the chip, so in

In the design, an address mapping needs to be completed. The specific mapping relationship is as follows: (FN1 access space)

Table 91 SDIO Fn1 address mapping relationship

SDIO host access window	Corresponding to the actual physical address space	actual physical address space content
0X0000 ~ 0X00FF	0X0000 ~ 0X00FF	SDIO module internal register address space.
0X1000 ~ 0X1FFF	Configurable	CIS0 physical space, the specific physical space is configured by firmware.
0X2000 ~ 0X2FFF	Configurable	CIS1 physical space, the specific physical space is configured by firmware.
0X4000 ~ 0X4FFF	Configurable	Downlink and uplink cmd physical space, specific physical space The address is configured by firmware.
0X5000 ~ 0X5FFF 0X15000 ~ 0X15FFF	variable	Send buffer address space, according to sdio_txbd instructions in .
0X6000 ~ 0X7FFF 0X16000 ~ 0X17FFF	variable	Receive buffer address space, according to sdio_rxbd instructions in .
0X8000 ~ 0X9FFF	0X0E000000 ~ 0X0E002000	AHB bus config address space.
0XA000 ~ 0XBFFF	0X0F000000 ~ 0X0F002000	AHB bus APB address space.

Drivers should avoid accessing space beyond the above range, as doing so may bring unexpected results.

Among them, the first address space register is inside SDIO, and can only be accessed by SDIO HOST; other address space accesses will be based on

The description is mapped to other spaces inside the chip.

This section only introduces the registers in the SDIO 0x0000 ~ 0x00FF address space, which are controlled by the SDIO host through the CMD52 command

For direct access, the offset address is the access address, and the function number is 1.

Table 92 Some registers of SDIO Fn1 (accessed by HOST)

offset address name		Bit width access description			reset value
0X00~0X03			RO	RSV	
0X04		[7:1]	RO	RSV	7'b0
	int_read_data	[0]	RW	Uplink data interruption. Active high, write 1 to clear. When reading 0x1C, it will also be automatically cleared to 0.	1'b0
0X05		[7:1]	RO	RSV	7'b0
	int_mask	[0]	RW	corresponds to the mask enable signal of int_src. 1 to mask the corresponding interrupt. 1'd0	
0X06		[7:2]	RO	RSV	6'b0
	wlan_awake_stts [0]		RO	current WLAN status: 1 is ACTIVE; 0 is SLEEP.	1'b1
0X1C	dat_len0	[6:0]	RO	uplink data length is 7 bits higher. A total of 12 bits, the lower 5 bits are at 0x1D middle.	7'b0
	dat_vld	[7]	RO	1'b1	1'b1

0X1D	dat_len1	[7:3]		The length of RO upstream data is 5 bits lower. A total of 12 bits, the high 7 bits are at 0x1C middle.	5'b0
		[2:0]	RO	RSV	3'b0
0X1E			RO	RSV	
0X1F		[7:2]	RO	RSV	6'b0
	down_cmdbuf_vl d	[1]		RO Downlink command buffer is available, 1 is valid.	1'b1
	txbuf_vld	[0]		RO downlink data buffer is available. 1 is active, indicating that there is a send bufferÿ	1'b0
0X20	wlan_wake_en	[0]		RW Enable chip wake-up issued by SDIO in SLEEP state, high effective. After the chip wakes up, this bit will be automatically cleared to 0 by hardware.	1'b0
0X21		[7:1]	RO	RSV	7'b0
	fn1_rst	[0]		RW soft reset, 1 is valid. After the software writes 1, (that is, the chip wlan part of the circuit) is reset, write 0 After that, function1 reset is released.	1'b0
0X22		[7:1]	RO	RSV	7'b0
	fn1_recov	[0]		RW Error recovery enable, 1 is valid, after the command response ends, the Bit is automatically cleared to 0. This function is used to complete the same function as fn0/fn1 io abort. When cmd is abnormal or the command is terminated early, this register can be set to 1, to complete the io abort operation. Due to the limitation of io abort command in some bus driver versions	1'b0

				(that is, the register access address space is limited), and the user driver is not allowed to use At this time, writing 1 to this register can replace the io abort operation, and produces the same effect.	

11.4.3.1 SDIO AHB Interface Slave Registers

The following registers are used when SDIO slave device is initialized.

When transmitting data, it needs to be used in conjunction with the wrapper controller. For the part of the Wrapper controller, refer to the HSPI part of the documentation.

Table 93 SDIO AHB bus registers

offset address	name	Bit width	access	description	reset value
0X0000 0X0004			RO	Rsv	
0X0008	CIS function0 address	[31:0]	RW	The offset address of CIS0 stored in the internal memory of the system. CIS0 actual storage start address = 0x01000 (read command start address address) + the offset address	32'b0
0X000C	CIS function1 address	[31:0]	RW	The offset address of CIS1 stored in the internal memory of the system. CIS1 actual storage start address = 0x02000 (read command start address address) + the offset address	32'b0
0X0010	CSA address	[31:0]	When RW firmware is initialized, set the cheap address to access CSA, its principle is the same as CIS settings are the same. CSA is not supported in this design.		32'b0
0X0014	Read address	[31:0]	RW	is used to set the starting address for DMA to read data from memory. With 32'b0	

				Combining the Data Port register can realize the internal memory of the system Read operation (that is, the access address is 0x00+ Read address). in this In the design, this method is not used, so the default is 0	
0X0018	Write address [31:0] RW is used to set the starting address for DMA to write data to memory. Cooperate			The Data Port register can realize writing to the internal memory of the system operation (that is, the access address is 0x00+ write address). in this setting In the calculation, this method is not used, so the default is 0	32'b0
0X001C	AHB Transfers count	[20:0] RW		The SDIO device informs the host that in the read data operation to be initiated, it needs How many bytes of data to read. [23:21]RFU	32'b0
0X001F		RO	-	rsv	
0X0020	SDIO Transfer count	[20:0] RO In a data transmission, the bytes sent from the host to the SDIO device		number. When the data transmission is completed, the internal high-speed device through this register The number of bytes sent. [31:21]RFU	32'b0
0X0024	CIA register	-	RW	[3:0]: CCCR Revision. Default is 4'h2 4'h0 CCCR/FBR Version 1.00 4'h1 CCCR/FBR Version 1.10 4'h2 CCCR/FBR Version 1.20 [7:4] SDIO Revision. Default is 4'h3 4'h0 SDIO Specification 1.00	32'h06017 232

				4'h1 SDIO Version 1.10 4'h2 SDIO Version 1.20 4'h3 SDIO Version 2.0 [11:8] SD Revision. Default is 4'h2 4'h0 SD Physical Specification 1.01 4'h1 SD Physical -Spec-1.10 4'h2 SD Physical Spec 2.0 [15:12] IO-Device Code. The default is 4'h7, that is, this product is Wi-Fi devices. [16]csa_support, the default is 1. 0: does not support CSA 1: Support CSA During initialization, the firmware needs to configure this register to be 0. [24:17],Extended IO-device code The default is 8'b0. Extension of the standard IO-device code of Fn1. [25], SPS, the default is 1, that is, Fn1 supports high power consumption 0: not supported	

				1: support [26], SHS, the default is 1, supporting high speed 0: not supported 1: support [31:27]:RFU	
0X0028	Program Register	-	RW	[0], function ready. hc8051 is required to complete SD initialization After the desired Fn1 register is configured, set this register to send to the SD host Indicates that Fn1 is ready to work. Default is 0 0:Fn1 not ready 1ÿFn1 ready [1], fun1 read data ready. Fn1 is ready to send to the SD host When sending data, set this register. After the host responds to the read interrupt, the read Interrupt source register (Interrupt Identification), the bit from Automatically reset. The default is 0. 0: no data sent to the host 1: There is data sent to the host. [2], SCSI. Supports continuous SPI interrupts. The default is 1. 0: not supported 1: support [3], SDC. Indicates that the SDIO card supports execution while data is being transmitted CMD52 command. Default is 1 0: not supported	16'h 02fc

				1: support [4], SMB. SDIO card supports block transmission of CMD53. default is 1. 0: not supported 1: support [5], SRW. Indicates that the SDIO card supports read waiting. The default is 1. 0: not supported 1: support [6], SBS. Indicates that the SDIO card supports suspend/resume. Default is 1 0: not supported 1: support [7], S4MI. Indicates that the SDIO card supports data transmission in 4bit multi-block An interrupt is generated on output. The default is 1. 0: not supported 1: support [8], LSC. Indicates that the SDIO card is a low-speed device. The default is 0. 0: full speed device 1: Low speed device [9], 4BLS. Indicates that the SDIO card is a low-speed device, but supports 4bit data transmission. The default is 1. 0: not supported 1: support	

				[10], card ready. Signals belonging to the SD clock domain, power-on reset After release, this register automatically becomes 1, indicating that SDIO (Interface points) are ready. When the firmware detects this signal, it can configure the initial Fn1 register required for initialization. It is 0 at power-on reset. [15:11], RFU. Default is 0	
0X0034	OCR register	-	RW [23:0], working status register, internally programmable, mainly used to communicate with	Host operating voltage range matches. Default: 24'hff8000. Register Bit Support Voltage Range 0-3 Reserved 4 Reserved 5 Reserved 6 Reserved 7 Reserved 8 2.0-2.1 9 2.1-2.2 10 2.2-2.3 11 2.3-2.4 12 2.4-2.5 13 2.5-2.6 14 2.6-2.7 15 2.7-2.8	32'h00ff80 00

				16 2.8-2.9 17 2.9-3.0 18 3.0-3.1 19 3.1-3.2 20 3.2-3.3 21 3.3-3.4 22 3.4-3.5 23 3.5-3.6 [31:24]ÿ8'b0	
0X0038		RW	-	rsv	32'h0
0X003C	CD_State Register	-	RW	[0] indicates the state of the pull-up resistor on the SD dat[3] data line. 0: pull-up is valid 1: pull-up invalid All on-chip AHB buses can access this register. Note: The result of AHB's write operation to this register will indirectly modify CCCR7[7] CD value. [31:1],RFU	32'b0
0X0040	Fn1_Ena Register	-	RW	[0], indicates whether Fn1 is enabled or not. 0: disabled 1: enable	32'b0

				All on-chip AHB buses can access this register. Note: The result of AHB's write operation to this register will indirectly modify CCCR1[1] IOE1 value. [31:1],RFU	

12 HSPI/SDIO Wrapper Controller

12.1 Function overview

Cooperate with the interface controller (SDIO and HSPI) to complete the DMA operation of data between the host and the internal cache of the chip. Including uplink and downlink data cache

The interactive control of software and hardware, the filling and release of sending and receiving buffers, the generation of uplink data interruption, etc.

Both SDIO and HSPI exchange data with the host computer through the wrapper controller, and the control commands and processes of the two are the same. to describe

For the convenience of description, some registers are prefixed with SDIO. The corresponding register operations are also applicable to HSPI

It should be noted that for the prefix SDIO_TX related registers, it controls the device to receive data, and for SDIO_RX related registers,

Control is the device sending data.

For the cmd field that appears in the register, it is only distinguished from the data frame in terms of description, and it does not mean that the command channel can only transmit commands, or it can

Used to transfer data. The difference between commands and data here is that the command channel has only one buffer area, and the buffer length is generally less than 256

bytes, while the data channel has multiple cache areas, each of which has a length of more than 1K. Multiple caches form a linked list structure, and the specific length

Configured by software. Due to the linked list cache structure of the data frame, the transmission rate will be faster than the command channel.

12.2 Main Features

- ÿ Support word-aligned data movement
- ÿ Support DMA function
- ÿ Support linked list structure management
- ÿ Support interrupt generation
- ÿ Can receive up to 4096 bytes of data

12.3 Functional description

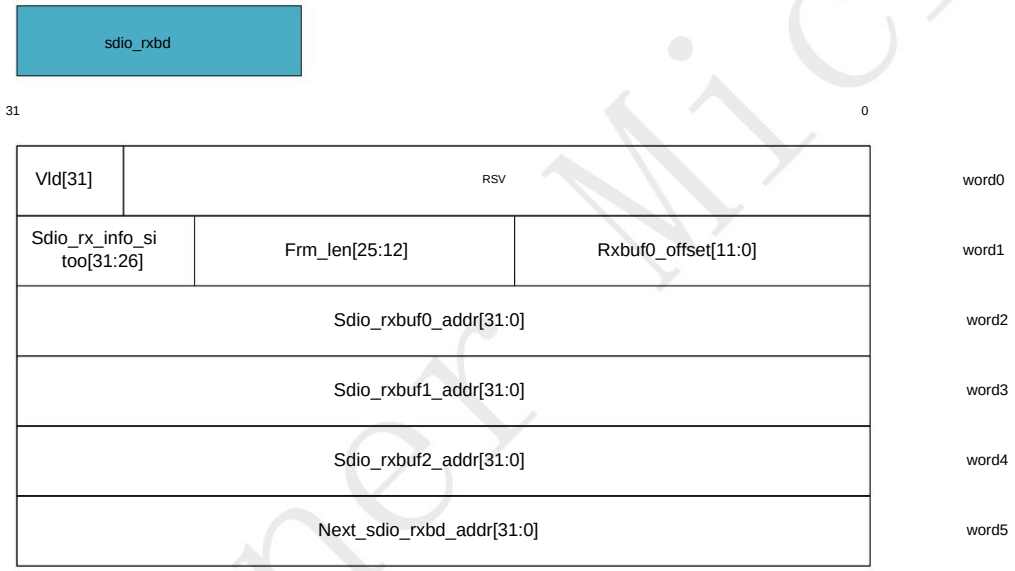
12.3.1 Uplink data receiving function

The upstream direction refers to the direction in which the master device (SDIO or HSPI) sends data to the slave device (W800).

When the master device sends data to the slave device, after the SDIO or HSPI module receives the data, it will link the data to the interface through WRAPPER.

Receive BD, and generate an interrupt to notify the application software to process the data.

Receive BD descriptor:



Note: 1.
vld, 1 is valid, indicating that the current descriptor points to a valid received frame. 2.
rxbuf0_offset: byte address, word alignment, indicating the offset address of the word where the first valid byte of the 802.3 frame is located relative to sdio_rxbuf0. 3.sdio_rxbuf0_addr: byte address, word alignment, buf base address of the first fragment of the upstream data frame. 4. sdio_rxbuf1_addr: byte address, word alignment, buf base address of the second fragment of the uplink data frame. 5.sdio_rxbuf2_addr: byte address, word alignment, buf base address of the third fragment of the uplink data frame. 6.next_sdio_rxbd_addr, byte address, word alignment, base address of the next sdio_rxbd. 7. frm_len, byte length, indicates the length of uplink data, excluding sdio_rx_info.

8. Sdio_rx_info_size, byte length, indicates the number of sdio_rx_info bytes, which must be an integer multiple of 4 bytes.
Note: The longest received frame is 4096, occupying up to three fragments. Rxbuf0_offset is only valid for the first fragment of the stored frame (that is, the fragment in sdio_rxbuf0), and for the remaining two fragments, the data is stored sequentially from the base address. The hardware should determine whether the frame exists in multiple fragments according to the length of the uplink data and the fixed size of each buf of 1600 bytes.

Figure 26 SDIO receive BD descriptor

When the SDIO module or HSPI module of W800 detects that the receiving enable is valid, it reads RXBD and judges the Vld flag.

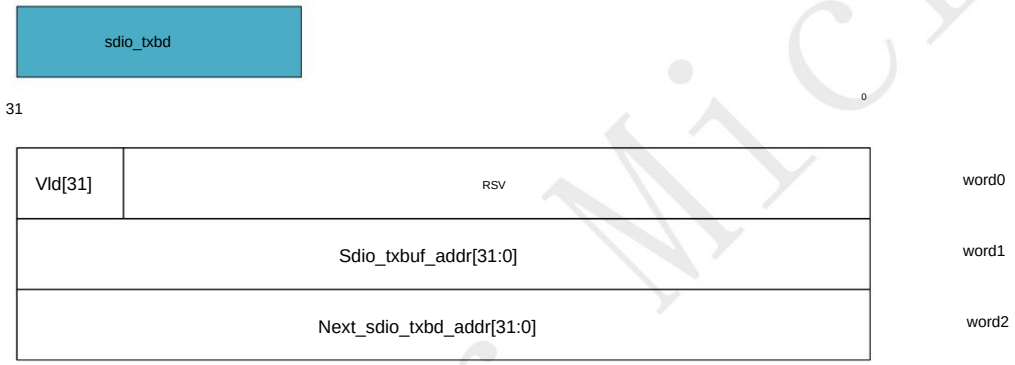
12.3.2 Downlink Data Migration Function

When W800 has data to send to the master device, the software prepares the sending description first, and then notifies WRAPPER to send the data

Relocation, WRAPPER notifies the master device to read the device to be sent through SDIO or HSPI interrupt signal, when the data transmission is completed,

WRAPPER generates a transmit completion interrupt notification routine.

Send BD descriptor:



Note: 1.

vld, 1 is valid, indicating that the current descriptor points to an available send buffer.

2. Sdio_txbuf_addr, byte address, must be word-aligned. Indicates that this descriptor points to the base address where the sent data is stored. This address is offset relative to the base address of each sdio_txbuf to ensure that the base address of the sdio_txbuf is not exceeded when the firmware adds the LLC up at the beginning of the frame.

3.next_sdio_txbd_addr, byte address, word alignment. The base address of the next sdio_txbd.

Figure 27 SDIO sends BD descriptor

12.4 Register Description

12.4.1 Register List

Table 94 WRAPPER controller registers

offset address	name	abbreviation	access	describe	reset value
0X0000	WRAPPER interrupt status register	INT_STTS	RW	command or data frame interrupt status	0X0000

0X0004 WRAPPER interrupt configuration register INT_MASK			Whether the RW command or data frame interrupt is masked		0X0000
0X0008	WRAPPER Uplink command ready send memory	UP_CMD_AVAIL	RW whether the uplink command is ready		0X0000
0X000c	WRAPPER downlink command buf is thread register	DOWN_CMD_BUF_A VAIL	RW Whether the downlink command buf is ready		0X0000
0X0010 SDIO_TX link enable register		SDIO_TX_BD_LINK_E N	RW indicates whether the sdio_txbd linked list descriptor is	efficient	0X0001
0X0014 SDIO_TX link address register SDIO_TX_BD_ADDR			RW The address pointed to by the current sdio_txbd needs to be	For word alignment, configuration is required during initialization	0X0000
0X0018 SDIO_TX enable register		SDIO_TX_EN	RW SDIO transmit frame enable		0X0000
0X001c SDIO_TX status register		SDIO_TX_STTS	RO	SDIO send status	0X0000
0X0020 SDIO_RX link enable register		SDIO_RX_BD_LINK_E N	RW indicates whether the sdio_rxbd linked list descriptor	efficient	0X0001
0X0024 SDIO_RX link address register SDIO_RX_BD_ADDR			RW The address pointed to by the current sdio_rxbd needs to be	For word alignment, configuration is required during initialization	0X0000
0X0028 SDIO_RX enable register		SDIO_RX_EN	RW SDIO receive frame enable		0X0000
0X002c SDIO_RX status register		SDIO_RX_STTS	RO	SDIO receive status	0X0000
0X0030	WRAPPER CMD BUF base address register	CMD_BUF_BASE_AD DR	RW downlink cmd buf base address		0X0000
0X0034	WRAPPER CMD BUF SIZE register	CMD_BUF_SIZE	RW Cmd buf byte size		0X0064

12.4.2 WRAPPER INTERRUPT STATUS REGISTER

Table 95 WRAPPER interrupt status register

bit	access	Instructions	reset value
[31: 4] RO		reserve	
[3]	RW	int_down_cmd down cmd frame complete interrupt. Write 1 to clear 0.	1'b0
[2]	RW	int_up_cmd Up cmd frame word complete interrupt. Write 1 to clear 0.	1'b0
[1]	RW	int_sdio_txfrm Downstream data frame complete interrupt. Write 1 to clear 0.	1'b0
[0]	RW	int_sdio_rxfrm Upstream data frame complete interrupt. Write 1 to clear 0.	1'b0

12.4.3 WRAPPER INTERRUPT CONFIGURATION REGISTER

Table 96 WRAPPER interrupt configuration register

bit	access	Instructions	reset value
[31: 4] RO		reserve	
[3]	RW	int_mask_down_cmd Down cmd frame complete interrupt mask register. 1 is shielding, the same below.	1'b0
[2]	RW	int_mask_up_cmd Up cmd frame complete interrupt mask register.	1'b0
[1]	RW	int_mask_sdio_txfrm Downstream data frame complete interrupt mask register.	1'b0
[0]	RW	int_mask_sdio_rxfrm Upstream data frame complete interrupt mask register.	1'b0

12.4.4 WRAPPER upstream command ready register

Table 97 WRAPPER uplink command ready register

bit		access	Instructions	reset value

[31: 1]		RO	reserve	
[0]		RW	Firmware sets this bit to 1 when there is an upstream cmd frame. When the upstream cmd transmission is complete, the hard The software automatically clears it to 0 and generates an int_up_cmd interrupt.	1'b0

12.4.5 WRAPPER downlink command buf ready register

Table 98 WRAPPER downlink command buf ready register

bit	access	Instructions	reset value
[31: 1] RO		reserve	
[0]	RW	After sending the downlink cmd, the hardware will clear this bit to 0 and generate an interrupt at the same time. When the firmware processing is complete this next command After command, set this bit to 1.	1'b0

12.4.6 SDIO TX link enable register

Table 99 SDIO TX link enable register

bit	access	Instructions	reset value
[31: 1] RO		reserve	
[0]	RW	sdio_txbd link enable, active high. If this bit is valid, the hardware is processing an sdio_txbd descriptor and directly processes it The next descriptor pointed to by next_sdio_txbd_addr. If this bit is invalid, the hardware is processing a After sdio_txbd, it will stop immediately. The same applies to HSPI.	1'b1

12.4.7 SDIO TX link address register

Table 100 SDIO TX link address register

bit	access	Instructions	reset value
[31: 0] RW		The current sdio_txbd byte address, the software needs to strictly ensure the word alignment, the same below. Initially, the firmware needs to configure this register. After the hardware completes a sending buf each time, it will set the sdio_txbd The next_sdio_txbd_addr in the descriptor is updated to this register. The same applies to HSPI.	32'h0

12.4.8 SDIO TX Enable Register

Table 101 SDIO TX enable register

bit	access	Instructions	reset value
[31: 1] RO		reserve	
[0]	RW	SDIO transmit frame enable, active high. After the firmware completes the construction of each descriptor sdio_txbd descriptor, set this bit to 1 to notify the SDIO module A new send descriptor exists. The SDIO module detects that this bit is valid, and starts to read the current sdio_txbd, and complete the sending frame process. Hardware clears this register to 0 automatically. The same applies to HSPI.	1'b0

12.4.9 SDIO TX Status Register

Table 102 SDIO TX Status Register

bit	access	Instructions	reset value

[31: 1] RO		reserve	
[0]	RW	SDIO send status. 0: SDIO has stopped sending process because no send descriptor is available 1: SDIO is in the process of sending The same applies to HSPI.	1'b0

12.4.10SDIO RX Link Enable Register

Table 103 SDIO RX Link Enable Register

bit	access	Instructions	reset value
[31: 1] RO		reserve	
[0]	RW	sdio_rxbd link enable, active high. If this bit is valid, the hardware is processing an sdio_rxbd descriptor and directly processes it The next descriptor pointed to by next_sdio_rxbd_addr. If this bit is invalid, the hardware is processing a After sdio_rxbd, it will stop immediately. The same applies to HSPI.	1'b1

12.4.11 SDIO RX link address register

Table 104 SDIO RX link address register

bit	access	Instructions	reset value
[31: 0] RW		Current sdio_rxbd byte address. Initially, the firmware needs to configure this register. After the hardware completes a sending buf each time, it will set the sdio_rxbd The next_sdio_rxbd_addr in the descriptor is updated to this register.	32'h0

		The same applies to HSPI.	

12.4.12 SDIO RX Enable Register

Table 105 SDIO RX enable register

bit	access	Instructions	reset value
[31: 1] RO		reserve	
[0]	RW	SDIO receive frame enable, active high. After the firmware completes the construction of each descriptor sdio_rxbd descriptor, set this bit to 1 to notify the SDIO module A new receive descriptor exists. The SDIO module detects that this bit is valid, and starts to read the current sdio_txbd, and complete the process of receiving frames. Hardware clears this register to 0 automatically. The same applies to HSPI.	1'b0

12.4.13 SDIO RX Status Register

Table 106 SDIO RX Status Register

bit	access	Instructions	reset value
[31: 1] RO		reserve	
[0]	RW	SDIO receive status. 0: SDIO has stopped receiving process because there is no valid uplink descriptor and uplink command 1: SDIO is in the process of receiving The same applies to HSPI.	1'b0

12.4.14 WRAPPER CMD BUF base address register

Table 107 WRAPPER CMD BUF base address register

bit	access	Instructions	reset value
[31: 0] RW		Downstream cmd buf base address. The base address of upstream cmd buf is the base address plus cmd_buf_size.	32'h0

12.4.15 WRAPPER CMD BUF SIZE Register

Table 108 WRAPPER CMD BUF SIZE register

bit	access	Instructions	reset value
[31:12] RO		reserve	
[11: 0] RW		The byte size of cmd buf must be an integer multiple of 4 bytes.	12'd64

13 SDIO HOST device controller

13.1 Function overview

The SDIO HOST device controller provides a digital interface capable of accessing Secure Digital Input Output (SDIO) and MMC cards.

Able to access SDIO devices and SD card devices compatible with SDIO 2.0 protocol. The main interfaces are CK, CMD and 4 data lines.

13.2 Main Features

- ÿ Compatible with SD Card Specification 1.0/1.1/2.0(SDHC)
- ÿ Compatible with SDIO Memory Card Specification 1.1.0
- ÿ Compatible with MMC specification 2.0~4.2
- ÿ Configurable interface clock rate, support host rate 0~50MHz,
- ÿ Support standard MMC interface
- ÿ Support up to 1024 bytes Block
- ÿ Support soft reset function
- ÿ Automatic Command/Response CRC generation/verification;
- ÿ Automatic data CRC generation/verification;
- ÿ Configurable timeout detection;
- ÿ Support SPI, 1-bit SD and 4-bit SD modes
- ÿ Support DMA data transfer

13.3 Functional description

13.4 Register Description

13.4.1 Register List

Offset site	register name	Name Bit Width	Attribute	Description		reset value
0x00	mmc_ctrl	RSV	[15:11]	RO		
			[10]	RW	SDIO read wait enable '1' : enable SDIO read wait '0' : Disable SDIO read wait	1'b0
			[9]	RW	SDIO interrupt enable '1' : SDIO interrupt enable '0' : SDIO interrupt disabled	1'b0
			[8]	RW	SDIO mode enable '1': SDIO '0' : SD/MMC	1'b0
			[7]	RW	SD/MMC/SDIO interface data width '1' : 4 bits '0' : 1 bit	1'b0
			[6]	RW	SD/MMC/SDIO transfer mode '1' : High-Speed Mode '0' : Low-Speed Mode	1'b1
			[5:3]	RW	SDIO/MMC/SDIO port clock rate selection Refer to Table 2	3'b000

			[2]	RW	SDIO/MMC/SDIO interface driver mode selection ‘1’ : open_DrainMode ‘0’ :Push-Pull Mode	1'b1
			[1]	RW signal mode selection	'1' : Automatically select the transfer mode '0' : Use mmc_port register to select	1'b0
			[0]	RW port mode selection	‘1’ : MMC mode ‘0’ :SPI mode	1'b1
0x04	mmc_io	RSV	RO	[15:10]		6’d0
			RW	[9]	SDIO cmd12/IO Abort flag '1' : mark the current command as cmd12/IO Abort '0' : flag current command is not cmd12/IO Abort	1'b0
			RW	[8]	SDIO Command Properties '1' : mark that there is a data block after the current command; '0' : mark no data block and command response after current command answer;	1'b0
			RW	[7]	Enable auto generate 8 null clock after response/command or single block data	1'b0

					Automatically generate 8 after a response/command or a single chunk of data null clock function '1' : enable '0' : off	
			RW	[6]	Enable auto receive response after command Automatically receive response function after command '1' : enable '0' : off	1'b0
			RW	[5]	8 nulls on SD/MMC/SDIO port clock line clock generation '1' : generate 8 null clocks '0' : Receive response/send command according to bit 3 setting	1'b0
			RW	[4]	Designed for CID and CSD reads. When sending CID or When the CSD command is issued, the SD/MMC/SDIO card device will be in CMD reply online with 136bit CID or 11 CSD data. When setting this bit to 1, CID or CSD The data will be stored in [135:8] of the command buffer area middle;	1'b0
			RW	[3]	Response/command selection when bit[5] is '0' '1': Response received	1'b0

					'0': send command.	
			RW	[2]	Set up automatic 8 null clock/command/response transfers Function '1': Enable automatic 8 null clock/command/response transfer '0': Disable automatic 8 null clock/command/response transmission lose. Generate 8 nulls according to bit 5 and bit3 settings Clock, receive response or transmit command. when pass After the input is completed, this bit is automatically cleared;	1'b0
			RW	[1]	Set data transfer direction '1' : read data;. '0' : write data;	1'b0
			RW	[0]	Set up automatic data transfer '1' : enable automatic data transfer '0' : Disable automatic data transfer. When the data transmission is completed, this bit will be automatically cleared;	1'b0
0x08	mmc_bytecntl		RW	[15:0] Data Transfer Byte Count Register		16'h0200
0x0C	mmc_tr_blockcnt		RO	[15:0] For multi-block transfer, completed block counter 16'h0000		
0x10	mmc_crcctl		RW	[7]	SD/MMC/SDIO port CMD Line CRC circuit enable. SD/MMC/SDIO port CMD line CRC function	1'b0

					can '1': enable. '0': off.	
			RW	[6]	SD/MMC/SDIO port data line CRC function '1': enable. '0': off.	1'b0
			RW	[5] Enable automatic CRC check crc_status	'1': enable, when crc_status !=3'b010, A crc status interrupt will be generated, and the write data transfer will be The stop command was interrupted and mmc_io[0] or mmc_io_mbctl[2:0] will be cleared; 0: off;	1'b0
			RW	[4]	Read multi-block function before response '1': enable. '0' : off	1'b0
			RW	[3:2]	DAT CRC selection. DAT CRC selection Refer to Table 4	1'b0
			RO	[1]	CMD CRC error.	1'b0
			RO	[0]	DAT CRC error	1'b0
0x14	cmd_crc	RSV	RO	[7]	RSV	1'b0
			RO	[6:0]	CMD CRC register value	7'd0

0x18	dat_ccl		RO	[7:0]	The DAT CRC low register value	THAT
0x1C	dat_crch		RO	[7:0]	The DAT CRC high register value	THAT
0x20	mm_port		RW	[7]	SD/MMC/SDIO port clock line signal.	1'b0
			RW	[6]	SD/MMC/SDIO port CMD line signal	1'b1
			RW	[5]	SD/MMC/SDIO port data line signal	1'b1
			RW	[4]	Automatically check Ncr timeout function 1': Enable automatic Ncr timeout check. '0' : turn off auto-check Ncr timeout	1'b1
			RW	[3:0]	Ncr timeout count value (SD/MMC/SDIO clock count).	4'hF
0x24	mmc_int_mask RSV		RO	[15:9]		7'd0
				[8]	SDIO data line 1 interrupt mask '1': no masking '0': shielded	1'b0
				[7]	CRC status token error interrupt mask '1': no masking '0': shielded	1'b0
				[6] Command and response Ncr timeout interrupt mask	'1': no masking '0': shielded	1'b0
				[5]	Multi-block timeout interrupt mask. '1': no masking	1'b0

					'0': shielded	
				[4]	Multi-block transfer complete interrupt mask. '1': no masking '0': shielded	1'b0
				[3]	Command CRC error interrupt mask '1': no masking '0': shielded	1'b0
				[2]	Data CRC Error Interrupt Mask '1': no masking '0': shielded	1'b0
				[1]	Data transfer complete interrupt mask '1': no masking '0': shielded	1'b0
				[0]	Command transfer complete interrupt mask '1': no masking '0': shielded	1'b0
0x28	clr_mmc_int	RSV	RO	[15:9]		7'd0
			RW	[8]	W: Clear SDIO data line 1 interrupt R: SDIO data line 1 interrupt status	1'b0
			RW	[7]	W: Clear CRC status token error interrupt R: CRC status token error interrupt status;	1'b0

					When this bit is 1, judge mmc_sig[6:4]	
			RW	[6] W: Clear command and respond to Ncr timeout interrupt;	R: command and response Ncr timeout interrupt status;	1'b0
			RW	[5]	W: Clear multi-block timeout interrupt;. R: multi-block timeout interrupt status;	1'b0
			RW	[4]	W: Clear multi-block transfer completion interrupt;. R: Multi-block transmission complete interrupt status;	1'b0
			RW	[3]	W: clear command CRC error interrupt;. R: command CRC error interrupt status;	1'b0
			RW	[2]	W: clear data CRC error interrupt;. R: Data CRC error interrupt status;	1'b0
			RW	[1]	W: Clear data transfer complete interrupt;. R: Data transmission complete interrupt status;	1'b0
			RW	[0]	W: Clear command transfer complete interrupt;. R: command transmission complete interrupt status;	1'b0
0x2C	mmc_cardsel		RW	[7]	SD/MMC/SDIO controller enable '1': enable	1'b0

					'0': off	
			RW	[6]	Enable SD/MMC/SDIO card device clock line '1': enable '0': off	1'b1
			RW	[5:0]	SD/MMC/SDIO Time Reference Factor Use this register to establish a 1MHz clock; 1MHz=Fhclk/((mmc_cardsel[5:0]+1)*2)	6'd0
0x30	mmc_sig		RW	[7]	SD/MMC/SDIO port CMD line signal When this register is read, the SDIO controller will A clock pulse is generated on the clock line. and on the clock The state of the CMD line will be stored to this register at the rising edge device.	1'b1
			RW	[6:4]	CRC status[2:0] When writing data CRC status token time;	3'b111
			RW	[3]	SD/MMC/SDIO port DAT3 data signal. 1'b1	
			RW	[2]	SD/MMC/SDIO port DAT2 data signal.	1'b1
			RW	[1]	SD/MMC/SDIO port DAT1 data signal.	1'b1
			RW	[0]	SD/MMC/SDIO port DAT0 data signal.	1'b1
0x34	mmc_io_mbctl		RW	[7:6]	SD/MMC/SDIO NAC timeout range selection 2'b0	

					Reference Appendix 2 -Table 5.	
			RW	[5:4]	SD/MMC/SDIO Busy timeout scale selection. SD/MMC/SDIO device busy timeout range selection. Reference Appendix 2 -Table 6	2'd1
			RW	[3]	SD/MMC/SDIO port clock line polarity 1: Send on the falling edge of the clock and collect on the rising edge; 0: Send on the rising edge of the clock, collect on the falling edge;	1'b0
			RW	[2]	Set SD/MMC/SDIO port fully automatic command and multi-block transfer '1': enable '0': off Setting this bit to 1 (mmc_io[7:6]==11) will trigger Send a SD/MMC/SDIO command, response, 8 null Clock, multi-block transfers. When the data transfer is complete After completion, this bit will be automatically cleared.	1'b0
			RW	[1]	Select multiple block data transfer direction. Set the direction of multi-block transfer '1' : read data. '0' : Write data.	1'b0

			RW	[0]	Set SD/MMC/SDIO port auto multiple block data ransfer. Set SD/MMC/SDIO port auto multi-block transmission '1' : enable. '0' : off. Setting this bit to 1 (mmc_io[7:6]==11) will trigger Send a SD/MMC/SDIO multi-block transfer. The number of data blocks is in mmc_blocknt set in the register. When the data transfer is complete, this bit will Automatically cleared.	1'b0
0x38	mmc_blockcnt		RW	[15:0]	Data block number register. Data Block Number Register Configuring this register defines a total of The number of data blocks to transmit.	16'h0001
0x3C	mmc_timeoutcnt		RW	[7:0]	Data transfer timeout count register. Data Transfer Timeout Counter Time = Scale* bit[7:0]. Scale via register mmc_io_mbctl[7:6]/[5:4] definition;	8'h40
0x40	cmd_buf0		RW	[7:0]	The cmd_buf byte 0. Mapped to command line bit [15:8]	8:00 a.m

					command buf byte 0 , mapped to the command line bit[15:8]	
0x44	cmd_buf1		RW	[7:0]	The cmd_buf byte 1. Mapped to command line bit [23:16] command buf byte 1 , mapped to the command line bit[23:16]	8:00 a.m
0x48	cmd_buf2		RW	[7:0]	The cmd_buf byte 2. Mapped to command line bit [31:24] command buf byte 2 , mapped to the command line bit[31:24]	8:00 a.m
0x4C	cmd_buf3		RW	[7:0]	The cmd_buf byte 3. Mapped to command line bit [39:32] command buf byte 3 , mapped to the command line bit[39:32]	8:00 a.m
0x50	cmd_buf4		RW	[7:0]	The cmd_buf byte 4. Mapped to command line bit [47:40] command buf byte 4 , mapped to the command line bit[47:40]	8:00 a.m
0x54	cmd_buf5		RW	[7:0]	The cmd_buf byte 5. Mapped to command line bit [55:48] command buf byte 5 , mapped to the command line bit[55:48]	8:00 a.m

0x58	cmd_buf6		RW	[7:0]	The cmd_buf byte 6. Mapped to command line bit [63:56] command buf byte 6 , mapped to the command line bit[63:56]	8:00 a.m
0x5C	cmd_buf7		RW	[7:0]	The cmd_buf byte 7. Mapped to command line bit [71:64] command buf byte 7 , mapped to the command line bit[71:64]	8:00 a.m
0x60	cmd_buf8		RW	[7:0]	The cmd_buf byte 8. Mapped to command line bit [79:72] command buf byte 8 , mapped to the command line bit[79:72]	8:00 a.m
0x64	cmd_buf9		RW	[7:0]	The cmd_buf byte 9. Mapped to command line bit [87:80] command buf byte 9 , mapped to the command line bit[87:80]	8:00 a.m
0x68	cmd_buf10		RW	[7:0]	The cmd_buf byte 10. Mapped to command line bit [95:88] Command buf byte 10, mapped to the command line bit[95:88]	8:00 a.m
0x6C	cmd_buf11		RW	[7:0]	The cmd_buf byte 11. Mapped to command line bit [103:96]	8:00 a.m

					Command buf byte 11, mapped to the command line bit[103:96]	
0x70	cmd_buf12		RW	[7:0]	The cmd_buf byte 12. Mapped to command line bit [111:104] Command buf byte 12, mapped to the command line bit[111:104]	8:00 a.m
0x74	cmd_buf13		RW	[7:0]	The cmd_buf byte 13. Mapped to command line bit [119:112] Command buf byte 13, mapped to the command line bit[119:112]	8:00 a.m
0x78	cmd_buf14		RW	[7:0]	The cmd_buf byte 14. Mapped to command line bit [127:120] Command buf byte 14, mapped to the command line bit[127:120]	8:00 a.m
0x7C	cmd_buf15		RW	[7:0]	The cmd_buf byte 15. Mapped to command line bit [135:128] Command buf byte 15, mapped to the command line bit[135:128]	8:00 a.m
0x80	buf_ctrl		RW	[15] Data buffer clear enable	1: Trigger data buffer clearing; 0: keep When writing 1, this register is automatically cleared after one clock	1'b0

					zero;	
			RW	[14]	DMA request masking 0: no shielding; 1: shielded; Note: Please configure this register before enabling dma; Configuring this register after enabling dma has no effect;	1'b0
			RW	[13]	RSV	1'b0
			RW	[12] Data FIFO status signal mask configuration	1: activate 0: default Data FIFO status signal, high effective; Shield the FIFO full signal when reading the card device; Shield the FIFO empty signal when writing to the card device;	1'b0
			RW	[11] Set buffer access direction	1: write 0: read	1'b0
			RW	[10]	DMA hardware interface enable: 1: Enable DMA interface; 0: AHB interface accesses data cache; When using the DMA interface, when the data transfer is complete the Automatically reset this bit (single data block transfer or multiple data	1'b0

					block transfer)	
			RW	[9:2] Data cache data watermark setting; only when Valid when buf_ctl[10]=1; Note: The data cache depth is 128 words, not To configure this register is greater than 127.	8'd0
			RO	[1] Data buffer empty signal		1'b1
			RO	[0] Data buffer full signal		1'b0
0x100~ 0x2FF	data_buf		RW		data cache	THAT

Bit5	Bit4	Bit3	Speed
0	0	0	1/2 base clock
0	0	1	1/4 base clock
0	1	0	1/6 base clock
0	1	1	1/8 base clock
1	0	0	1/10 base clock
1	0	1	1/12 base clock
1	1	0	1/14 base clock
1	1	1	1/16 base clock

Table 2 Mmc_ctrl[5:3] detailed definition

Note: When mmc_ctrl[6] = 1 and the controller works in high speed mode, base clk = hclk;

When mmc_ctrl[6] = 0, when the controller works in low speed mode, base clk = clk1m;

Clk1m= Fhclk/((mmc_cardsel[5:0]+1)*2)ÿ

Bit7 Bit6	Bit5 Bit4	Bit3 Bit2	Bit1 Bit0	Operation mode	transmission byte			
x	x	x	x	x	0	x	0 no action	N/A
x	x	1	x	0	Trig x		0 generates 8 null clk	N/A
0	0	0	x	0	Trig x		0 send command	6
0	0	0	0	1	Trig x		0 received response	6
0	0	0	1	1	Trig x		0 received response	17
1	0	0	0	0	Trig x		0 transfer command + generate 8 null clk	N/A

0	1	0	0	0	Trig x		0 transmit command + receive ring answer		N/A
1	1	0	0	0	Trig x		0 transmit command + receive ring should + generate 8 null clk		N/A
x	x	x	x	x	0	1	Trig reads a single data+8 null clk		mmc_bytecnt
x	x	x	x	x	0	0	Trig writes a single data +8 null clk		mmc_bytecnt

Table 3 Mmc_io[7:0] detailed definition

Noteÿ

1. Except for the last two lines in Table 3, other operations will generate a CMD DONE interrupt;

2. The last two operations in Table 3 will generate a data transfer completion interrupt.

Bit3 Bit2		**data_crcl and data_crch register display contents**
0	0	DAT0 Online data CRC value
0	1	DAT1 online data CRC value
1	0	DAT2 Online data CRC value
1	1	DAT3 online data CRC value

Table 4 mmc_crctrl[3:2] detailed definition

Bit7	Bit6	Bit2	Bit1 Bit0		Operation description	Bytes transferred
mmc_io		mmc_io_mbctl				
1	1	Trig	0	0 write	multi-block command+response+8 null clock+data	mmc_blockcnt
1	1	Trig	1	0 read	multi-block command+response+8 null clock+data	mmc_blockcnt
x	x	0	0	Trig	write multiple blocks of data	mmc_blockcnt
x	x	0	1	Trig	read multiple blocks of data	mmc_blockcnt

Table 5 mmc_io[7:6] and mmc_io_mbctl[2:0] detailed definition

Noteÿ

1. The first two column operations in Table 5 will generate multi-block data completion interrupts, and each block of data will generate data completion interrupts, and CMD DONE interrupt;

2. When a timeout occurs, the multi-block data completion interrupt will not be generated, but a timeout interrupt will be generated.

Bit7	Bit6	time unit
0	0	1us
0	1	100us
1	0	10ms
1	1	1s

Table 6 mmc_io_mbctl[7:6] NAC timeout interrupt unit selection

Bit7	Bit6	time unit
0	0	1us
0	1	100us
1	0	10ms
1	1	1s

Table 7 mmc_io_mbctl[5:4] port timeout interrupt unit selection

Noteÿ

1. When using the DMA interface, the DMA enable needs to be turned on first;

2. If interrupt is not used, mmc_io[2] can be queried when transmitting command/response/8 null clock; when needed

When transferring data, you can query mmc_io[0]; when transferring multiple blocks of data, you can query mmc_io_mbctl[2]

/mmc_io_mbctl0]ÿ

3. When the Ncr timeout occurs, the data transmission will be interrupted, and the controller needs to reconfigure a new transmission;

14 SPI controllers

14.1 Function overview

SPI is an acronym for Serial Peripheral Interface. SPI is a high-speed, full-duplex, synchronous communication bus

Wire. The communication principle of SPI is very simple. It works in a master-slave mode. This mode usually has a master device and one or more slave devices. It needs

At least 4 wires, in fact, 3 wires are also possible (during one-way transmission), including SDI (data input), SDO (data output), SCLK (time

clock), CS (chip select).

14.2 Main Features

- ÿ Can be used as both SPI master and SPI slave
- ÿ 8-word-deep FIFOs for transmit and receive paths
- ÿ master supports 4 formats of motorola spi (CPOL, CPHA), TI timing, macrowire timing
- ÿ slave supports 4 formats of motorola spi (CPOL, CPHA)
- ÿ Support full duplex and half duplex
- ÿ The master device supports bit transmission, up to 65535bit transmission
- ÿ The slave device supports transmission modes of various byte lengths
- ÿ The maximum clock frequency of spi_clk input from the device is 1/6 of the system APB clock

14.3 Functional description

14.3.1 Master-slave can be configured

The SPI controller supports both the device as a SPI communication master and the device as an SPI communication slave. Registered by setting SPI_CFG

Bit2 of the device can switch back and forth between the master and slave roles of the device.

14.3.2 Multiple Mode Support

When acting as a master device, by setting Bit1 (CPHA) and Bit0 (CPOL) of the SPI_CFG register, it can be used as MOTOROLA

Four formats of SPI transmit data. CPOL is used to determine the level of the SCK clock signal when it is idle, CPOL=0, the idle level is low,

When CPOL=1, the idle level is high level. CPHA is used to determine the sampling moment, CPHA=0, at the first clock edge of each cycle

Sampling, CPHA=1, sampling on the second clock edge of each cycle. The master can also be set by setting the TRANS_MODE register

The data is transmitted in TI timing or microwire timing, and the length of transmission data under the two timings can be adjusted.

As a slave device, it only supports the four formats of MOTOROLA SPI, and the format selection is also done by setting the same register as the master device

memory to achieve.

14.3.3 Efficient Data Transfer

The FIFO memory is a first-in-first-out dual-port buffer, that is, the first data that enters it is the first to be moved out, and one of the memory

One is the input port, and the other is the output port of the memory. The SPI controller integrates two FIFOs with a depth of 8 words each (one for each transceiver) to store

In order to increase the data transmission rate, handle a large number of data streams, and match systems with different transmission rates, the system performance is improved. able to pass

Setting the Bit[8:6] and Bit[4:2] of the MODE_CFG register can set the trigger level of RXFIFO and TXFIFO to meet different

Performance requirements at the same transmission rate. After the trigger level of the FIFO is triggered, an interrupt or DMA can be triggered to transfer data from the memory

Move to TXFIFO or move data from RXFIFO to memory.

14.4 Register Description

14.4.1 Register List

Table 109 SPI register list

offset address name	abbreviation	access	describe	reset value
0X0000 channel configuration register CH_CFG		RW is used to carry out some related items on the transceiver channel configuration		0X0000_0000
0X0004	SPI configuration register SPI_CFG	RW	Configure SPI communication related items	0X0000_0004
0X0008 Clock configuration register Clk_CFG		RW is used to set the clock frequency division factor		0X0000_0000
0X000C mode configuration register MODE_CFG		RW configuration transfer mode		0X0000_0000
0X0010 Interrupt control register SPI_INT_MASK		RW mask or enable related interrupt		0X0000_00FF
0X0014 Interrupt status register SPI_INT_SOURCE	RW is used to query the interrupt source			0X0000_0000
0X0018	SPI status register SPI_STATUS	RO enumerates the relevant status in SPI communication		0X0000_0000
0X001C	SPI timeout register SPI_TIME_OUT	RW Set SPI communication timeout		0X0000_0000
0X0020 Data transmission register SPI_TX_DATA		RW	TX FIFO, used to store data to be sent 0X0000_0000	
0X0024 transfer mode register TRANS_MODE		RW set transfer mode		0X0000_0000
0X0028 Data length register SLV_XMIT_LEN		When RO is used as a slave device, it is used to store and send out or The length of the received data		0X0000_0000
0X002C	RSV		reserve	0X0000_0000
0X0030 Data receiving register SPI_RX_DATA		RW/RO RX FIFO, used to store received data 0X0000_0000		

14.4.2 Channel configuration register

Table 110 SPI channel configuration register

bit access		Instructions	reset value
[31]		RSV	1'b0
[30:23] RW		RX_INVALID_BIT	8'h0

		Indicates how many first bits are invalid data when the receiving channel starts to receive, and these invalid data need to be thrown directly out, does not enter the Rx FIFO. Only subsequent data enters the Rx FIFO This register is used together with Tx/Rx length. Finally, the amount of data actually stored in the Rx FIFO is Tx/Rx length - RX_INVALID_BIT Note: master mode is valid Motorola/TI mode valid	
[22]	RW	Clear FIFOs, clear the contents of Tx and Rx FIFO, and simultaneously reset all circuits of master and slave (except configuration registers) 1'b0: Do not clear FIFO 1'b1: clear valid Set to 1 by software, cleared to 0 by hardware Note: master/slave are both valid Motorola/TI/microwire mode valid	1'b0
[21]	RW	continue mode, in this mode, spi sending is not affected by Tx FIFO empty. , continue to transmit until the entire The transfer process is complete. 1'b0: normal, after the Tx FIFO is empty, it needs to wait for data to appear in the FIFO, and SCK stops toggling. Similarly, After the Rx FIFO is full, SCK stops toggling and waits for the RX FIFO to have space to receive data 1'b1: continue mode, Tx fifo is empty, can still transmit until the transmission is completed, but at this time if the rx fifo If it is full, you need to suspend the transmission until the rx fifo can store the number Note: master is valid Normally, this mode is not set.	1'b0

		When this mode is turned on, if there is no data in the tx fifo, invalid data may be sent out first. so please Fill in the data first, then start the spi master Motorola/TI/microwire mode valid	
[20]	RW	RxChOn, whether the receiving channel is open 1'b0: Rx channel off 1'b1: Rx channel on Note: master/slave are both valid Motorola/TI/microwire mode valid	1'b0
[19]	RW	TxChOn, whether the sending channel is open 1'b0: Tx channel off 1'b1: Tx channel on Note: master/slave are both valid Motorola/TI/microwire mode valid	1'b0
[18:3] RW		Tx/Rx length When Spi is transmitting, the valid SCK number It also indirectly reflects the length of the sent or received data. When (TxChOn=1, RxChOn=1), it indicates the number of bits sent and the maximum number of bits received (with How much the body receives is related to RX_INVALID_BIT) When (TxChOn=1, RxChOn=0), Indicates the number of bits sent, When (TxChOn=0, RxChOn=1), Indicates the maximum number of received bits (the specific number of received bits is related to RX_INVALID_BIT, the actual received number is Tx/Rx	16'h0

		length - RX_INVALID_BITÿ When (TxChOn=0, RxChOn=0), meaningless. Note: master is valid Motorola/TI/microwire mode available	
[2]	RW	Chip selects 1'b0: SPI_CS valid signal is 0 1'b1: SPI_CS valid signal is 1 Note: master is valid Motorola/TI/microwire mode valid	1'b0
[1]	RW	Force CS out 1'b0: spi_cs signal output is controlled by hardware 1'b1: spi_cs signal output is controlled by software, the specific output value is Chip selects This signal can be combined with Chip selects to realize the programmable output csn signal, that is, when the signal is 1, spi_cs = Chip selects Note: master is valid Motorola/TI/microwire mode valid	1'b0
[0]	RW	SPI start, Command SPI to start receiving or sending, write 1 for spi to start working, after that, it will automatically reset to zero 1'b0: Stop spi from working 1'b1: Start a sending or receiving of spi, automatically reset to zero Note: master is valid	1'b0

		Motorola/TI/microwire mode valid	

14.4.3 SPI Configuration Registers

Table 111 SPI configuration register

bit access		Instructions	reset value
[31:19]		RSV	13'h0
[18:17] RW		FRONT FORMAT 2'b00: motorola 2'b01:TI 2'b10:microwire 2'b11:RSV Select the master to support the protocol of that manufacturer Note: master is valid	2'b0
[16]	RW	SPI_TX pin always driven 1'b0: The spi output is only driven when spi_cs is valid, and is tri-state at other times 1'b1: spi output is always driven, even when no data is being transferred Note: master/slave are both valid Motorola/TI/microwire mode valid	1'b0
[15]		RSV	1'b0
[14:12] RW		cs hold, the time that spi_cs continues to be valid after the data transmission is completed, that is, the hold time of spi_cs 3'b000 >=1 APB bus CLK 3'b001 >=2 APB bus CLK 3'b010 >=4 APB bus CLK	3'b0

		3'b011 >=8 APB bus CLK 3'b100 >=16 APB bus CLK 3'b101 >=32 APB bus CLK 3'b110 >=64 APB bus CLK 3'b111 >=127 APB bus CLK Note: master is valid Motorola mode works	
[11:9] RW		cs setup, spi_cs is valid ahead of time before data transmission, that is, the setup time of spi_cs 3'b000 >=1 APB bus CLK 3'b001 >=2 APB bus CLK 3'b010 >=4 APB bus CLK 3'b011 >=8 APB bus CLK 3'b100 >=16 APB bus CLK 3'b101 >=32 APB bus CLK 3'b110 >=64 APB bus CLK 3'b111 >=127 APB bus CLK Note: master is valid Motorola mode works	3'b0
[8:7] RW		SPI-out delay, SPI data output relative to SCK delay, mainly for the consideration of hold time. [8:7] System clock cycle number (APB clock) 2'b00 0 2'b01 1	2'b0

		2'b10 2 2'b11 3 Note: master/slave are both valid Motorola mode works	
[6:4] RW		Frame delay, the default interval between the end of a frame (spi_cs valid period) transmission and the start of the next frame is SCK Half of the clock period, that is, the SPI_CS inactive time. But for compatibility, it is configurable here. Default at least 0.5SCK [6:4] SCK clock 3'b000 0 3'b001 2 3'b010 4 3'b011 8 3'b100 16 3'b101 32 3'b110 64 3'b111 127 For example, if 128byte data is transmitted in block mode, after the data transmission is completed, the set delay will be added. Late time. Note: master is valid	3'b0
[3]	RW	Bigendian 1'b0: The data format adopts the little-endian mode, that is, the low byte is sent first during transmission 1'b1: The data format adopts big-endian mode, that is, the high byte is sent first during transmission	1'b0
[2]	RW	MASTER/SLAVE	1'b1

		1'b0: slave, the device is slave 1'b1: master, the device is the master Note: master/slave are both valid	
[1]	RW	SPI CPHA 1'b0: Transmission mode A 1'b1: Transmission mode B Note: master/slave are both valid Motorola mode works	1'b0
[0]	RW	SPI CPOL, polarity of SCK at IDLE 1'b0: 0 when SCK is IDLE 1'b1: 1 when SCK is IDLE Note: master/slave are both valid Motorola mode works	1'b0

14.4.4 Clock configuration register

Table 112 SPI clock configuration register

bit access		Instructions	reset value
[31:16]		RSV	16'h0
[15:0]	RW	Divider FSCK = FAPB_CLK/ (2 x (Divider +1)) Note: master is valid Motorola/TI/microwire mode valid	16'h0
[31:16]		RSV	16'h0

[15:0] RW		Divider FSCK = FAPB_CLK/ (2 x (Divider +1)) Note: master is valid Motorola/TI/microwire mode valid	16'h0

14.4.5 Mode configuration register

Table 113 SPI mode configuration register

bit access		Instructions	reset value
[31:9]		RSV	23'h0
[8:6] RW		RxTrigger level The data stored in RX FIFO triggers the threshold of interrupt or DMA request: 0~7word Only when the data in rxbuffer is greater than the RxTrigger level, will an interrupt be triggered or a DMA move requested Note: master/slave are both valid Motorola/TI/microwire mode valid	3'b0
[5]		RSV	1'b0
[4:2] RW		TxTrigger level The data stored in TX FIFO triggers the threshold of interrupt or DMA request: 0~7word Only when the data in the txbuffer is greater than or equal to the TxTrigger level, will the interrupt be triggered or the DMA move will be requested. shift Note: master/slave are both valid Motorola/TI/microwire mode valid	3'b0
[1]	RW	RxDMA On, using DMA to move data enable	1'b0

		1'b0: do not use DMA, 1'b1: Using DMA Note: master/slave are both valid Motorola/TI/microwire mode valid	
[0]	RW	TxDMA On, using DMA to move data enable 1'b0: do not use DMA, 1'b1: Using DMA Note: master/slave are both valid Motorola/TI/microwire mode valid	1'b0

14.4.6 Interrupt Control Register

Table 114 SPI interrupt control register

bit access		Instructions	reset value
[31:8]		RSV	24'h0
[7]	RW	IntEn_spi_timeout 1'b0: enable spi_timeout interrupt 1'b1: spi_timeout interrupt not allowed Note: master/slave are both valid Motorola/TI/microwire mode valid	1'b1
[6]	RW	IntEn_spi_done 1'b0: spi send or receive complete, enable interrupt 1'b1: spi sending or receiving is complete, interrupts are not allowed	1'b1

		Note: both master/slave are valid Motorola/TI/microwire mode is valid	
[5]	RW	IntEnRxOverrun 1'b0: Rx FIFO overflow interrupt enable 1'b1: Rx FIFO overflow interrupt disabled Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b1
[4]	RW	IntEnRxUnderrun 1'b0: Rx FIFO underflow interrupt disabled 1'b1: Rx FIFO underflow interrupt enable Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b1
[3]	RW	IntEnTxOverrun 1'b0: Tx FIFO overflow interrupt enable 1'b1: Tx FIFO overflow interrupt disabled Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b1
[2]	RW	IntEnTxUnderrun 1'b0: Tx FIFO underflow interrupt enable 1'b1: Tx FIFO underflow interrupt disabled Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b1
[1]	RW	IntEnRxFifoRdy 1'b0: Rx FIFO has data upload interrupt enable 1'b1: Rx FIFO has data upload interrupt disabled Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b1
[0]	RW	IntEnTxFifoRdy	1'b1

		1'b0: Tx FIFO can write data to TX FIFO interrupt enable 1'b1: Tx FIFO can write data to TX FIFO interrupt is not enabled Note: both master/slave are valid Motorola/TI/microwire mode is valid	

14.4.7 Interrupt Status Register

Table 115 SPI Interrupt Status Register

bit access		Instructions	reset value
[31:8]		RSV	24'h0
[7]	RW	spi_timeout 1'b0: There is no end data in rxfifo to be fetched by CPU 1'b1: There is end data in rxfifo that needs to be fetched by the CPU Write 1 to clear Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0
[6]	RW	spi_done 1'b0: SPI send or receive is not completed 1'b1: SPI send or receive completed Write 1 to clear Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0
[5]	RW	RxOverrun 1'b0ýRx FIFO overflow 1'b1ýRx FIFO overflow Write 1 to clear	1'b0

		Note: both master/slave are valid Motorola/TI/microwire mode is valid	
[4]	RW	RxUnderrun 1'b0ÿRx FIFO underflow 1'b1ÿRx FIFO underflow Write 1 to clear Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0
[3]	RW	TxOverrun 1'b0ÿTx FIFO overflow 1'b1ÿTx FIFO overflow Write 1 to clear Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0
[2]	RW	TxUnderrun 1'b0ÿTx FIFO underflow 1'b1ÿTx FIFO underflow Write 1 to clear With continue mode = 1, this interrupt is never generated. Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0
[1]	RW	RxFifoRdy 1'b0: Rx FIFO data volume <= RxTrigger level, no need to upload 1'b1: Rx FIFO data volume > RxTrigger level, request to upload Write 1 to clear Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0

[0]	RW	TxFifoRdy 1'b0: Tx FIFO data volume > TxTrigger level, cannot write data to TX FIFO 1'b1: Tx FIFO data volume <= TxTrigger level, can write data to TX FIFO Write 1 to clear Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0

14.4.8 SPI Status Register

Table 116 SPI Status Register

bit access		Instructions	reset value
[31:13]		RSV	19'h0
[12]	RO	SPI Busy 1'b0: SPI has no send and receive tasks 1'b1: SPI is in the process of sending or receiving Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0
[11:6] RO		RX FIFO fill level The amount of data in the Rx FIFO, in bytes Note: both master/slave are valid Motorola/TI/microwire mode is valid	6'h0
[5:0]	RO	Tx FIFO fill level The amount of data in the Tx FIFO, in bytes Note: both master/slave are valid Motorola/TI/microwire mode is valid	6'h0

14.4.9 SPI Timeout Register

Table 117 SPI Timeout Register

bit access		Instructions	reset value
[31]	RW	sleep_timer_en 1'b0: Timer is not allowed 1'b1: Allow timer timing Note: both master/slave are valid Motorola/TI/microwire mode is valid	1'b0
[30:0] RW		SPI_TIME_OUT When a transmission is completed, in the receiving path rxfifo, if the end data cannot trigger the receiving interrupt When RxFifoRdy or DMA requests, a timing mechanism needs to be used to notify the CPU to remove the end data. Specific method: when rxfifo is in idle state (no read and write operations, no dma request, cs is invalid, There is data in rxfifo, and the amount of data is less than or equal to RxTrigger level), start counting, and reach the register setting If the set value is set, a timeout interrupt is triggered and the CPU is requested to remove the end data. Any read or write operation to rxfifo will clear the timeout timer. The time represented is: T = SPI_TIME_OUT/FAPB_CLK Note: both master/slave are valid Motorola/TI/microwire mode is valid	31'h0

14.4.10 Data Transmit Register

Table 118 SPI data transmission register

bit access		Instructions	reset value

[31:0] RW		Window address for writing data to Tx FIFO Note: Both master/slave are valid Motorola/TI/microwire mode valid	32'h0

14.4.11 Transfer Mode Register

Table 119 SPI transfer mode register

bit access		Instructions	reset value
[31:30]		RSV	16'd0
[29:24] RW		TI_BLK_LEN In the timing mode of TI, the length of each block transmission is the transmission data length after each CS is valid. Support 4~32bit 6'h4: 4bit long data 6'h5: 5bit long data 6'h6: 6bit long data …….. 6'h20: 32bit long data Note: master is valid TI mode is valid	6'd0
[16]	RW	MICRO_BURST 1b'1: In Microwire mode, burst transmission is used, that is, Tx sends the control word, Rx receives data, in turn Alternately, MICRO_CONTROL_LEN represents the length of the control word, and MICRO_DAT_LEN represents	1'b0

		It is the length of the sent or received word, Tx/Rx length represents the effective sck, burst during the entire transmission process In this mode, the number of alternate sending and receiving is (Tx/Rx length)/(MICRO_CONTROL_LEN+ MICRO_DAT_LEN+1) 1'b0: In Microwire mode, do not use burst transmission In this mode, there are two cases 1) tx_ch_on = 1, rx_ch_on = 0, at this time, only send, MICRO_CONTROL_LEN means Control word length, Tx/Rx length indicates the effective sck during the entire transmission process, and the length of the data sent at this time The degree is m*MICRO_DAT_LEN = Tx/Rx length - MICRO_CONTROL_LEN, where m represents How many (MICRO_DAT_LEN) length words to send 2) tx_ch_on = 1, rx_ch_on = 1, at this time, Tx sends the control word, Rx receives data, MICRO_CONTROL_LEN represents the control word length, Tx/Rx length represents the entire transmission process Valid sck during the process, the received data length is m*MICRO_DAT_LEN = Tx/Rx length MICRO_CONTROL_LEN-1, where m indicates how many (MICRO_DAT_LEN) length words are received Note: master is valid microwire mode is valid	
[13:8] RW		MICRO_DAT_LEN In Microwire mode, in burst mode, the length of each burst transmission data From 1~32: 6'h1: 1bit long data 6'h2: 2bit long data	6'd0

		6'h3: 3bit long data …….. 6'h20 32bit long data Note: master is valid microwire mode is valid	
[5:0] RW		MICRO_CONTROL_LEN In Microwire mode, the length of the command word From 1~32: 6'h1: 1bit long command 6'h2: 2bit long command 6'h3: 3bit long command …….. 6'h20 32bit long command Note: master is valid microwire mode is valid	6'd0

14.4.12 Data Length Register

Table 120 SPI data length register

bit access		Instructions	reset value
[31:16] RO		When working as a slave, the length of the data sent out during the effective period of cs, the unit is bit Note: slave is valid Motorola mode works	16'h0

[15:0] RO		When working as a slave, the received data length during the effective period of cs, the unit is bit Note: slave is valid Motorola mode works	16'h0

14.4.13 Data Receive Register

Table 121 SPI Data Receive Register

bit access		Instructions	reset value
[31:0] RO		Window address for reading data from Rx FIFO Note: master/slave are both valid Motorola/TI/microwire mode valid	32'h0

15 I2C controller

15.1 Function overview

The I2C bus is a simple, bidirectional, two-wire synchronous serial bus. It requires only two wires to transmit information between devices connected to the bus

breath.

The master device is used to start the bus to transmit data and generate the clock to open the device for transmission. At this time, any addressed device is considered a slave

pieces. The relationship between master and slave, send and receive on the bus is not constant, but depends on the direction of data transmission at this time. If the master wants to send data to the slave

device, the host first addresses the slave device, then actively sends data to the slave device, and finally the host terminates the data transmission; if the host wants to receive

For the data from the device, the slave device is first addressed by the master device. Then the host receives the data sent from the device, and finally the host terminates the receiving process.

in this case. The host is responsible for generating the timing clock and terminating data transfers.

15.2 Main Features

- ÿ APB bus protocol standard interface
- ÿ Can only be used as a master device controller
- ÿ I2C working rate can be configured, 100KHz~400KHz
- ÿ Multiple GPIOs can be multiplexed as I2C communication interface
- ÿ Fast output and detection of timing signals

15.3 Functional description

15.3.1 Transmission rate selection

By setting the register PRERlo and register PRERhi, the data transmission rate on the I2C bus can be configured from 100KHz to

Integer divider value for any bus frequency between 400KHz.

15.3.2 Interruption and start-stop controllable

Enable or disable the I2C controller to generate interrupts by setting Bit6 of the register CTR, and can also be started at any time by setting Bit7

Or stop the work of the I2C controller.

15.3.3 Fast output and detection signal

By setting the corresponding bit of the register CR_SR, the controller can quickly output or detect the bus START signal, bus STOP signal, total

Line ACK signal, bus NACK signal. In master mode, the I2C interface initiates data transfers and generates clock signals. a serial data transfer

Output always begins with a start signal and ends with a stop signal. Once a START signal is generated on the bus, master mode is selected.

15.4 Register Description

15.4.1 Register List

Table 122 I2C register list

offset address name		abbreviation	access	describe	reset value
0X0000	Clock Divider Register_1	PRESERVE IT	RW Stores the lower 8-bit frequency division value for APB	The bus clock is divided	0X0000_00FF
0X0004	Clock Divider Register_2	PRERhi	RW stores the high 8-bit frequency division value for APB	The bus clock is divided	0X0000_00FF
0X0008 Control register		CTR	RW is used to control interrupt enable and I2S control	device enable	0X0000_0040
0X000C data register		TXR_RXR	RW is used to store the data to be sent or receive 0X0000_0000		

				The data	
0X0010 Transceiver control register		CR_SR	RW is used to control some data read and write related operations		0X0000_0000
0X0014	TXR readout register	TXR	RO reads the TXR register value	0X0000_0000 when I2C sends	
0X0018	CR read register	CR	RO Read the setting value of I2C control register	CR 0X0000_0000	

15.4.2 Clock frequency division register_1

Table 123 I2C clock frequency division register_1

bit access		Instructions	reset value
[31: 8]		reserve	
[7 : 0] RW		Clock frequency division configures the lower 8 bits of prescale. E.g: apb_clk=40MHz, SCL=100KHz prescale = (40*1000)/(5*100) – 1 = 16'd79 apb_clk = 40M, SCL=400K prescale=(40*1000)/(5*400) – 1 = 16'd19	8'hff

15.4.3 Clock frequency division register_2

Table 124 I2C clock frequency division register_2

bit access		Instructions	reset value
[31: 8]		reserve	
[7 : 0] RW		Clock frequency division configures the upper 8 bits of prescale. E.g:	8'hff

		apb_clk=40MHz, SCL=100KHz prescale = (40*1000)/(5*100) – 1 = 16'd79 apb_clk = 40M, SCL=400K prescale=(40*1000)/(5*400) – 1 = 16'd19	

15.4.4 Control Registers

Table 125 I2C Control Register

bit access		Instructions	reset value
[31:8]		reserve	
[7]	RW	I2C enable control, 1'b0: disable 1'b1: enable	1'b0
[6]	RW	Interrupt MASK, 1'b0: enable interrupt generation 1'b1: Interrupts are not allowed	1'b1
[5:0]		reserve	

15.4.5 Data Registers

Table 126 I2C Data Register

bit access		Instructions	reset value
[31:8]		reserve	
[7:0] WR		When writing this register, it is the transmit register TXR,	8'h0

		Indicates the next byte to be sent, When it is a device address, [0]: 1 means read, 0 means write. When reading this register, it is the receive register RXR, is the most recent byte received from I2C.	

15.4.6 Transceiver Control Register

Table 127 I2C transceiver control register

bit access		Instructions	reset value
[31:8]		reserve	
[7:0] WR		When writing this register, it is CR, and its function is as follows: [7]: STA, control to generate START sequence; 1'b0: invalid 1'b1: Generate START timing [6]: STO, control to generate STOP sequence; 1'b0: invalid 1'b1: Generate STOP timing [5]: RD, read from SLAVE; 1'b0: invalid	8'h0

		1'b1: read from SLAVE [4]: WR, write to SLAVE; 1'b0: invalid 1'b1: Write to SLAVE [3]: Control sends ACK/NACK to SLAVE; 1'b0: times ACK 1'b1ÿÿ WANT [2:1]: reserved; [0]: IACK, clear interrupt status, 1 is valid; 1'b0: invalid 1'b1: Clear interrupt flag When reading this register, it is SR, and its function is as follows: [7]: RxACK, ACK/NACK status received from SLAVE; 1'b0: ACK received from SLAVE 1'b1: NAK received from SLAVE [6]ÿBUSYÿ 1'b0: STO followed by 0	

		1'b1: STA is set to 1 [5]: AL, Arbitration Lost, this bit is reserved; [4:2]: reserved; [1]ÿTIPÿ 1'b0: No transfer in progress 1'b1: There is a transfer in progress [0]: IF, interrupt status bit; 1'b0: No interrupt generated 1'b1: Set to 1 when transfer complete or AL	

15.4.7 TXR Readout Register

Table 128 I2C TXR readout register

bit access		Instructions	reset value
[31:8]		reserve	
[7:0]	RO	Read only, the read value of the TXR register, See the TXR_RXR register for the function description;	8'h0

15.4.8 CR Read Register

Table 129 I2C CR readout register

bit access		Instructions	reset value

[31:8]		reserve	
[7:0]	RO	Read only, read value of CR register, See the CR_SR register for the function description;	8'h0

16 I2S controller

16.1 Function overview

I2S (Inter-IC Sound) is for digital audio devices (such as CD players, digital sound processors, digital TV audio systems)

A bus standard developed for the transmission of audio data between It adopts the design of independent wire transmission clock and data signal, by connecting the data

The separation of data and clock signals avoids the distortion caused by time difference, and saves the cost of purchasing professional equipment to resist audio jitter for users. mark

The standard I2S bus cable is composed of 3 serial wires: 1 is a time division multiplexing (referred to as TDM) data line; 1 is a word selection

wires; 1 is a clock wire.

16.2 Main Features

- ÿ Implement I2S interface, support I2S and PCM protocols
- ÿ Support amba APB bus interface, 32bit single read and write operations
- ÿ Support master-slave mode
- ÿ Support 8, 16, 24, 32 bit width, the highest sampling frequency is 192KHz
- ÿ Support mono and stereo mode
- ÿ Compatible with I2S and MSB justified data format, compatible with PCM A/B format
- ÿ Support DMA request read and write operations, only support word-by-word operations

16.3 Functional description

16.3.1 Multiple Mode Support

By setting Bit[25:24] of the I2S Control register, the data format can be set to I2S format, MSB justified format, PCM A

format, or PCM B format; the mono or stereo mode can be selected by setting Bit[22] of the I2S Control register. pass

Setting Bit[5:4] of the I2S Control register can set the bit width of the data transmission word, which can be set to 8bit, 16bit, 24bit, or 32bit.

16.3.2 Zero Crossing Detection

By setting Bit[17:16] of the I2S Control register, you can set whether to enable the zero-cross detection function of the left and right channels; by setting

Bit[9:8] of the I2S_IMASK register can set whether the zero-cross detection function of the left and right channels generates an interrupt. If detection is enabled, and

And the interrupt is enabled, when the zero-cross phenomenon is detected, the program will execute the interrupt subroutine, and at the same time, Bit[9:8] of the I2S_INT_FLAG register

The corresponding bit will be set to 1.

16.3.3 Efficient Data Transfer

The FIFO memory is a first-in-first-out dual-port buffer, that is, the first data that enters it is the first to be moved out, and one of the memory

One is the input port, and the other is the output port of the memory. The I2S controller integrates two FIFO storages with a depth of 8 words each (one for each transceiver)

In order to increase the data transmission rate, handle a large number of data streams, and match systems with different transmission rates, the system performance is improved. able to pass

Setting Bit[14:12] and Bit[11:9] of the I2S Control register can set the trigger level of RXFIFO and TXFIFO to meet

Performance requirements at different transfer rates. After the trigger level of the FIFO is triggered, it can trigger an interrupt or DMA to transfer the data from the internal

Store and move to TXFIFO or move data from RXFIFO to memory.

16.4 I2S/PCM Timing Diagram

This module provides support for 4 protocols, standard I2S, MSB Justified, PCM-A, PCM-B. Through configuration register 0x00[25:

24] to choose which protocol to use. The specific interface timing of each protocol is shown in Fig1 to Fig4.

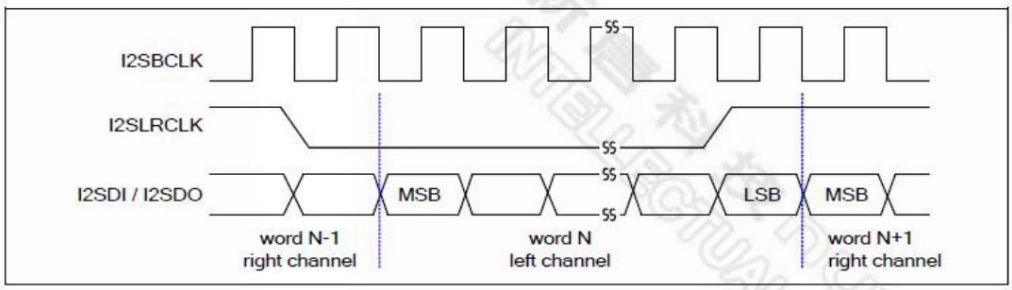


Fig1. I2S Bus Timing DiagramÿPCM=0ÿFormat=0ÿ

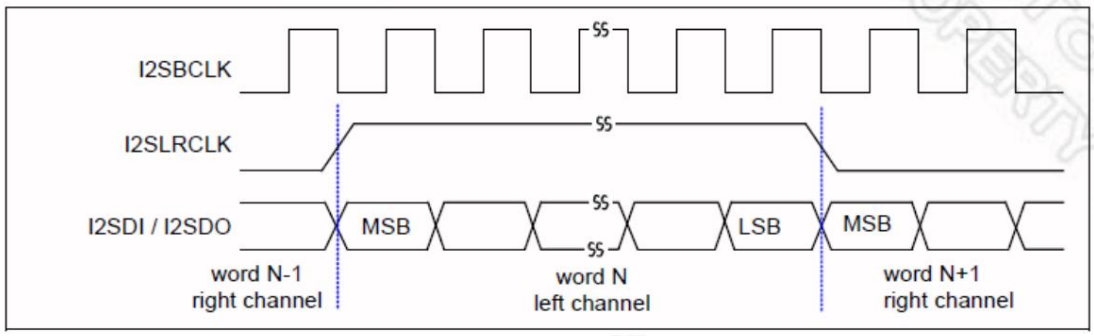


Fig2. MSB Justified Timing DiagramÿPCM=0ÿFormat=1ÿ

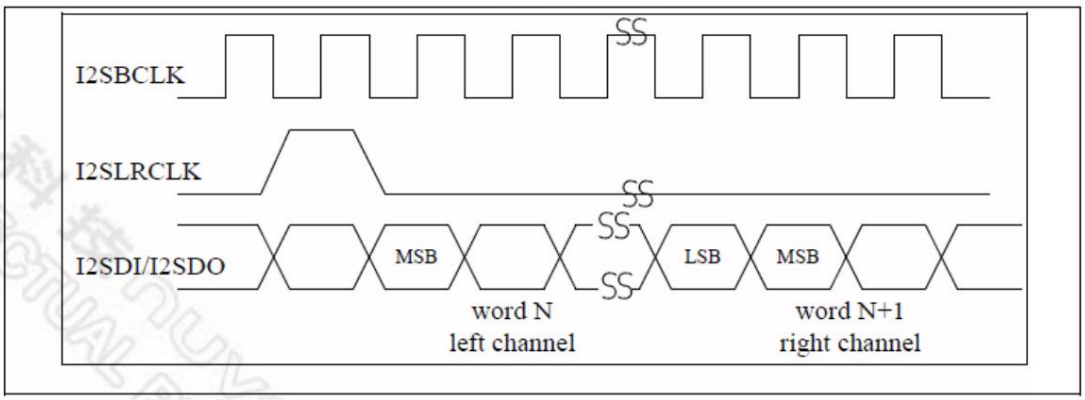


Fig3. PCM A Audio DiagramÿPCM=1ÿFormat=0ÿ

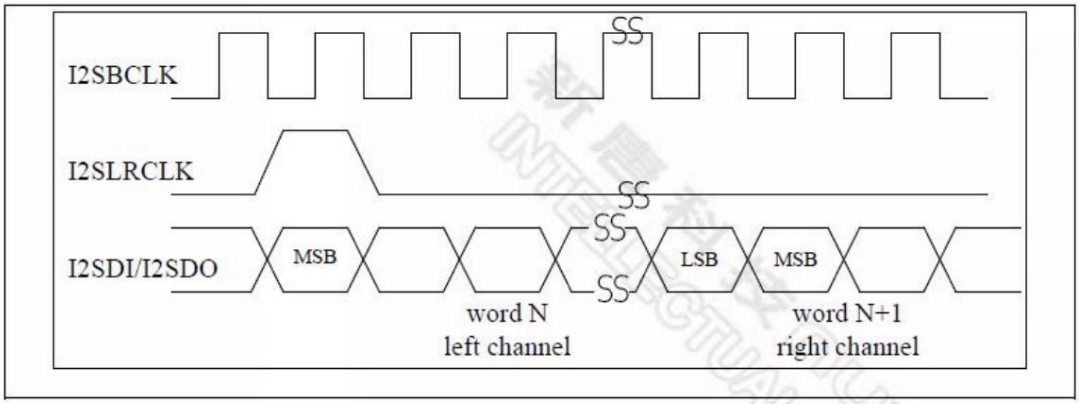


Fig4. PCM B Audio DiagramÿPCM=1ÿFormat=1ÿ

16.5 FIFO storage structure diagram

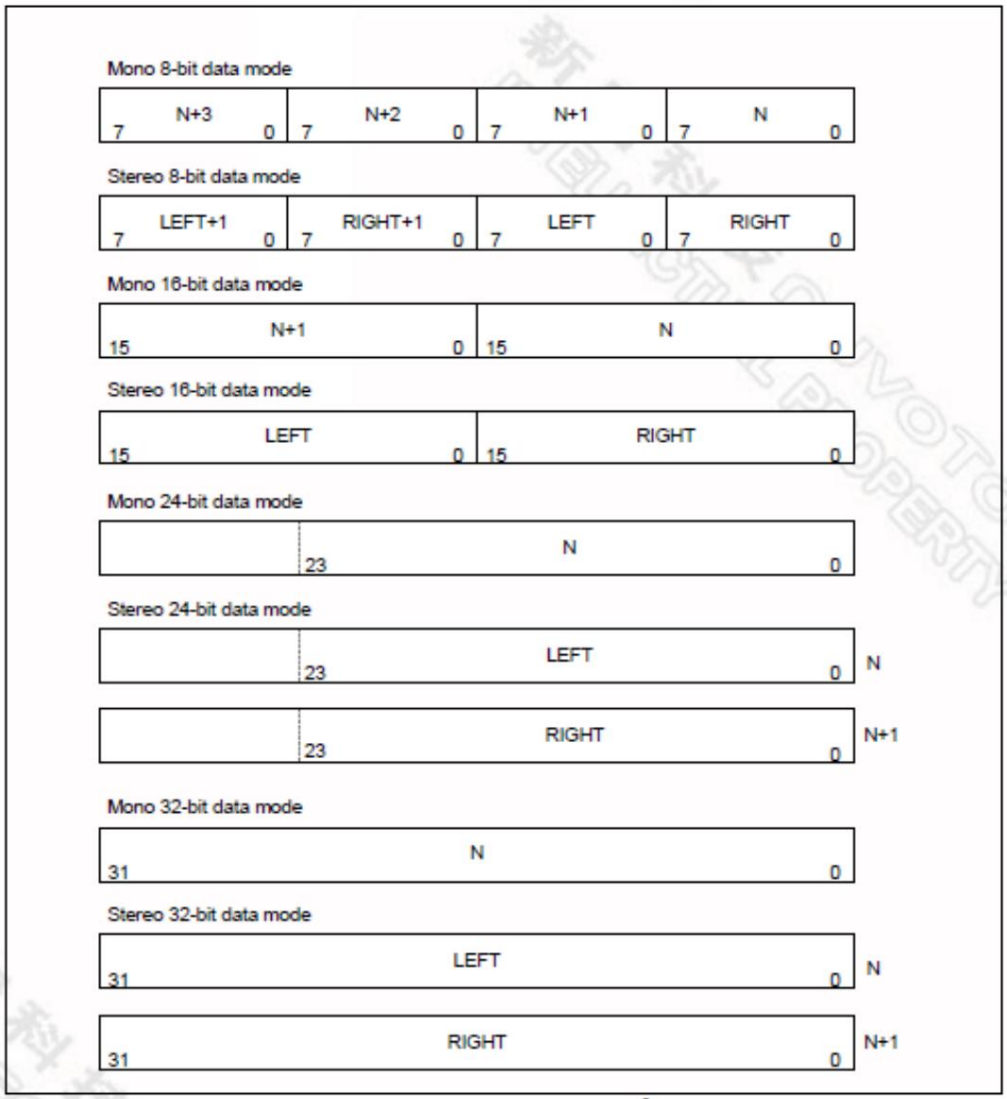


Fig 5. FIFO storage structure

The I2S module provides two 8x32bit FIFOs, one as TX FIFO and one as RX FIFO.

The storage structure of data in FIFO is different according to different working modes. When working in mono 8bit mode, each word in FIFO

Can store 4 sample data. When working in dual-channel 8bit mode, left and right channel data are alternately stored in FIFO, at low

The bit byte stores the right channel data, and the high bit byte stores the left channel data. A word can store 2 sample numbers

according to. In simplex 16bit mode, one word can store 2 samples; in duplex 16bit mode, one word can store one

For sample data, the lower 16 bits are the right channel, and the upper 16 bits are the left channel. In 24bit and 32bit mode, each word can only store one

The sample data or the data of a channel, the specific storage method is shown in Figure 5.

16.6 I2S module working clock configuration

The working clock configuration of the I2S module can be configured by setting the 0x40000718 register in the clock and reset module to select an external clock source

The clock is still the internal 160MHz clock, whether to enable the MCLK clock, and the operating frequency of MCLK and BCLK.

By setting the register 0x40000718[0], you can choose to use the internal 160MHz clock or use the external clock as the I2S module clock

source. If an external clock is selected, the I2S module will use the clock input from IO- I2S_M_EXTCLK (PA_5, Option 4) as the module time

Zhong Yuan.

And 0x40000718[1] selects whether to enable MCLK clock. [7:2] bits are the area to configure the MCLK divider ratio. Calculated as follows:

F_mclk=F_I2SCLK/MCLKDIV

F_mclk is the actual MCLK frequency;

F_I2SCLK is the clock source of the I2S module. If the internal clock is selected, then F_I2SCLK =160MHz; if the external clock is selected, then

F_I2SCLK is equal to the clock frequency of external input;

MCLKDIV is the clock frequency division ratio configured by register 0x40000718[7:2]. It should be noted that MCLKDIV>=2;

Bits [17:8] of register 0x40000718 are the area for configuring the division ratio of BLCK clock. The frequency division ratio calculation formula is as follows:

F_BCLK=F_I2SCLK/BCLKDIV

F_BCLK is the actual working frequency of BCLK;

F_I2SCLK is the clock source of the I2S module. If the internal clock is selected, then F_I2SCLK =160MHz; if the external clock is selected, then

F_I2SCLK is equal to the clock frequency of external input;

BCLKDIV is the frequency division ratio that needs to be configured. Different frequency division ratios need to be selected according to different working modes. The formula for selecting the divider ratio is as follows:

BCLKDIV=round (F_I2SCLK/(Fs*W*F))

Fs is the sampling frequency of audio data on the I2S interface, up to 192KHz;

W is the sampling bit width, 8/16/24/32 bit can be selected;

F is mono/dual channel selection. When the transmission data is monophonic, F=1, and when the transmission data is dual-channel, F=2;

The final calculated frequency division ratio is rounded up.

The following is an example of selecting BCLKDIV according to the actual working mode:

If the internal clock is selected as the module clock source, 128KHz sampling rate and 24bit two-channel data need to be transmitted, then BCLKDIV needs to be set for calculation

The process is as follows:

BCLKDIV=round(160*10e6/128*10e3*24*2)=10'd26ÿ

The frequency division register is described as follows:

0x18	I2S_Clk_Ctrl	[0]	RW	EXTAL_EN External clock select Select whether use External or Internal clock for I2S block 0=internal clk 1=external clk Note: When External clock is enabled, the external clk must be 2*N*256 fs, where fs is sample frequency and N must be integer.	1'b0

		[1]	RW	MCLKEN MCLK enable 0=MCLK disabled 1=MCLK enabled	1'b0
		[7:2]	RW	MCLKDIV MCLK divider If external clock is selected, this divider is used to produce proper MCLK frequency. F_mclk=F_I2SCLK/MCLKDIV Where MCLKDIV >=2 F_mclk=F_I2SCLK when MCLKDIV=0 Where F_I2SCLK is external clk. Note: F_mclk should be configured as 256 * fs where fs is sample frequency.	6'd0
		[17:8]	RW	BCLKDIV BCLK divider F_BCLK=F_I2SCLK/BCLKDIV Note: When EXTAL_EN is not selected, Internal PLL is used and F_I2SCLK =160MHz	10'd0

				When WXTAL_EN is enabled, F_I2SCLK = External crystal frequency **BCLKDIV=round (F_I2SCLK/(Fs*W*F))** Where Fs is sample frequency of audio data and W is word width. F=2 when data is stereo and F=1 when data is mono. For example, if internal PLL is used and the data width is 24bit, Format is stereo format, sample frequency is 128KHz. Then the BCLKDIV should be configured as (160*10e6/128*10e3*24*2) =10'd26	
		[31:18] RO		RSV	14'd0

16.7 Other function instructions:

16.7.1 Zero Crossing Detection:

In order to avoid noise caused by sudden frequency changes caused by data corruption, the I2S module provides a zero-crossing detection function for each channel.

When the two adjacent data sign bits sent change, the module will generate an interrupt to remind the MCU to detect and process; at the same time, the latter

Data is forcibly muted.

16.7.2 Mute function

When the mute function is turned on, the data will still be sent, but the output data will be forced to be 0;

16.7.3 Interrupts

This module provides send/receive completion interrupt, left and right channel zero-crossing detection interrupt, send/receive FIFO threshold interrupt (number in send FIFO

The data is lower than the threshold, the data in the receiving FIFO is higher than the threshold), the underflow/overflow of the sending/receiving FIFO is interrupted. The interrupt state is at

Query in register 0x08.

16.7.4 FIFO Status Query

Register 0x10 provides CPU status query function for sending/receiving FIFO in I2S module. CPU can check FIFO by register

How much data is left unprocessed in . Several bytes in the last word in the receive FIFO are valid data.

16.8 Data transmission process

16.8.1 Master sends audio data

1. Configure the used pins, SDO, BCLK, LRCLK

2. Refer to Section 2.4 to set the register 0x40000718 in the clock and reset module, and configure the working clock frequency;

3. Set the I2S register 0x00, set it to master mode, configure the transmission format, channel selection, data bit width, left and right channels are

   No Enable zero-crossing detection, and set the transmit path -txen to open.

4. Set register 0x4 to enable the desired interrupt;

5. If DMA is used to transfer data, select the channel used in the DMA module, and configure the address and length of the transmitted data. and in I2S

   DMA is enabled in module register 0x0, and the FIFO threshold is set. When the data in FIFO is lower than the threshold range, it will automatically request DMA

   Move data.

6. Write the data to be transmitted to the transmit FIFO address-register 0x10, enable the register 0x0[0], and the module will automatically read from the FIFO

   Take out the data and send it to the I2S bus.

7. When the data in the FIFO is less than the set threshold, the module will request data from the DMA module, or send a TXTHIF interrupt.

8. When all the data in the FIFO is taken out, the TXDONE interrupt will be set. When the last frame is sent, the TXUDIF interrupt will be set.

   When the CPU is notified that the sending is complete, the module stops sending.

16.8.2 Receiving Audio Data from the Slave

1. Configure the used pins, SDI, BCLK, LRCLK

2. Set the I2S register 0x00, set it to slave mode to configure the transmission format, channel selection, data bit width, whether the left and right channels are on

   Enable zero-crossing detection, and set the receive channel -rxen to open.

3. Set register 0x4 to enable the interrupt you want to use;

4. If DMA is used to transfer data, select the channel used in the DMA module, and configure the address and length of the transmitted data. and in I2S

DMA is enabled in module register 0x0, and the FIFO threshold is set. When the data is higher than the threshold range, it will automatically request DMA transfer data.

5. Enable register 0x0[0], after the module detects valid BCLK and LRCLK, it will automatically collect data from SDI and store it in FIFO middle. When the data in the FIFO is higher than the set threshold, the module will request the DMA to move the data to the memory, or send the RXTHIF broken. RXDONE interrupt is generated when CPU turns off RXEN. The CPU can query how many bits are in the receive FIFO through register 0x10 data, how many bytes of valid data are there in the last word. The last remaining data is then processed according to the FIFO status.

16.8.3 Master Receives Audio Data

1. Configure the hardware used, SDI, SCLK, LRCLK

2. Refer to Section 2.4 to set the register 0x40000718 in the clock and reset module, and configure the working clock frequency;

3. Set the I2S register 0x00, set it as master mode, configure the transmission format, channel selection, data bit width, whether the left and right channels are enabled Enable zero-crossing detection, and set the receive channel rxen to open.

4. Set register 0x4 to enable the desired interrupt;

5. If DMA is used to transfer data, select the channel used in the DMA module, and configure the address and length of the transmitted data. and in I2S DMA is enabled in module register 0x0, and the FIFO threshold is set. When the data is higher than the threshold range, it will automatically request DMA transfer data.

6. Enable register 0x0[0], the module will automatically send BCLK and LRCLK, and at the same time collect data from SDI and store it in FIFO. when When the data in the FIFO is higher than the set threshold, the module will request the DMA to transfer the data to the memory, or send an RXTHIF interrupt. when RXDONE interrupt is generated when CPU turns off RXEN. The CPU can query how many data are in the receiving FIFO through the register 0x10, How many bytes of valid data are there in the last word. The last remaining data is then processed according to the FIFO status.

7. Turn off the i2s enable after the data reception is completed.

16.8.4 Slave sends audio data

1. Configure the used pins, SDO, BCLK, LRCLK

2. Set the I2S register 0x00, set it to slave mode, configure the transmission format, channel selection, data bit width, whether the left and right channels
   Enable zero-crossing detection, and set the transmit channel txen to open.

3. Set register 0x4 to enable the desired interrupt;

4. If DMA is used to transfer data, select the channel used in the DMA module, and configure the address and length of the transmitted data. and in I2S
   DMA is enabled in module register 0x0, and the FIFO threshold is set. When the data in FIFO is lower than the threshold range, it will automatically request DMA
   Move data.

5. Write the data to be transmitted to the transmit FIFO address-register 0x10, enable the register 0x0[0], when the module detects a valid BCLK
   and LRCLK, the module will automatically fetch data from FIFO and send it to SDO.

6. When the data in the FIFO is less than the set threshold, the module will request data from the DMA module, or send a TXTHIF interrupt.

7. When all the data in the FIFO is taken out, the TXDONE interrupt will be set. When the last frame is sent, the TXUDIF interrupt will be set.
   When the CPU is notified that the sending is complete, the module stops sending.

16.8.5 Full Duplex Mode

1. Configure the used pins, SDI, SDO, BCLK, LRCLK

2. If it is master mode, you need to configure clock frequency division.

3. Set the I2S register 0x00, configure the working mode (master/slave), configure the transmission format, channel selection, data bit width, left and right channels are
   No Enable zero-crossing detection, and set the transmit channel txen and receive channel rxen to open.

4. Set register 0x4 to enable the desired interrupt;

5. If DMA is used to transfer data, select the channel used in the DMA module, and configure the address and length of the transmitted data. and in I2S
   DMA is enabled in module register 0x0, and the FIFO threshold is set. When the data in FIFO is lower than the threshold range, it will automatically request DMA
   Move data.

6. Write the data to be transmitted to the transmit FIFO address-register 0x10

7. If it is master mode, enable the register 0x0[0], the module will start sending BCLK and LRCLK, and take out the data from the sending FIFO

The data starts to be sent out from the SDO port, and the data received from the SDI is stored in the receive FIFO.

8. If it is in slave mode, when the module detects valid BCLK and LRCLK, the module will automatically fetch data from the sending FIFO

Sent to SDO, and at the same time receive data from SDI and store in receive FIFO.

9. When the data in the transmit FIFO is less than the set threshold, the module will request data from the DMA module, or send a TXTHIF interrupt.

10. When the data in the FIFO is higher than the set threshold, the module will request the DMA to transfer the data to the memory, or send an RXTHIF interrupt.

11. When all the data in the TXFIFO is taken out, the TXDONE interrupt will be set. When the last frame is sent, the TXUDIF interrupt will be set.

Notify the CPU that sending is complete, and the module stops sending.

12. RXDONE interrupt is generated when CPU turns off RXEN. The CPU can query how many numbers are in the receiving FIFO through the register 0x10

Data, how many bytes of valid data are there in the last word. The last remaining data is then processed according to the FIFO status.

16.9 Register Description

16.9.1 Register List

Table 130 I2S register list

offset address name		abbreviation	access	describe	reset value
0X0000 Control Register		I2S Control	RW/RO controls I2S related functions, see the following chapters for details;		0X0000_4800
0X0004 interrupt mask register I2S_IMASK			RW control turns on or off all interrupts in I2S		0X0000_03FF
0X0008 Interrupt flag register I2S_INT_FLAG			RW/RO Interrupt flag bit, can be used to check whether an interrupt is generated and	Clear related interrupts	0X0000_0000
0X000c status register		I2S_STATUS	RO is used to query the relevant status of FIFO in the process of I2S communication	state	0X0000_0000

0X0010 Data transmission register I2S_TX			The WO controller will send the data in it to the bus 0X0000_0000		
0X0014 Data receiving register I2S_RX			The RO controller will receive the data on the bus into it 0X0000_0000		

16.9.2 Control Registers

Table 131 I2S Control Register

bit access		Instructions	reset value
[31:29] RO		RSV	7'h0
[28]	RW	mode selection 0 = master mode 1 = slave mode	1'b0
[27]	RW	Duplex mode selection 1 = Enable duplex mode 0 = disable duplex mode	1'b0
[26]	RW	Timeout count control bit, when this bit is set to 1 and the transmission process is forced to stop by the master device, no reception will occur Completion (RXDONE) interrupt	1'h0
[25:24] RW		FORMAT Data Format Selection 2'b00: I2S data format 2'b01: MSB Justified data format 2'b10: PCM A sound data format 2'b11: PCM B sound data format	2'b0
[23]	RW	RXLCH	1'b0

		Channel receive enable control bit 1'b0: Enable receiving right channel data 1'b1: Enable receiving left channel data Note: This bit is valid only when the mono mode of MONO_STEREO is selected	
[22]	RW	MONO_STEREO Mono stereo select bit 1'b0: Data is transmitted in stereo format 1'b1: Data is transmitted in mono format	1'b0
[21]	RW	RXDMAEN Receive DMA request enable bit 1'b0: Disable sending DMA requests 1'b1: Enable sending DMA requests Note: When the transfer DMA request is enabled and the number of words in the RXFIFO is equal to or greater than RXTH, The I2S controller will send a transfer request to the DMA until the RXFIFO is empty before stopping the DMA transfer.	1'b0
[20]	RW	TXDMAEN Send DMA request enable bit 1'b0: Disable sending DMA requests 1'b1: Enable sending DMA requests Note: When the transmit DMA request is enabled and the number of words in the TXFIFO is less than TXTH, the I2S controller It will send a transfer request to the DMA until the TXFIFO is full before stopping the DMA transfer.	1'b0
[19]	WHERE	RXCLR empty RXFIFO	1'b0

		1'b0: invalid 1'b1: Empty RXFIFO Note: Write 1 to clear the RXFIFO, which is automatically cleared by hardware. Reading this bit always returns 0	
[18] WHERE		TXCLR empty TXFIFO 1'b0: invalid 1'b1: Empty TXFIFO Note: Write 1 to clear the TXFIFO, which is automatically cleared by hardware. Reading this bit always returns 0	1'b0
[17]	RW	LZCEN Left channel zero cross detection enable control bit 1'b0: Stop left channel zero-crossing detection 1'b1: Enable left channel zero-crossing detection	1'b0
[16]	RW	RZCEN Right channel zero cross detection enable control bit 1'b0: Stop right channel zero-crossing detection 1'b1: Enable right channel zero-crossing detection	1'b0
[15]	RW	Rx_clk_phase_sel Receive clock phase selection 1'b0: default mode Shown with the I2S bus timing mentioned above 1'b1: reverse mode Shown in reverse form of the I2S bus timing mentioned above	1'b0

[14:12] RW		RXTH RXFIFO Threshold 3'b000: set the threshold to 0 words 3'b000: set the threshold to 1 word … 3'b111: set the threshold to 7 words Note: The RXTHIF bit will be set when the existing words in the RXFIFO are equal to or more than the value of RXTH. this You can choose to trigger RXDMA or I2S interrupt according to the setting	3'h4
[11:9] RW		TXTH TXFIFO Threshold 3'b000: set the threshold to 0 words 3'b000: set the threshold to 1 word … 3'b111: Set the threshold to 7 characters Note: The TXTHIF bit will be set when the word present in the TXFIFO is equal to or less than the value of TXTH. this You can choose to trigger TXDMA or I2S interrupt according to the setting	3'h4
[8]	RW	Tx_clk_phase_sel Select transmit clock phase mode 1'b0: default mode Shown with the I2S bus timing mentioned above 1'b1: reverse mode Shown in reverse form of the I2S bus timing mentioned above	1'b0

[7:6] RW		RSV	2'h0
[5:4] RW		WDWIDTH Transmission word length setting bit 2'b00: word length 8 bits 2'b01: word length 16 bits 2'b10: word length 24 bits 2'b11: word length 32 bits	2'b0
[3]	RW	MUTE Transmit mute enable flag bit 1'b0: Transfer data from shift register, normal operation mode 1'b1: Set the transmission data to 0 and mute the sound	1'b0
[2]	RW	RXEN Receive enable flag 1'b0: Stop I2S data reception 1'b1: Enable I2S data reception	1'b0
[1]	RW	CHEN transfer enable flag 1'b0: Stop I2S data transfer 1'b1: Enable I2S data transfer	1'b0
[0]	RW	I2SEN I2S enable flag bit 1'b0: disable	1'b0

		1'b1: enable	

16.9.3 Interrupt Mask Register

Table 132 I2S interrupt mask register

bit access		Instructions	reset value
[31:10] RO		RSV	22'h0
[9]	RW	LZCIMASK Left channel zero cross interrupt enable flag bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable When interrupts are enabled and a zero crossing is detected on the left channel, an interrupt is generated	1'b1
[8]	RW	RZCIMASK Right channel zero cross interrupt enable flag bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable When interrupts are enabled and a zero crossing is detected on the right channel, an interrupt is generated	1'b1
[7]	RW	TXDONEMASK Transmit complete interrupt enable bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable Interrupt is generated when the interrupt is enabled and the TXFIFO is empty	1'b1
[6]	RW	TXTHIMASK	1'b1

		TXFIFO threshold interrupt enable bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable When the interrupt is enabled and the number of data in the TXFIFO is equal to or less than TXTH, an interrupt will be generated	
[5]	RW	TXOVIMASK TXFIFO overflow interrupt enable bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable Note: When the interrupt is enabled, the TXFIFO is full, and the CPU writes data to the TXFIFO again, the TXOVIF flag will be will be set	1'b1
[4]	RW	THUDIMASK TXFIFO underflow interrupt enable bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable Note: When the TXFIFO underflow interrupt is enabled and TXUDIF is detected as 1, an underflow interrupt will be generated	1'b1
[3]	RW	RXDONEMASK Receive complete interrupt enable flag bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable When the receive complete interrupt is enabled and the receive process is complete, a receive complete interrupt will be generated	1'b1
[2]	RW	RXTHIMASK RXFIFO threshold interrupt enable flag bit	1'b1

		1'b0: Interrupts are not enabled 1'b1: interrupt enable When the RXFIFO threshold interrupt is enabled and the number of data in the RXFIFO is equal to or more than the threshold value, a Generate RX interrupt	
[1]	RW	RXOVIMASK RXFIFO overflow interrupt enable bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable Note: When the RXFIFO outflow interrupt is enabled and TXOVIF is detected as 1, an overflow interrupt will be generated	1'b1
[0]	RW	RXUDIMASK RXFIFO underflow interrupt enable bit 1'b0: Interrupts are not enabled 1'b1: interrupt enable Note: When the RXFIFO underflow interrupt is enabled and TXUDIF is detected as 1, an underflow interrupt will be generated	1'b1

16.9.4 Interrupt Flag Register

Table 133 I2S interrupt flag register

bit access		Instructions	reset value
[31:13] RO		RSV	19'h0
[12]	RO	TXIF I2S transmit interrupt flag 1'b0: No I2S interrupt occurred	1'b0

		1'b1: I2S has a transmit interrupt generated	
[11]	RO	RXIF I2S receive interrupt flag 1'b0: No I2S interrupt occurred 1'b1: I2S has receive interrupt generated	1'b0
[10]	RO	I2SIF I2S interrupt flag bit 1'b0: No I2S interrupt occurred 1'b1: I2S interrupt generated Note: This bit will be set whenever there is an interrupt on either RX or TX	1'b0
[9]	RW	LZCIF Left channel zero cross detection flag This bit indicates that the next sample data sign bit of the left channel changes or all data bits are zero. 1'b0: No zero crossing detected 1'b1: Zero crossing detected Note: Write 1 to clear the interrupt flag	1'b0
[8]	RW	RZCIF Right channel zero cross detection flag This bit indicates that the next sample data sign bit of the right channel changes or all data bits are zero. 1'b0: No zero crossing detected 1'b1: Zero crossing detected Note: Write 1 to clear the interrupt flag	1'b0

[7]	RW	TXDONEIF Transmit Complete Interrupt Flag 1'b0: This sending is not completed 1'b1: This sending is complete Note: Write 1 to clear the interrupt flag	1'b0
[6]	RO	TXTIF RXFIFO interrupt flag 1'b0: The number of words in the TXFIFO is greater than the threshold 1'b1: The number of words in the TXFIFO is less than or equal to the threshold. Note: When the number of words in TXFIFO (TXCNT) is equal to or less than the threshold set by TXTH, this bit will be It is set to 1 until the data is written to the TXFIFO and the value of TXCNT is greater than the value of TXTH, it will not changes back to 0.	1'b0
[5]	RW	TXOVIF TXFIFO overflow interrupt flag 1'b0: TXFIFO overflow interrupt did not occur 1'b1: TXFIFO overflow interrupt occurred Note: Write 1 to clear the interrupt flag	1'b0
[4]	RW	THUDIF TXFIFO underflow interrupt flag 1'b0: TXFIFO underflow interrupt did not occur 1'b1: TXFIFO underflow interrupt occurred Note: Write 1 to clear the interrupt flag	1'b0

[3]	RW	RXDONEIF Receive complete interrupt flag 1'b0: This receive is not complete 1'b1: This reception is complete Note: Write 1 to clear the interrupt flag	1'b0
[2]	RO	RXTHIF RXFIFO interrupt flag 1'b0: The number of words in the RXFIFO is less than the threshold 1'b1: The number of words in the RXFIFO is equal to or greater than the threshold. Note: When the number of words in RXFIFO is equal to or more than the threshold set by RXTH, this bit will be set to 1, It will not change back to 0 until the data in RXFIFO is read and the value of RXCNT is less than the value of RXTH.	1'b0
[1]	RW	RXOVIF RXFIFO overflow interrupt flag 1'b0: RXFIFO overflow interrupt did not occur 1'b1: RXFIFO overflow interrupt occurred Note: Write 1 to clear overflow interrupt	1'b0
[0]	RW	RXUDIF RXFIFO underflow interrupt flag 1'b0: RXFIFO underflow interrupt did not occur 1'b1: RXFIFO underflow interrupt occurred Note: Write 1 to clear underflow interrupt	1'b0

16.9.5 Status Register

Table 134 I2S Status Register

bit access		Instructions	reset value
[31:10] RO		RSV	22'h0
[9:8]	RO	YOU WILL BE VALID The number of bytes available in the last word. 2'b00: After reception is complete, all bytes in the RXFIFO are available 2'b01: 1 byte is available in the RXFIFO after reception is complete 2'b10: 2 bytes are available in the RXFIFO after reception is complete 2'b11: 3 bytes are available in the RXFIFO after reception is complete	2'h0
[7:4]	RO	TXT Record the number of words in TXFIFO at the current moment. 4'b0000: no data 4'b0001: 1 word … 4'b1000: 8 words	4'h0
[3:0]	RO	RXCNT Record the number of words in the RXFIFO at the current moment. 4'b0000: no data 4'b0001: 1 word … 4'b1000: 8 words	4'h0

16.9.6 Data Transmit Register

Table 135 I2S data transmission register

bit access		Instructions	reset value
[31:0] WO		TXFIFO I2S has a built-in 8-word FIFO for storing data to be sent. Write a word to TXFIFO at a time, The word in the TXFIFO is increased by one. The I2S controller will automatically send out the word that enters the TXFIFO first.	32'h0

16.9.7 Data Receive Register

Table 136 I2S Data Receive Register

bit access		Instructions	reset value
[31:0] RO		RXFIFO I2S has a built-in 8-word FIFO for storing received data. Read from the RXFIFO one at a time word, there will be one less word in the RXFIFO.	32'h0

17 UART module

17.1 Function overview

UART is a universal serial data bus used for asynchronous communication. The bus supports bidirectional communication, enabling full-duplex transmission and reception.

W800 has a total of 6 groups of common UART ports, through the combination of fine clock frequency division, various baud rate settings can be realized, and the maximum support is 2Mbps communication rate. W800 UART can work with hardware DMA to realize high-efficiency asynchronous transmission of data.

17.2 Main features

- ÿ Comply with APB bus interface protocol, full-duplex asynchronous communication mode
- ÿ Support interrupt or polling mode
- ÿ Support DMA Byte transmission mode, send and receive each 32-byte FIFO
- ÿ Programmable baud rate, maximum support 2Mbps
- ÿ 5-8bit data length, and parity polarity can be configured
- ÿ 1 or 2 stop bits configurable
- ÿ Support RTS/CTS flow control
- ÿ Support Break frame sending and receiving
- ÿ Support Overrun, parity error, frame error, rx break frame interrupt indication

17.3 Functional description

17.3.1 UART Baud Rate

In asynchronous communication, since both sides do not have the same clock source as a reference, both parties need to send and receive data according to the negotiated baud rate. W800 Fine baud rate control can be achieved through the baud rate setting register BAUD_RATE_CTRL register ÿ

BAUD_RATE_CTRL[15:0] is named ubdiv, BAUD_RATE_CTRL[19:16] is named ubdiv_frac, the wave that needs to be set

Baudrate, the calculation formula is as follows:

ubdiv = apbclk / (16 * baudrate) – 1 //Take an integer

ubdiv_frac = (apbclk % (baud rate * 16)) / baud rate // lock

Take APB clock 40MHz, baud rate 19200bps as an example:

ubdiv = 40000000 / (16 * 19200) – 1 = 129

ubdiv_frac = (40000000 % (19200* 16)) / 19200 = 3

Calculate the APB clock 40MHz according to the above formula, and the baud rate is 19200bps, the baud rate register should be set as:

BAUD_RATE _CTRL = (3<<16) | 129 = 0x0003_0081ÿ

17.3.2 UART Data Format

ÿData **length**

The UART of W800 supports configurable data length of 5bit, 6bit, 7bit and 8bit. The definition of data length is as follows:

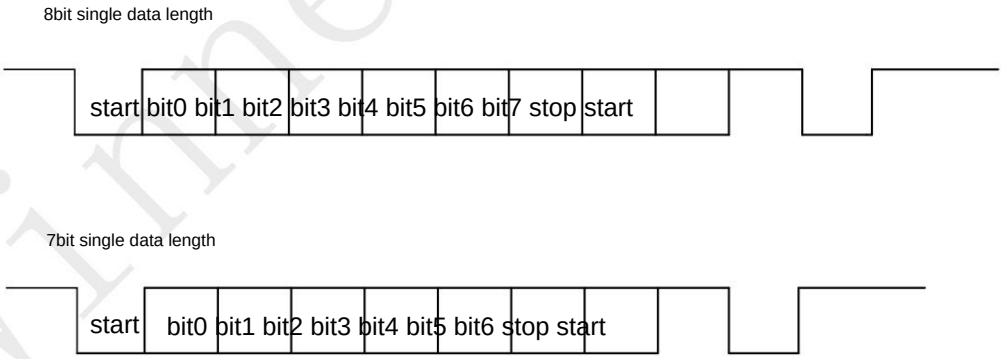


Figure 28 UART data length

Normal UART communication is composed of 1bit start bit, 1bit stop bit plus the middle data bit, and the middle data bit can be configured,

W800 supports configurable data bit lengths of 5bit, 6bit, 7bit, and 8bit, and the data bit length can be selected according to actual applications.

ÿ **stop bit**

The UART of W800 supports configurable 1bit stop bit and 2bit stop bit, which can be configured according to actual needs, as follows:

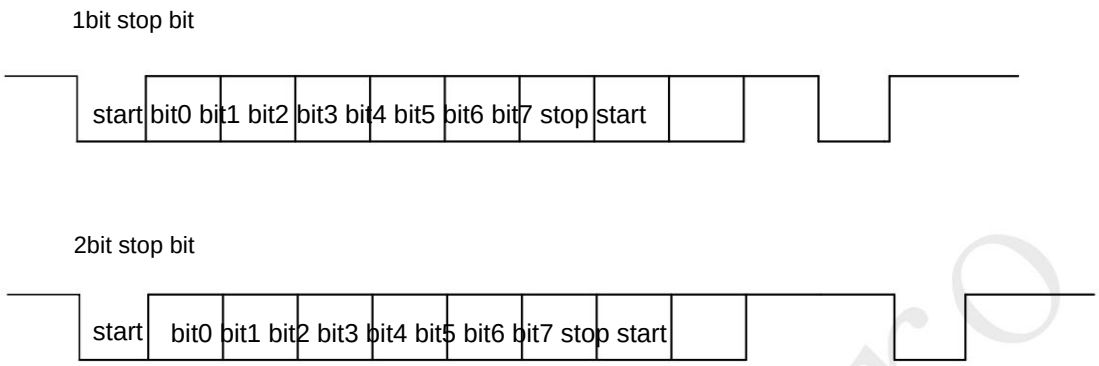


Figure 29 UART stop bit

ÿ **Parity bit**

The function of the parity bit is to check the correctness of the data, W800 can set odd parity, even parity and no parity.

Odd parity calculation method: If the number of 1s in the current data is odd, the odd parity bit is 0, and if the number of 1s in the current data is even

Several, the odd parity bit is 1. In short, an odd number of 1s is guaranteed.

Calculation method of even parity: If the number of 1s in the current data is odd, the even parity bit is 1, and if the number of 1s in the current data is even

Several, the even parity bit is 0. In short, an even number of 1s is guaranteed.

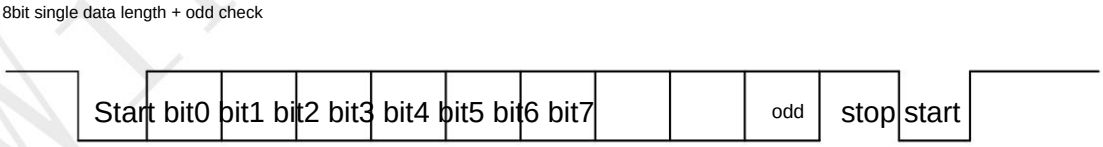
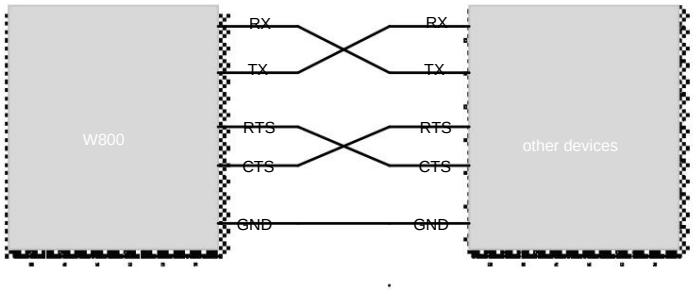


Figure 30 UART parity bit

17.3.3 UART hardware flow control

W800 UART supports RTS/CTS hardware flow control. The main purpose of flow control is to prevent the UART fifo



The data is lost because the software is too late to process it. RTS and CTS are used correspondingly, as shown in the figure below:

Figure 31 UART hardware flow control connection

The hardware flow control of W800 is controlled by AUTO_FLOW_CTRL register. When hardware flow control AUTO_FLOW_CTRL[0] is 1

, W800 will perform flow control settings according to the number of data in rxfifo set by AUTO_FLOW_CTRL[4:2], if it is greater than the set number,

When RTS is pulled high, other devices will no longer send data to W800; when the number is less than the set number, RTS is pulled low, and other devices continue to send data

according to. When AUTO_FLOW_CTRL[0] is 0, the software sets the level of RTS through AUTO_FLOW_CTRL[1].

When W800 sends data, it can judge whether the current CTS has changed through interrupts, and query the CTS status through FIFO_STATUS[12].

To decide whether to continue sending data to other devices.

17.3.4 UART DMA transfer

The UART of W800 supports DMA transmission mode, and the DMA_CTRL register in the UART register list needs to be configured for DMA transmission.

device to turn on the DMA enable of the UART. At the same time, you need to configure UART_FIFO_CTRL to configure how many bytes are left in txfifo and rxfifo.

Send DMA transfer.

DMA source or destination address is set to TX_DATA_WINDOW or RX_DATA_WIDNOW, other DMA register setting parameters

Check the DMA register chapter.

Note: UART DMA transfer can only be set to Byte mode, and half_word and word transfer modes are not supported.

17.3.5 UART Interrupts

UART supports interrupt operation mode, including fifo empty, fifo reaches the set trigger value, CTS changes, errors, etc. will generate UART interrupts,

The required interrupt can be set through the INT_MASK register.

When the UART interrupt is generated, the current interrupt status and the cause of the interrupt can be queried through INT_SRC. Write 1 to clear 0 by software.

17.4 Register Description

17.4.1 Register List

Table 137 UART register list

offset address	name	abbreviation	access	describe	reset value
0X0000 Data flow control register		UART_LINE_CTRL	Data format setting for RW uart communication		0X0000_000B
0X0004 Automatic hardware flow control register		AUTO_FLOW_CTRL	RW uart rts/cts hardware flow control setting		0X0000_0014
0X0008	DMA Setup Register	DMA_CTRL	RW uart dma transfer mode setting		0X0000_0024
0X000C	FIFO Control Register	UART_FIFO_CTRL	RW set uart fifo trigger level		0X0000_0014
0X0010 Baud rate control register		BAUD_RATE_CTRL	RW set uart communication baud rate		0X0003_0081
0X0014 interrupt mask register		INT_MASK	RW Set the interrupt 0X0000_01FF that uart needs to use		
0X0018 Interrupt Status Register		INT_SRC	RW uart interrupt status indicator		0X0000_0000

0X001C	FIFO status register	FIFO_STATUS	RW fifo status, cts status query		0X0000_1000
0X0020	TX start address register TX_DATA_WINDOW WO				0X0000_0000
0X0024 Reserved					
0X0028 Reserved					
0X002C Reserved					
0X0030	RX start address register RX_DATA_WINDOW RO				0X0000_0000
0X0034 Reserved					
0X0038 Reserved					
0X003C Reserved					

17.4.2 Data Flow Control Register

Table 138 UART data flow control register

bit access		Instructions	reset value
[31: 8]		reserve	
[7]	RW	uart_rx_enable Receive enable, active high	1'b0
[6]	RW	uart_tx_enable Transmit enable, active high.	1'b0
[5]	RW	send break enable Send a break packet. Uart will send a break packet after it is set, and the sending is completed Automatically cleared to 0 afterwards.	1'b0
[4]	RW	parity polarity	1'b0

		1'b0: even parity 1'b1: odd parity	
[3]	RW	parity one Parity check enable, high effective	1'b1
[2]	RW	number of stop bits 1'b0: 1 stop bit 1'b1: 2 stop bits	1'b0
[1 : 0] RW		uart bit length. 2'h0ÿ5bit 2'h1ÿ6bit 2'h2ÿ7bit 2'h3ÿ8bit	2'h3

17.4.3 Automatic Hardware Flow Control Registers

Table 139 UART automatic hardware flow control register

bit access		Instructions	reset value
[31: 5]		reserve	
[4 : 2] RW		RTS trigger level When afc_enable is active, decide when RTS needs to be deasserted. 3'h0: rxfifo has more than 4 bytes 3'h1: rxfifo has more than 8 bytes	3'h5

		3'h2: rxfifo has more than 12 bytes 3'h3: rxfifo has more than 16 bytes 3'h4: rxfifo has more than 20 bytes 3'h5: rxfifo has more than 24 bytes 3'h6: rxfifo has more than 28 bytes 3'h7: rxfifo has more than 31 bytes	
[1]	RW	RTS set When AFC_enable is invalid, software can complete receive flow control by setting this bit. when This bit is don't care when AFC_enable is active.	1'b0
[0]	RW	afc enable 1'b1: Valid, the reception condition rts is generated using rts_trigger_level control.	1'b0

17.4.4 DMA Setup Register

Table 140 UART DMA Setup Register

bit access		Instructions	reset value
[31: 8]		reserve	
[7 : 3] RW		rxfifo timeout num If the data in rxfifo is less than rxfifo_trigger_level, if there are no When new data is received, an rxfifo timeout interrupt is generated. After the timing function is enabled, no matter whether it is the first timing or the last timing is completed, only when at least 1 The timer starts after a packet	5'h04
[2]	RW	rxfifo timeout en	1'b1

		rxfifo timeout enable	
[1]	RW	rx dma enable Receive DMA enable, active high. 0: Indicates that the receiving process uses interrupts.	1'b0
[0]	RW	tx dma enable Transmit DMA enable, active high. 0: Indicates that the sending process uses interrupts.	1'b0

17.4.5 FIFO Control Register

Table 141 UART FIFO Control Register

bit access		Instructions	reset value
[31: 6]		reserve	
[5 : 4] RW		rxfifo trigger level When the number of data bytes in rxfifo is greater than or equal to this value, an interrupt is triggered, or rxdma req is triggered. 2'h0ÿ1byte 2'h1ÿ4byte 2'h2ÿ8byte 2'h3ÿ16byte	2'h1
[3 : 2] RW		txfifo trigger level When the number of data bytes in txfifo is less than or equal to this value, an interrupt is triggered, or txdma req is triggered. 2'h0ÿempty 2'h1ÿ4byte	2'h1

		2'h2ÿ8byte 2'h3ÿ16byte	
[1]	RW	rxfifo reset Reset rxfifo, clear the rxfifo status	1'b0
[0]	RW	txfifo reset Reset txfifo, clear txfifo status	1'b0

17.4.6 Baud Rate Control Register

Table 142 UART baud rate control register

bit access		Instructions	reset value
[31:20]		reserve	
[19:16] RW		ubdiv_frac Indicates the fractional part of the quotient of the system clock divided by 16 times the baud rate clock. The specific value is frac×16. (Refer to Chapter 2.3.2, Baud Rate Calculation Method)	4'h3
[15: 0] RW		ubdiv Subtract 1 from the integer part of the quotient of the system clock divided by 16 times the baud rate clock. The default system clock is 40MHz and the baud rate is 19200. (Refer to Chapter 2.3.2, Baud Rate Calculation Method)	16'h81

17.4.7 Interrupt Mask Register

Table 143 UART interrupt mask register

bit access		Instructions	reset value

[31: 9]		reserve	
[8]	RW	overrun error int mask, rxfifo overflow interrupt mask bit, effective high.	1'b1
[7]	RW	parity error int mask, parity check interrupt mask bit, high effective.	1'b1
[6]	RW	frame error int mask, data frame error interrupt mask bit, high effective.	1'b1
[5]	RW	break detect int mask, break signal detection interrupt mask bit, high effective.	1'b1
[4]	RW	cts changed indicate mask, CTS signal change interrupt mask bit, high effective.	1'b1
[3]	RW	rxfifo data timeout int mask, rxfifo receive data timeout interrupt mask bit, high effective.	1'b1
[2]	RW	rxfifo trigger level int mask, rxfifo trigger level interrupt mask bit, effective high.	1'b1
[1]	RW	txfifo trigger level int mask, txfifo trigger level interrupt mask bit, effective high.	1'b1
[0]	RW	txfifo empty int mask, txfifo is an empty interrupt mask bit, high effective.	1'b1

17.4.8 Interrupt Status Register

Table 144 UART interrupt status register

bit access		Instructions	reset value
[31: 9]		reserve	
[8]	RW	overrun error rxfifo overflowed. Software actively writes 1 to clear 0.	1'b0
[7]	RW	parity error The check digit of the received packet is wrong. In case of DMA, this interrupt will still be generated. But DMA operations don't care about this interrupt. Software actively writes 1 to clear 0.	1'b0

[6]	RW	frame error Received packet with wrong stop bit. In case of DMA, this interrupt will still be generated. But DMA operations don't care about this interrupt. Software actively writes 1 to clear 0.	1'b0
[5]	RW	break detect A break packet is received. In case of DMA, this interrupt will still be generated. But DMA operations don't care about this interrupt. Software actively writes 1 to clear 0.	1'b0
[4]	RW	cts changed This interrupt is generated when the cts signal changes. Software actively writes 1 to clear 0.	1'b0
[3]	RW	rxfifo data timeout The data length in rxfifo is less than rxfifo trigger level but no data is received for N data cycles, An interrupt is generated. Software actively writes 1 to clear 0.	1'b0
[2]	RW	rxfifo trigger level interrupt When the number of data in rxfifo changes from less than the number specified in rxfifo trigger level to greater than or equal to the number , this interrupt is generated. At this time, the current data frame size should be determined according to the rxfifo count. Software actively writes 1 to clear 0.	1'b0
[1]	RW	txfifo trigger level interrupt When the number of data in txfifo changes from greater than the number specified in txfifo trigger level to less than or equal to the number	1'b0

		, an interrupt is generated. Software actively writes 1 to clear 0.	
[0]	RW	tx fifo empty interrupt This interrupt is generated when the current packet has been sent and the txfifo is empty. Software actively writes 1 to clear 0.	1'b0

17.4.9 FIFO Status Register

Table 145 UART FIFO Status Register

bit access		Instructions	reset value
[31:13]		reserve	
[12]	RW	cts status Current state of cts	1'b0
[11: 6] RW		rxfifo count The number of data in rxfifo	6'h0
[5 : 0] RW		txfifo count The number of data in txfifo	6'h0

17.4.10TX Start Address Register

Table 146 UART TX start address register

bit access		Instructions	reset value
[31:0] WO		tx data window Send data start address.	32'h0

		Note: uart only supports byte operation when sending and receiving data. When using burst transmission, it is possible to use word The method of incrementing section address, the design supports up to 16-burst operation, that is, 16byte. So send from / A total of 16 bytes (4 words) after the receiving start address are reserved as the sending/receiving data window.	

17.4.11RX Start Address Register

Table 147 UART RX start address register

bit access		Instructions	reset value
[31: 0] RO		rx data window Receive data start address. Note: uart only supports byte operation when sending and receiving data. When using burst transmission, it is possible to use word The method of incrementing section address, the design supports up to 16-burst operation, that is, 16byte. So send from / A total of 16 bytes (4 words) after the receiving start address are reserved as the sending/receiving data window.	32'h0

18 UART&7816 modules

18.1 Function overview

UART&7816 module is compatible with UART function and 7816 interface function.

W800 supports 1 group of UART&7816 multiplexing interface (uart2). When used as UART, it can

Realize the setting of various baud rates, and the maximum communication rate can support 2Mbps.

W800 UART&7816 interface can cooperate with hardware DMA to realize high-efficiency data transmission.

18.2 Main Features

ÿ Conforms to the APB bus interface protocol, supports UART asynchronous full-duplex and 7816 asynchronous half-duplex communication methods;

ÿ Support interrupt or polling mode;

ÿ Support DMA Byte transmission mode, send and receive each 32-byte FIFO;

ÿ Serial port function:

- ÿ Programmable baud rate, maximum support 2Mbps
- ÿ 5-8bit data length, and parity polarity can be configured
- ÿ 1 or 2 stop bits configurable
- ÿ Support RTS/CTS flow control
- ÿ Support Break frame sending and receiving
- ÿ Support Overrun, parity error, frame error, rx break frame interrupt indication

ÿ 7816 interface function:

- ÿ Compatible with ISO-7816-3 T=0.T=1 mode

ÿ Compatible with EVM2000 protocol

ÿ Configurable guard time (11 ETU-267 ETU)

ÿ Forward/reverse agreement, software configurable

ÿ Support sending/receiving parity check and retransmission function

ÿ Support 0.5 and 1.5 stop bits configurable

18.3 UART Functional Description

Refer to Chapter 16 UART Function Module Description

18.4 7816 Functional Description

18.4.1 Introduction to 7816

ISO7816 is an international standard smart card protocol, which stipulates the physical characteristics, size and interface, electrical signal and transmission protocol, command Orders, safety and other information.

W800 mainly implements the part of ISO 7816-3 electrical signal and transmission protocol, and supports T0 and T1 cards. Through the 7816 interface of W800 The user can not care about the signal communication logic of the clock and data, and can directly interact with the smart card for data and commands. About Chi The user needs to refer to the ISO7816-4 protocol to realize the data and command interaction mode of the energy card.

18.4.2 7816 interface

W800 mainly integrates the clock and data interfaces of the smart card to realize the communication electrical signal logic of data and commands. The actual smart card should In use, there are three signals: RST, VCC, and GND. RST can be controlled by ordinary GPIO, mainly when the smart card is powered on and reset. use. VCC can be directly connected to a 3.3V power supply, or through GPIO and other circuits to control the on-off of the smart card VCC.

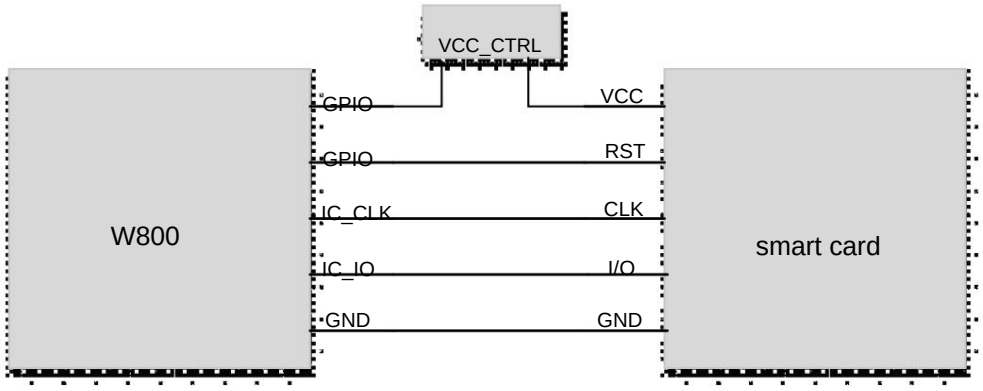


Figure 32 7816 connection diagram

18.4.3 7816 configuration

When used as a 7816 interface, relevant configurations are required:

ÿ Set the working mode of the interface, set UART_LIN_CTRL[24] to 1, and select the current interface as 7816 mode;

ÿ Set 7816 MSB or LSB transmission mode, UART_LIN_CTRL[3] is set to 1, through UART_LIN_CTRL[3]

Select whether the 7816 interface is MSB mode (bit7 is transmitted first), or LSB mode (bit0 is transmitted first);

ÿ Set the stop bit, UART_LIN_CTRL[2] can select 0.5 or 1.5 stop bits for the smart card;

ÿ Select the card type, UART_LIN_CTRL[8] can select T0 card or T1 card;

ÿ Configure the smart card communication timeout time, set the timeout time by WAIT_TIME, when receiving data and not receiving data after timeout

A timeout interrupt is generated.

18.4.4 7816 Clock Configuration

The smart card clock refers to the clock provided to the smart card through the CLK pin, which is set by BAUD_RATE_CTRL[21:16] and calculated

Methods as below:

$$clk\_div = \frac{\_}{2 \times \_} ÿ\ 1$$

fsc_clk: CLK that needs to be provided to the smart card;

fclk_apb: system APB bus clock;

clk_div: The clock division factor that needs to be set to BAUD_RATE_CTRL[21:16]

Because clk_div can only take integers, in order to reduce errors, we'd better adopt the rounding calculation method, and the rounding calculation in C language will lose

The decimal part is dropped, and the rounding conversion method adopted by the C language is as follows:

clk_div = (fclk_apb + fsc_clk)/(2 * fsc_clk) - 1;

18.4.5 7816 rate setting

There is a time unit ETU in the smart card, and the smart card transmits data and commands according to this time unit. ETU is set by

BAUD_RATE_CTRL[15: 0] to set, the calculation method is as follows:

1 1etu = ×

f: CLK of our smart card;

Both F and D are parameters given by the smart card.

In fact, the ubdiv of BAUD_RATE_CTRL[15: 0] we need to set is F/D, the above formula is just for calculating ETU

home reference. When we actually set it, we only need to set ubdiv = F/D. F and D can be queried from the table below.

Table 7 — *Fi* and *f* (max.)

Bits 8 to 5	0000	0001	0010	0011	0100	0101	0110	0111
Fi	372	372	558	744	1116	1488	1860	RFU
f **(max.) MHz**	4	5	6	8	12	16	20	—

Bits 8 to 5	1000	1001	1010	1011	1100	1101	1110	1111
Fi	RFU	512	768	1024	1536	2048	RFU	RFU
f **(max.) MHz**	—	5	7,5	10	15	20	—	—

— According to Table 8, bits 4 to 1 encode *Di*.

Table 8 — *Di*

Bits 4 to 1	0000	0001	0010	0011	0100	0101	0110	0111
Di	RFU	1	2	4	8	16	32	64

Bits 4 to 1	1000	1001	1010	1011	1100	1101	1110	1111
Di	12	20	RFU	RFU	RFU	RFU	RFU	RFU

Table 148 7816 rate setting

(Refer to the protocol document ISO_IEC_FDIS_7816-3_(E).PDF)

18.4.6 7816 power-on reset

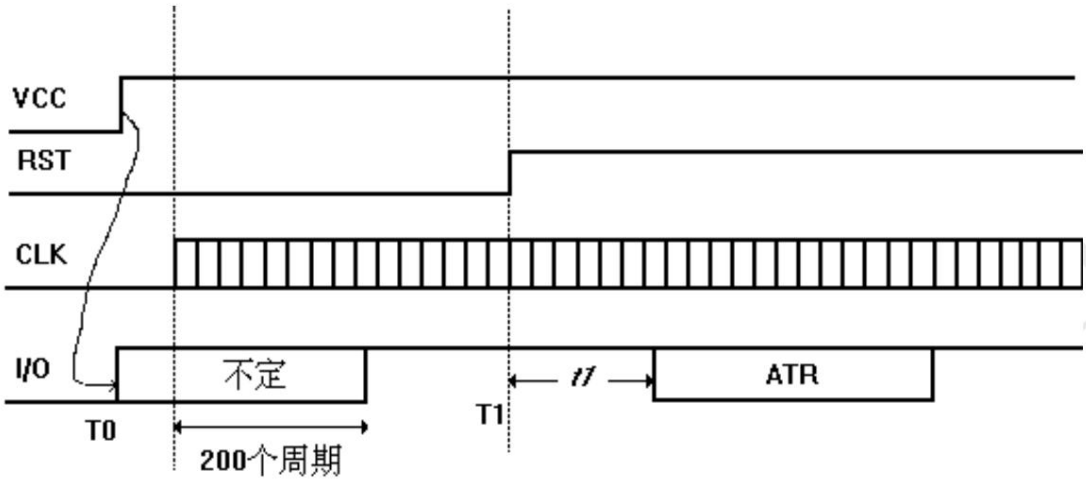


Figure 33 7816 power-on reset sequence

The above figure is the timing diagram of smart card power-on reset. The initial state of CLK and IO is low, and we need to configure it as GPIO mode and pull it low. VCC pull

After high, CLK and IO are configured as 7816 mode and controlled by 7816. Finally, we need to manually pull the RST pin high to complete the reset process.

Procedure. The configuration steps are as follows:

- ÿ I/O, CLK, and RST are configured as normal GPIO mode and kept low;
- ÿ Set 7816 to T=0 mode;
- ÿ Control VCC power-on through GPIO;
- ÿ Configure I/O, CLK as 7816 mode, and 7816 drives clock and data;
- ÿ Configure 7816 clock frequency and allow clock output;
- ÿ Set the RST pin and wait to receive ATR data. If no ATR data is received within 40000 clocks, the inactivation process will be executed.

The card is deactivated.

18.4.7 7816 Warm Reset

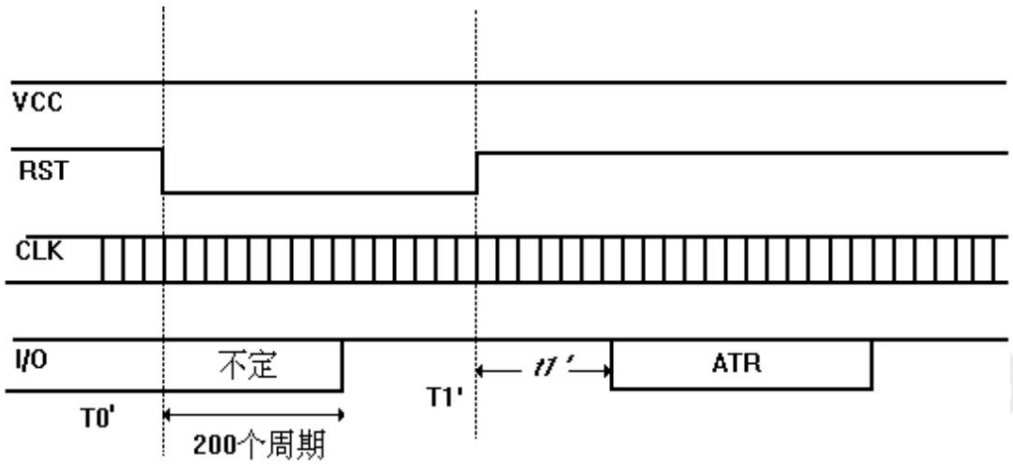


Figure 34 7816 warm reset

As shown in the figure above, the process of warm reset is very simple. In normal working mode, just pull the RST pin down for 400 cycles. The configuration steps are as follows:

ÿ Keep VCC powered on;

ÿ Pull down the RST pin for at least 400 clock cycles;

ÿ Pull the RST pin high, waiting to receive ATR data, if no ATR data is received within 40000 clocks, execute the inactivation process,

The card is deactivated.

18.4.8 7816 Inactivation process

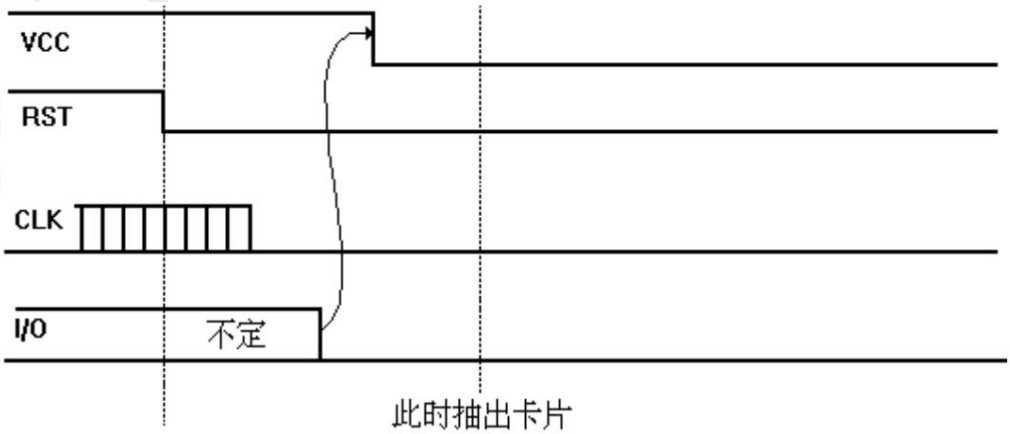


Figure 35 7816 inactivation process

As shown in the figure above, after RST is pulled low, CLK and IO need to be configured as normal IO mode and pulled low, and finally turn off the VCC power supply, the operation steps as follows:

ÿ Keep VCC powered on;

ÿ Pull down the RST pin;

ÿ Configure CLK and IO as GPIO mode and pull it low;

ÿ Control VCC power-down through GPIO;

18.4.9 7816 data transmission

The timing of data transmission of 7816 has been completed by W800 hardware, no user operation is required. If users want to know the specific content of this part, please Refer to the provisions in the ISO7816-3 protocol.

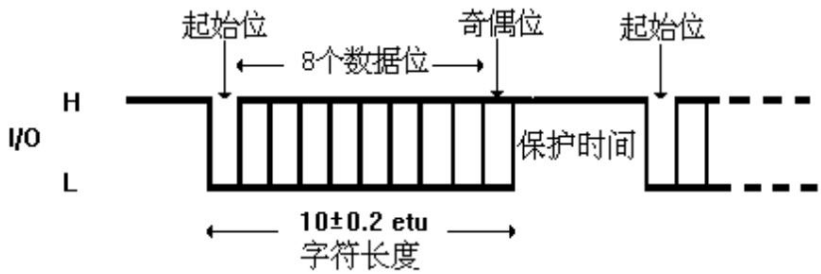


Figure 36 7816 data transmission

18.4.10UART&7816 DMA transfer

The UART&7816 of W800 supports DMA transmission mode, and the UART&7816 register list needs to be configured for DMA transmission.

DMA_CTRL register to enable DMA of UART. At the same time, you need to configure UART_FIFO_CTRL to configure txfifo and rxfifo How many bytes are left to trigger a DMA transfer.

DMA source or destination address is set to TX_DATA_WINDOW or RX_DATA_WIDNOW, other DMA register setting parameters

Check the DMA register chapter.

Note: UART&7816 DMA transmission can only be set to Byte mode, and half_word and word transmission modes are not supported.

18.4.11 UART&7816 interrupt

UART&7816 supports interrupt operation mode, including fifo empty, fifo reaches the set trigger value, CTS change, error, etc. will be generated

UART&7816 interrupt, you can set the required interrupt through INT_MASK register.

When the UART&7816 interrupt is generated, you can query the current interrupt status and the cause of the interrupt through INT_SRC. software write 1 clear

0ÿ

18.5 Register Description

18.5.1 Register List

Table 149 UART&7816 register list

offset address	name	abbreviation	access	describe	reset value
0X0000 Data flow control register		UART_LINE_CTRL	RW uart&7816 communication data related equipment place		0X0033_520B
0X0004 Automatic hardware flow control register		AUTO_FLOW_CTRL	RW uart rts/cts hardware flow control setting		0X0000_0014
0X0008	DMA Setup Register	DMA_CTRL	RW uart&7816 dma transfer mode setting		0X0000_0024
0X000C	FIFO Control Register	UART_FIFO_CTRL	RW set uart&7816 fifo trigger level		0X0000_0014
0X0010 Baud rate control register		BAUD_RATE_CTRL	RW set uart baud rate, 7816 clock		0X0003_0082
0X0014 interrupt mask register		INT_MASK	RW set uart&7816 need to use broken		0X0000_03FF

0X0018 Interrupt Status Register		INT_SRC	RW uart&7816 interrupt status indicator		0X0000_0000
0X001C	FIFO status register	FIFO_STATUS	RW fifo status, cts status query		0X0000_0000
0X0020	TX start address register TX_DATA_WINDOW WO				0X0000_0000
0X0024 Reserved					
0X0028 Reserved					
0X002C Reserved					
0X0030	RX start address register RX_DATA_WINDOW RO				0X0000_0000
0X0034 Reserved					
0X0038 Reserved					
0X003C Reserved					
0X0040	7816 guard time register GUARD_TIME		RW 7816 Guard time between data		0X0000_0000
0X0044	7816 timeout register WAIT_TIME		RW 7816 Receive data timeout		0X0007_8000

18.5.2 Data Flow Control Register

Table 150 UART&7816 data flow control register

bit access		Instructions	reset value
[31:25]		reserve	
[24]	RW	sc_mode 1'b0: uart mode 1'b1: 7816 mode	1'b0
[23]	RW	7816 card T0 mode rx_retrans_en 1'b0: Rx automatic retransmission is invalid	1'b0

		1'b1: Rx automatic retransmission enable	
[22:20] RW		7816 card T0 mode rx_retrans_cnt	3'h3
[19]	RW	7816 card T0 mode tx_retrans_en 1'b0: Tx automatic retransmission is invalid 1'b1: Tx automatic retransmission enable	1'b0
[18:16] RW		7816 card T0 mode tx_retrans_cnt tx automatic retransmission times	3'h3
[15:11] RW		The minimum MIN_BGT (Min Block Guard Time) of the 7816 card Min Block Guard Time calculation: 10+stop bits (default 2 bits)+configuration value MIN_BGT Note: T=0: The minimum time interval between the falling edges of the start bit of two consecutive characters sent and received in opposite directions Cannot be less than 16 ETUs. must be able to correctly interpret the falling edge of its start bit received and the last word transmitted The interval between the falling edges of the section start bit is 15 ETU characters. T=1: The minimum time interval between the falling edges of the start bit of two consecutive characters sent and received in opposite directions (Block Guard Time, BGT) must be 22 ETUs. must be able to correctly interpret the received start bit under its The falling edge and the falling edge of the start bit of the last transmitted byte are separated by characters received within 21 ETUs.	5'ha
[10]	RW	7816 Card Clock Control Configuration 1'b0: Generate card clock output when configured as card mode, otherwise card clock output is invalid 1'b1: clock stopped	1'b0
[9]	RW	7816 Whether to receive data when the parity of the card is wrong 1'b0: do not receive 1'b1: receive	1'b1

[8]	RW	T0/T1 mode configuration of 7816 card, 1'b0: T0 mode 1'b1: T1 mode	1'b0
[7]	RW	uart_rx_enable In uart/7816 mode, receive enable, active high	1'b0
[6]	RW	uart_tx_enable In uart/7816 mode, transmit enable, active high.	1'b0
[5]	RW	send break enable Send a break packet. Uart will send a break packet after it is set, and the sending is completed Automatically cleared to 0 afterwards.	1'b0
[4]	RW	parity polarity (UART mode) 1'b0: even parity 1'b1: odd parity Forward and reverse (7816 mode) 1'b0: LSB (b0 bit) is transmitted first 1'b1: MSB (b7 bit) transmitted first	1'b0
[3]	RW	parity enable, active high (UART mode)	1'b1
[2]	RW	Number of stop bits (UART mode) 1'b0: 1 stop bit 1'b1: 2 stop bits Number of stop bits (7816 mode) 1'b0: 0.5 stop bits	1'b0

		1'b1: 1.5 stop bits	
[1 : 0] RW		uart bit length (UART mode) 2'h0ÿ5bit 2'h1ÿ6bit 2'h2ÿ7bit 2'h3ÿ8bit	2'h3

18.5.3 Automatic Hardware Flow Control Registers

Table 151 UART&7816 automatic hardware flow control register

bit access		Instructions	reset value
[31: 5]		reserve	
[4 : 2] RW		RTS trigger level (UART mode) When afc_enable is active, decide when RTS needs to be deasserted. 3'h0: rxfifo has more than 4 bytes 3'h1: rxfifo has more than 8 bytes 3'h2: rxfifo has more than 12 bytes 3'h3: rxfifo has more than 16 bytes 3'h4: rxfifo has more than 20 bytes 3'h5: rxfifo has more than 24 bytes 3'h6: rxfifo has more than 28 bytes 3'h7: rxfifo has more than 31 bytes	3'h5

[1]	RW	RTS set (UART mode) When AFC_enable is invalid, software can complete receive flow control by setting this bit. when This bit is don't care when AFC_enable is active.	1'b0
[0]	RW	afc enable (UART mode) The reception condition rts is generated using rts_trigger_level control, high effective.	1'b0

18.5.4 DMA Setup Register

Table 152 UART&7816 DMA setting register

bit access		Instructions	reset value
[31: 8]		reserve	
[7 : 3] RW		rxfifo timeout num (UART mode) If the data in rxfifo is less than rxfifo_trigger_level, if there are no When new data is received, an rxfifo timeout interrupt is generated. After the timing function is enabled, no matter whether it is the first timing or the last timing is completed, only when at least 1 The timer starts after a packet	5'h4
[2]	RW	rxfifo timeout en (UART&7816 mode) rxfifo timeout enable, active high	1'b1
[1]	RW	rx dma enable (UART&7816 mode) Receive DMA enable, active high. 0 indicates that the receive process uses interrupts.	1'b0
[0]	RW	tx dma enable (UART&7816 mode) Transmit DMA enable, active high.	1'b0

		0 indicates that the sending process uses interrupts.	

18.5.5 FIFO Control Register

Table 153 UART&7816 FIFO control register

bit access		Instructions	reset value
[31: 6]		reserve	
[5 : 4] RW		rxfifo trigger level(UART&7816 mode) When the number of data bytes in rxfifo is greater than or equal to this value, an interrupt is triggered, or rxdma req is triggered. 2'h0ÿ1byte 2'h1ÿ4byte 2'h2ÿ8byte 2'h3ÿ16byte	2'h1
[3 : 2] RW		txfifo trigger level(UART&7816 mode) When the number of data bytes in txfifo is less than or equal to this value, an interrupt is triggered, or txdma req is triggered. 2'h0ÿempty 2'h1ÿ4byte 2'h2ÿ8byte 2'h3ÿ16byte	2'h1
[1]	RW	rxfifo reset(UART&7816 mode) Reset rxfifo, clear the rxfifo status	1'b0
[0]	RW	txfifo reset(UART&7816 mode) Reset txfifo, clear txfifo status	1'b0

18.5.6 Baud Rate Control Register

Table 154 UART&7816 baud rate control register

bit access		Instructions	reset value
[31:20]		reserve	
[19:16] RW		ubdiv_frac UART mode: Indicates the fractional part of the quotient of the system clock divided by 16 times the baud rate clock. The specific value is frac×16. (refer to chapter baud rate calculation method) 7816 mode: ubdiv_frac = (fclk_apb + fsc_clk)/(2 * fsc_clk) - 1; (refer to 7816 clock calculation method)	4'h3
[15: 0] RW		ubdiv UART mode: Subtract 1 from the integer part of the quotient of the system clock divided by 16 times the baud rate clock. The default system clock is 40MHz and the baud rate is 19200. (Refer to the baud rate calculation method) 7816 mode: ubdiv=Fi/Di (Fi, Di are the parameters fed back by the smart card, edu frequency: f_etuclk = fsc_clk/(ubdiv+1)) (Refer to Section 7816 Rate Calculation Method)	16'h82

18.5.7 Interrupt Mask Register

Table 155 UART&7816 interrupt mask register

bit access		Instructions	reset value
[31:10]		reserve	
[9]	RW	The 7816 card received an error signal error signal when sending. (7816 mode)	1'b1
[8]	RW	overrun error int mask, rxfifo overflow interrupt mask bit, effective high. (UART&7816 mode)	1'b1
[7]	RW	parity error int mask, parity check interrupt mask bit, high effective. (UART&7816 mode)	1'b1
[6]	RW	frame error int mask, data frame error interrupt mask bit, high effective. (UART mode)	1'b1
[5]	RW	break detect int mask, break signal detection interrupt mask bit, high effective. (UART mode)	1'b1
[4]	RW	cts changed indicate mask, CTS signal change interrupt mask bit, high effective. (UART mode)	1'b1
[3]	RW	rxfifo data timeout int mask, rxfifo receive data timeout interrupt mask bit, high effective. (UART & 7816 model)	1'b1
[2]	RW	rxfifo trigger level int mask, rxfifo trigger level interrupt mask bit, effective high. (UART&7816 module Mode)	1'b1
[1]	RW	txfifo trigger level int mask, txfifo trigger level interrupt mask bit, effective high. (UART&7816 module Mode)	1'b1
[0]	RW	txfifo empty int mask, txfifo is an empty interrupt mask bit, high effective. (UART&7816 mode)	1'b1

18.5.8 Interrupt Status Register

Table 156 UART&7816 Interrupt Status Register

bit access		Instructions	reset value
[31: 9]		reserve	

[9]	RW	The 7816 card received an error signal error signal when sending. (7816 mode)	
[8]	RW	overrun error (UART&7816 mode) rxfifo overflowed. Software actively writes 1 to clear 0.	1'b0
[7]	RW	parity error (UART&7816 mode) The check digit of the received packet is wrong. In case of DMA, this interrupt will still be generated. But DMA operations don't care about this interrupt. Software actively writes 1 to clear 0.	1'b0
[6]	RW	frame error (UART mode) Received packet with wrong stop bit. In case of DMA, this interrupt will still be generated. But DMA operations don't care about this interrupt. Software actively writes 1 to clear 0.	1'b0
[5]	RW	break detect (UART mode) A break packet is received. In case of DMA, this interrupt will still be generated. But DMA operations don't care about this interrupt. Software actively writes 1 to clear 0.	1'b0
[4]	RW	cts changed (UART mode) This interrupt is generated when the cts signal changes. Software actively writes 1 to clear 0.	1'b0
[3]	RW	rxfifo data timeout (UART&7816 mode) The data length in rxfifo is less than rxfifo trigger level but no data is received for N data cycles, An interrupt is generated.	1'b0

		Software actively writes 1 to clear 0.	
[2]	RW	rxfifo trigger level interrupt (UART&7816 mode) When the number of data in rxfifo changes from less than the number specified in rxfifo trigger level to greater than or equal to the number , this interrupt is generated. At this time, the current data frame size should be determined according to the rxfifo count. Software actively writes 1 to clear 0.	1'b0
[1]	RW	txfifo trigger level interrupt (UART&7816 mode) When the number of data in txfifo changes from greater than the number specified in txfifo trigger level to less than or equal to the number , an interrupt is generated. Software actively writes 1 to clear 0.	1'b0
[0]	RW	tx fifo empty interrupt (UART&7816 mode) This interrupt is generated when the current packet has been sent and the txfifo is empty. Software actively writes 1 to clear 0.	1'b0

18.5.9 FIFO Status Register

Table 157 UART&7816 FIFO status register

bit access		Instructions	reset value
[31:13]		reserve	
[12]	RW	cts status (UART mode) Current state of cts	1'b0

[11: 6] RW		rxfifo count (UART&7816 mode) The number of data in rxfifo	6'h0
[5 : 0] RW		txfifo count (UART&7816 mode) The number of data in txfifo	6'h0

18.5.10 TX Start Address Register

Table 158 UART&7816 TX start address register

bit access		Instructions	reset value
[31:0] WO		tx data window (UART&7816 mode) Send data start address. Note: uart only supports byte operation when sending and receiving data. When using burst transmission, it is possible to use word The method of incrementing section address, the design supports up to 16-burst operation, that is, 16byte. So send from / A total of 16 bytes (4 words) after the receiving start address are reserved as the sending/receiving data window.	32'h0

18.5.11 RX Start Address Register

Table 159 UART&7816 RX start address register

bit access		Instructions	reset value
[31: 0] RO		rx data window (UART&7816 mode) Receive data start address. Note: uart only supports byte operation when sending and receiving data. When using burst transmission, it is possible to use word The method of incrementing section address, the design supports up to 16-burst operation, that is, 16byte. So send from / A total of 16 bytes (4 words) after the receiving start address are reserved as the sending/receiving data window.	32'h0

18.5.12 7816 Guard Time Register

Table 160 7816 guard time register

bit access		Instructions	reset value
[31: 8]		reserve	
[7 : 0] RW		ex_gt_num In 7816 mode, guard time calculation: 10+stop bits+configuration value ex_gt_num	8'h0

18.5.13 7816 Timeout Time Register

Table 161 7816 timeout register

bit access		Instructions	reset value
[31:24]		reserve	
[23: 0] RW		wait time counter (in ETU) In 7816 mode: CWT and BWT times, configured to maximum default values. (In T1 mode: BWT = (11 etu+ 2BWI*960*Fd/fsc))	24'h78000

19 Timer module

19.1 Function overview

The timer consists of a 32-bit auto-reload counter driven by a divided system clock. W800 has 6 completely independent

stand timer. Accurate timing and interrupt functions are realized, which can be used for delay or periodic event processing.

19.2 Main Features

- ÿ 6 fully independent timers
- ÿ 32-bit auto-reload counter
- ÿ The timing unit can be configured as ms, us
- ÿ Can realize single timing or repeat timing function
- ÿ Timing interrupt function
- ÿ The timer count value can be queried at any time;

19.3 Functional description

The timer module is composed of 6 completely independent timers, independent of each other, and 6 channels can work at the same time.

The system clock is divided by the frequency division factor to obtain the us standard clock, which is used for the input clock of the counter. Timing unit can be configured as us, ms

kind of level.

The timing value is a 32-bit configurable register, which can meet the needs of different timing durations. Each timer corresponds to an interrupt, when

After the timing time is satisfied, if the interrupt function is enabled, an interrupt request will be generated, which can be used to process periodic events.

19.3.1 Timing function

The timing function refers to the time set by the user, and when the time is up, a hardware interrupt is generated, and the user is notified to implement a specific function. Timed Trigger Support Ticket

There are two kinds of times and periods, one can be used to handle single events, and the other can be used to handle periodic events.

The user obtains the APB bus clock frequency according to the frequency division factor of the system clock, and sets the reference microsecond count configuration register of the timer

(TMR_CONFIG), set the timing value, configure the timing unit, work mode, enable interrupt, and then start the timing function. when timed

When the time is up, the program enters the timer interrupt processing function to clear the interrupt.

19.3.2 Delay function

The delay function means that the user can keep the program in a waiting state according to the countdown function of the timer, and the program will not continue to run until the timing is completed.

Row.

19.4 Register Description

19.4.1 Register List

Table 162 Timer register list

offset address name		abbreviated access		describe	reset value
0X0000 Standard us configuration register		TMR_CONFIG RW		Standard us Timing frequency division value, by bus time clock frequency division to get the standard us timing, the value Equal to APB bus frequency (MHz) minus 1	0X0000_0027
0X0004 Timer control register		TMR_CSR	RW	Timer Control Register	0X0631_8C63
0X0008 Timer 1 timing value configuration register	TMR1_PRD		RW	Timer1 timing value configuration register	0X0000_0000
0X000C Timer 2 timing value configuration register	TMR2_PRD		RW	Timer2 timing value configuration register	0X0000_0000
0X0010 Timer 3 timing value configuration register	TMR3_PRD		RW	Timer3 timing value configuration register	0X0000_0000

0X0014 Timer 4 timing value configuration register TMR4_PRD			RW	Timer4 timing value configuration register 0X0000_0000	
0X0018 Timer 5 timing value configuration register TMR5_PRD			RW	Timer5 timing value configuration register 0X0000_0000	
0X001C Timer 6 timing value configuration register TMR6_PRD			RW	Timer6 timing value configuration register 0X0000_0000	
0X0020 Timer 1 current count value		TMR1_CNT	RO	Timer1 current count value	0X0000_0000
0X0024 Timer 2 current count value		TMR1_CNT	RO	Timer2 current count value	0X0000_0000
0X0028 Timer 3 current count value		TMR1_CNT	RO	Timer3 current count value	0X0000_0000
0X002C Timer 4 current count value		TMR1_CNT	RO	Timer4 current count value	0X0000_0000
0X0030 Timer 5 current count value		TMR1_CNT	RO	Timer5 current count value	0X0000_0000
0X0034 Timer 6 current count value		TMR1_CNT	RO	Timer6 current count value	0X0000_0000

19.4.2 Standard us configuration register

Table 163 Timer standard us configuration register

bit access		Instructions	reset value
[31: 7]		reserve	
[6 : 0] RW		Clock frequency division configuration prescale. E.g: apb_clk=40MHz prescale = 40 – 1 = 8'd39	7'h27

19.4.3 Timer Control Register

Table 164 Timer timer control register

bit access		Instructions	reset value

[31:30] RW		reserve	2'h0
[29:25] RW		TMR6_CSR, same as TMR1_CSR	5'h3
[24:20] RW		TMR5_CSR, same as TMR1_CSR	5'h3
[19:15] RW		TMR4_CSR, TMR1_CSR	5'h3
[14:10] RW		TMR3_CSRÿÿ TMR1_CSR	5'h3
[9 : 5] RW		TMR2_CSR, same as TMR1_CSR	5'h3
[4 : 0] RW		[4 : 0] is TMR1_CSR, as follows:	
		[4]: Interrupt status register, write 1 to clear 1'b0: Timer does not generate interrupt; 1'b1: Timer generates an interrupt;	1'b0
		[3]: Interrupt enable register 1'b0: No interrupt will be generated after the timing is completed; 1'b1: Generate an interrupt after the scheduled time is completed;	1'b0
		[2]: Timer enable register 1'b0: The timer does not work; 1'b1: Enable timer	1'b0
		[1]: Timer working mode 1'b0: The timer repeats timing; 1'b1: The timer only times once, and it will be automatically turned off after the timing is completed;	1'b1
		[0]: Timer timing unit 1'b0: The timing unit is us; 1'b1: timing unit is ms;	1'b1

19.4.4 Timer 1 timing value configuration register

Table 165 Timer 1 timing value configuration register

bit access	Instructions	reset value
[31: 0] RW	Configure the timing value of timer 1	32'b0

19.4.5 Timer 2 timing value configuration register

Table 166 Timer 2 timing value configuration register

bit access	Instructions	reset value
[31: 0] RW	Configure the timing value of timer 2	32'b0

19.4.6 Timer 3 timing value configuration register

Table 167 Timer 3 timing value configuration register

bit access	Instructions	reset value
[31: 0] RW	Configure the timing value of timer 3	32'b0

19.4.7 Timer 4 timing value configuration register

Table 168 Timer 4 timing value configuration register

bit access	Instructions	reset value
[31: 0] RW	Configure the timing value of timer 4	32'b0

19.4.8 Timer 5 timing value configuration register

Table 169 Timer 5 timing value configuration register

bit access	Instructions	reset value
[31: 0] RW	Configure the timing value of timer 5	32'b0

19.4.9 Timer 6 timing value configuration register

Table 170 Timer 6 timing value configuration register

bit access	Instructions	reset value
[31: 0] RW	Configure the timing value of timer 6	32'b0

19.4.10 Timer 1 Current Count Value Register

Timer 1 current count value register

bit access	Instructions	reset value
[31: 0] RO	Read the current count value of timer 1;	32'b0

19.4.11 Timer 2 Current Count Value Register

Timer 2 current count value register

bit access	Instructions	reset value
[31: 0] RO	Read the current count value of timer 2;	32'b0

19.4.12 Timer 3 Current Count Value Register

Timer 3 current count value register

bit access	Instructions	reset value
[31: 0] RO	Read the current count value of timer 3;	32'b0

19.4.13 Timer 4 Current Count Value Register

Timer 4 current count value register

bit access	Instructions	reset value
[31: 0] RO	Read the current count value of timer 4;	32'b0

19.4.14 Timer 5 Current Count Value Register

Timer 5 current count value register

bit access	Instructions	reset value
[31: 0] RO	Read the current count value of timer 5;	32'b0

19.4.15 Timer 6 Current Count Value Register

Timer 6 current count value register

bit access	Instructions	reset value
[31: 0] RO	Read the current count value of timer 6;	32'b0

20 power management module

20.1 Function overview

The PMU realizes the switching of the working state of the chip hardware and the power management during the state switching process. It also provides timers, real-time clocks and 32K clock.

20.2 Main Features

- ÿ Provide chip power control
- ÿ Provide timer function
- ÿ Provide real-time clock control
- ÿ Provide 32K RC oscillator calibration function
- ÿ Provide wake-up function;

20.3 Functional description

20.3.1 Full Chip Power Control

The PMU module controls the power switch of the chip, including 40M oscillation circuit, BandGap, digital PLL, voltage detection circuit, digital circuit LDOÿ

When the chip is powered on, the PMU module guides each module to turn on the power in turn according to the preset power-on sequence;

When the software configuration register enters the sleep mode, guide each functional module to turn off the power in turn according to the safe power-off sequence;

When the software configuration register enters sleep mode, turn off the clock, crystal oscillator circuit and related power supply in order;

Provides three wake-up modes in sleep/sleep mode: Timer scheduled wake-up, RTC scheduled wake-up or through the special WAKEUP pin Pull high to wake up.

20.3.2 Low Power Modes

Two low-power modes can be selected by configuring the PMU register chip;

Standby mode:

In this mode, the power supply of the digital power domain will be turned off, and only the PMU module of the whole chip will work, providing wake-up and reset functions;

Consumption is about 15uA or so. After the power is turned off, all the data and content stored in the memory will be lost, and the firmware will be reloaded after waking up, which is equivalent to restarting the

start up;

Sleep mode:

In this mode, the power of the digital power domain will be reserved, only the DPLL and the crystal oscillator circuit will be turned off, and the clock will be cut off. At this time, the power consumption of the whole chip is about

About 1mA; the data and code stored in the memory will still be retained; the program will continue to run after waking up;

20.3.3 Wake-up Mode

PMU supports 3 wake-up modes, Timer wake-up, RTC wake-up and external IO wake-up.

Timer wake up

Before the software sets the sleep/sleep mode, configure the Timer0 module in the PMU and set the sleep time. When the system enters sleep mode, the

When Timer0 reaches the sleep time, it will wake up the system and generate a corresponding Timer interrupt. After the system resumes operation, it is necessary to correct the interruption status.

Write '1' to the corresponding status bit in the status register to clear the interrupt status, otherwise, it will be woken up by the interrupt immediately after entering the sleep mode next time;

RTC wakeup

Before the software sets the sleep/sleep mode, configure the RTC module in the PMU and set the sleep time. When the system enters sleep mode, when

When the RTC timer reaches the sleep time, the system will be woken up and the corresponding RTC interrupt will be given. After the system resumes running, please set the interrupt register 0x14

Write '1' to the corresponding status bit in the corresponding status bit to clear the interrupt status, otherwise, it will be woken up by the interrupt immediately after entering the sleep mode next time;

Wake up from external IO

After software dormancy/sleep, the PMU will detect a specific Wakeup pin, the external controller can wake up the system by pulling this IO high, and give

Corresponding IO wake-up interrupt. The PMU no longer detects this IO state after leaving sleep mode. After the system resumes running, please set the interrupt register 0x14

Write '1' to the corresponding status bit in the corresponding status bit to clear the interrupt status, otherwise, it will be woken up by the interrupt immediately after entering the sleep mode next time;

20.3.4 Timer0 Timer

Configure the timer enable signal and timing time through the AHB register. First set the timing value, then set the timer enable BIT to start timing

When the timer reaches the specified time, an interrupt is generated, and the software clears the interrupt flag by writing BIT0 of the status register.

20.3.5 Real Time Clock Function

Reference Real Time Clock Module

20.3.6 32K clock source switching and calibration

The W800 chip integrates a 32K RC oscillator as the clock source of the PMU module.

Due to changes in the working environment and temperature, the output frequency of the 32K RC oscillator may change, resulting in timing deviation. Therefore, in the PMU module

The 32K RC oscillator calibration function is introduced in the block, and the switching function of the 32K clock is used to correct the timing deviation.

1) Switching of 32K clock source

The 32K clock can be switched from the 32K RC oscillator to the frequency divided by the 40M clock by setting bit3 of the PS_CR register to 1.

to the 32K clock. However, when the chip enters sleep mode, because the 40M clock will be turned off, bit3 will be cleared to 0 automatically. after waking up

If the firmware still needs to use the precise timing function, it is necessary to reset bit3 to 1.

2) Calibration of 32K RC oscillator circuit

First set bit2 of PS_CR register to 0, and then set bit2 of PS_CR register to 1.

After calibration, the 32K RC oscillator will be relatively accurate. But if you want more accurate timing, it is recommended to use 40M clock frequency division

The obtained 32K clock will be an accurate RTC clock at this time, and the deviation will only be related to the crystal frequency deviation.

20.4 Register Description

20.4.1 Register List

Table 171 PMU register list

offset address	name	abbreviated access		describe	reset value
0X0000	PMU Control Register PS_CR		RW	is used to configure 32K calibration, configure 32K clock Source, set the STANDBY function of the chip	0X0000_0002
0X0004	PMU Timer 0	TIMER0	RW	Configure timing value (unit is second), enable timing device	0X0000_0000
0X0008 Reserved					
0X0014	PMU interrupt source register INT_SRC		RW	provides PMU interrupt flag	0X0000_0000

20.4.2 PMU Control Registers

Table 172 PMU Control Register

bit access		Instructions	reset value

[31: 11] RO			24'b0
[10]	RW	Wake-up button interrupt polarity selection: 0: Set an interrupt when the wake-up button is at a high level; 1: Set an interrupt when the wake-up button is at a high level; Only valid in Sleep interrupt;	1'b0
[9:6]	RW	The minimum hold time for key-press wake-up: Unit: 128ms;	4'd01
[5]	RW	DLDO_CORE reference voltage source selection 1: ABG 0: DBG	1'b1
[4]	RW	32K oscillator circuit BYPASS signal High effective, after set to 1, 32K is obtained by dividing the frequency of 40M clock. •2	1'b0
[3]	RW	RC 32K oscillator calibration circuit start switch; 1'b0: Calibration circuit reset; 1'b1: start the calibration circuit; To start the calibration function, this bit needs to be set to 0 first and then to 1.	1'b0
[2]	RW	Enable the key to trigger the sleep function 0: disable; 1: In active mode, press the io_wakeup button to reach the threshold time , will trigger io_sleep_flag is interrupted and reported to MCU.	1'b0
[1]	RW	Sleep enable signal, high effective. 1'b0: chip wake-up state	1'b1

		1'b1: The chip enters the Sleep state If WAKEUP pin is inactive level, and TIMER0/1 interrupt wakeup is not configured, this register When valid, the chip enters the Sleep state; If the wake-up interrupt is generated, the chip will switch from the Sleep state to the wake-up state, and the wake-up condition is met. This bit is automatically cleared to 0. **Wakeup source: WAKEUP pin, TIMER0/TIMER1, RTC** 1) The WAKEUP pin is active high; if the chip enters the Sleep state, the WAKEUP pin must be in low level. To wake up, pull the WAKEUP pin high to generate a wake-up interrupt and make the chip leave Sleep state. 2) TIMER0, timer wake-up interrupt. When the WAKEUP pin is low, TIMER0 sets the timing time and enables it, and when the timing time is up, it will generate a call Wake up interrupt to make the chip leave the Sleep state. 3) RTC, timing time to wake up When the WAKEUP pin is low and the RTC timing is up, a wake-up interrupt will be generated to make the chip leave the Sleep state. state	
[0]	RW	STANDBY enable signal, high effective. 1'b0: chip wake-up state 1'b1: chip enters STANDBY state If WAKEUP pin is inactive level, and TIMER0/1 interrupt wakeup is not configured, this register When valid, the chip enters the STANDBY state; If the wake-up interrupt is generated, the chip will switch from the STANDBY state to the wake-up state, and the wake-up condition is satisfied sufficient, this bit is automatically cleared to 0.	1'b0

		Wakeup source: WAKEUP pin, TIMER0/TIMER1, RTC 4) The WAKEUP pin is active high; if the chip enters the STANDBY state, the WAKEUP pin must be in the at low level. To wake up, pull up the WAKEUP pin to generate a wake-up interrupt and make the chip leave STANDBY state. 5) TIMER0, timer wake-up interrupt. When the WAKEUP pin is low, TIMER0 sets the timing time and enables it, and when the timing time is up, it will generate a call Wake up interrupt to make the chip leave STANDBY state. 6) RTC, timing time to wake up When the WAKEUP pin is low and the RTC timing time is up, a wake-up interrupt will be generated to make the chip leave STANDBY state	

20.4.3 PMU Timer 0

Table 173 PMU Timer 0 Register

bit access		Instructions	reset value
[31:17] RO reserved			15'b0
[16]	RW	Timer0 enable bit 1'b0: bit enable. 1'b1: enable;	1'b0
[15: 0] RW Timer0 timing value, unit: second			16'b0

20.4.4 PMU Interrupt Source Registers

Table 174 PMU interrupt source register

bit access		Instructions	reset value
[31: 9]	R reserved		
[8]	RW shows the current power-on status:	1'b0: Power-on or reset start 1'b1: Wake up from sleep state, write 1 to clear	1'b0
[7]	RO reserved		1'b0
[6]	RO reserved		1'b0
[5]	RW RTC timing interrupt flag bit:	1'b0: A timer interrupt is generated 1'b1: no scheduled interrupt, write 1 to clear	1'b0
[4]	RW Reserved		1'b0
[3]	RW Reserved		1'b0
[2]	RW WAKEUP pin wakeup interrupt flag bit	1'b0: No WAKEUP interrupt generated 1'b1: WAEKUP wakeup interrupt is generated, write 1 to clear	1'b0
[1]	RW Reserved		1'b0
[0]	RW Timer0 timing interrupt flag bit:	1'b0: No Timer0 interrupt generated 1'b1: Timer0 interrupt is generated, write 1 to clear	1'b0

21 Real Time Clock Module

21.1 Function overview

RTC is a BCD counter/timer provided by the PMU module. Two 32-bit registers contain seconds, minutes, hours, days, months, years, and

The decimal format representation (BCD) of the base code can automatically correct the months with 28, 29 (leap year), 30, and 31 days.

Under the corresponding software configuration, RTC can not only provide the clock calendar function, but also can be used as a timer, when the timer reaches the set time

Then an RTC interrupt will be generated, which can be used to wake up the system in sleep state.

The RTC module has two clock sources that can be configured: 40M clock frequency division and internal 32K clock. Which can be used by software configuration during normal work

clock source; only 32K clock can be used in sleep state. If the RTC clock source in the normal working state is obtained by dividing the frequency of the 40M clock, then

After entering the sleep state, it will automatically switch to the 32K clock, and the system will still use the 32K clock after it is woken up. So as long as the power supply voltage

The RTC module will not stop working no matter whether the module is in normal working state or sleeping state.

21.2 Main Features

- ÿ Provide timing function
- ÿ Provide timing function
- ÿ Provide timing interrupt
- ÿ Interrupt to wake up the system

21.3 Functional Description

21.3.1 Timing function

The initial value of day, hour, minute and second can be configured in RTC configuration register 1, and the initial value of year and month can be configured in RTC configuration register 2.

The RTC configuration register 2 enables the timekeeping function.

After the RTC timing function is enabled, read the RTC configuration register 1 to get the current date, hour, minute, second value, read the RTC configuration register

Register 2 can get the current year and month value.

21.3.2 Timing function

In RTC configuration register 1, the timing value of day, hour, minute and second can be configured; in RTC configuration register 2, the timing value of year and month can be configured; in

RTC configuration register 1 enables timing functions.

When the RTC timer reaches the timing time, an RTC interrupt will be generated. At this time, set the RTC interrupt bit of the PMU interrupt source register to 1 to clear

In addition to interrupt status.

When the system enters the sleep mode, the interrupt generated by the RTC timer will wake up the system.

21.4 Register Description

21.4.1 Register List

The RTC module has two 32-bit dedicated registers, and the RTC interrupt status needs to query the PMU interrupt source register.

Table 175 RTC register list

offset address	name	abbreviated	access	describe	reset value
0X000C	RTC configuration register 1	RTC_R1	RW	Configure the value of RTC day, hour, minute and second, and configure the timing to enable	0X0000_0000
0X0010	RTC configuration register 2	RTC_R2	RW	Configure RTC year and month value, configuration enable timing	0X0000_0000

21.4.2 RTC Configuration Register 1

Table 176 RTC configuration register 1

bit	access	Instructions	reset value
[31]	RW	RTC timing interrupt function enable	1'b0

		1'b0: disable 1'b1: enable	
[30:29]		reserve	
[28:24] RW		Daily initial value/daily fixed value	5'b0
[23:21]		reserve	
[20:16] RW		Hour initial value/hour timer value	5'b0
[15:14]		reserve	
[13: 8]		reserve	
[7 : 6]		reserve	
[5 : 0] RW		Second initial value/second timing value	6'b0

21.4.3 RTC Configuration Register 2

Table 177 RTC configuration register 2

bit access		Instructions	reset value
[31:17]		reserve	
[16]	RW	RTC timing function enable bit 1'b0: disable 1'b1: enable	1'b0
[15]		reserve	
[14: 8] RW		Beginning of the year value/annual fixed value	7'b0
[7 : 4]		reserve	
[3 : 0] RW		Initial value/Monthly fixed value	4'b0

22 Watchdog module

22.1 Function overview

Realize the "watchdog" function. Designed for global reset on system crash.

The "watchdog" will generate a periodic interrupt, and the system software will clear its interrupt flag after the interrupt is generated, if it is not cleared after the set time

In addition, a hard reset signal will be generated to reset the system.

22.2 Main Features

- ÿ Provide timing function
- ÿ Provide reset function
- ÿ Provide timing interrupt

22.3 Functional Description

22.3.1 Timing function

After setting the timing value to the register WD_LD, set the BIT0 of WDG_CTRL to 1 to start the timer, and the WDG module will generate

The life timer is interrupted, and the program is notified to process it. If the bit0 of the register WD_CLR is not cleared, a timer interrupt will be generated periodically.

The value of WD_LD is based on the APB clock unit, and the APB clock is divided from the 160M clock.

22.3.2 Reset Function

After setting the chip timing value WD_LD, start the timing and reset function (set BIT1/BIT0 of WDG_CTRL), and the WDG module will start up and down.

Timing, when the timing time is up, WDG will generate a timing interrupt, and if the BIT0 of WD_CLR is not cleared, the chip will

A reset signal is generated in the next cycle.

22.4 Register Description

22.4.1 Register List

Table 178 WDG register list

offset address	name	abbreviated access	describe	reset value
0X0000	WDG Timing load register WD_LD	RW	configuration timing value for repeated loading	0XFFFF_FFFF
0X0004	WDG current value register WD_VAL	RO	Gets the value of the current timer	0XFFFF_FFFF
0X0008	WDG Control Register WD_CTRL	RW	Control Register	0X0000_0000
0X000C	WDG interrupt clear register WD_CLR	WO	interrupt clear register	0X0000_0000
0X0010	WDG interrupt source register WD_SRC	RO	Interrupt Source Register	0X0000_0000
0X0014	WDG Interrupt output register WD_STATE	RO	Interrupt output status register	0X0000_0000

22.4.2 WDG Timing Value Loading Register

Table 179 WDG timing value loading register

bit access		Instructions	reset value
[31: 0] RW		Configure the timing value for repeated loading The value of this register is counted in APB clock. For example: if the APB clock is 40MHZ, the maximum duration of the timing value is about 107s, that is 0FFFFFFFF/40000000	32'hffff_ffff

22.4.3 WDG Current Value Register

Table 180 WDG current value register

bit access		Instructions	reset value
[31: 0] RO		Get the value of the current timer To calculate the remaining time, just read the current value. To calculate the elapsed time, just subtract the value of register WD_VAL from the value of register WD_LD	32'hffff_ffff

22.4.4 WDG Control Register

Table 181 WDG Control Register

bit access		Instructions	reset value
[31: 2]		reserve	30'h0
[1]	RW	reset enable bit 1'b0: No reset signal is generated when WDG reset condition occurs 1'b1: A reset signal is generated when a WDG reset condition occurs	1'b0
[0]	RW	timing enable bit 1'b0: Timer does not work 1'b1: Timer works and generates periodic interrupt	1'b0

22.4.5 WDG Interrupt Clear Register

Table 182 WDG interrupt clear register

bit access		Instructions	reset value
[31: 1]		reserve	31'h0

[0]	WHERE	Interrupt status clear bit, write any value to clear the current interrupt status	1'b0

22.4.6 WDG Interrupt Source Register

Table 183 WDG interrupt source register

bit access		Instructions	reset value
[31: 1]		reserve	31'h0
[0]	RO	Interrupt source register, the timer function is turned on, and the interrupt will be generated at the same time	1'b0

22.4.7 WDG Interrupt Status Register

Table 184 WDG interrupt status register

bit access		Instructions	reset value
[31: 1]		reserve	31'h0
[0]	RO	Interrupt output status register. This interrupt is not generated after the timer is closed, but WD_SRC may be 1	1'b0

23 PWM controllers

23.1 Function overview

PWM is a method of digitally encoding the level of an analog signal. Through the use of a high-resolution counter, the duty cycle of the square wave is modulated with

to encode the level of a specific analog signal. The PWM signal is still digital because at any given moment, the full-scale direct

DC power supply is either fully present (ON) or completely absent (OFF). A voltage or current source is a repetitive pulse of on (ON) or off (OFF)

The impulse sequence is added to the simulated load. When it is on, it means that the DC power supply is added to the load, and when it is off, it means that the power supply is disconnected.

when. Any analog value can be encoded using PWM as long as the bandwidth is sufficient.

23.2 Main Features

ÿ Support 2-channel input signal capture function (PWM0 and PWM4 two channels)

ÿ Input signal capture function supports interrupt interactive mode and DMA transfer mode; DMA mode supports word-by-word operation

ÿ Support 5-channel PWM signal generation function

ÿ 5-channel PWM signal generation supports one-shot generation mode and auto-load mode

ÿ Support 5-channel braking function

ÿ PWM output frequency range: 3Hz~160kHz

ÿ Maximum accuracy of duty cycle: 1/256, counter width for inserting dead zone: 8bit

ÿ Support channel 0 channel 1 synchronization function, support channel 2 channel 3 synchronization function

ÿ Support complementary and non-complementary modes of channel 0 and channel 1, support complementary and non-complementary modes of channel 2 and channel 3

ÿ Support 5-channel synchronization function

23.3 Functional Description

23.3.1 Input Signal Capture

The PWM controller supports the signal capture function of two channels, and the capture of channel 0 can be activated by setting Bit24 of the PWM_CTL register function, the capture function of channel 4 can be activated by setting Bit1 of the PWM_CAP2CTL register. The level of the captured signal can also be to set whether to flip the function. After the channel captures the corresponding signal, the capture number is updated to the corresponding capture register PWM_CAPDAT (through channel 0 capture number) and PWM_CAP2DAT (channel 4 capture number).

23.3.2 DMA Transfer Capture Number

After the capture function is enabled on channel 0 or channel 4, the count of the capture register can be quickly transferred to the memory through the DMA channel, speeding up user process.

23.3.3 Support for one-shot and automount modes

Each of the five output channels of the PWM controller supports one-shot output mode and auto-reload mode. In single load mode, the channel outputs a specified cycle After three waveforms, the PWM wave will no longer be output; in the automatic loading mode, after the channel outputs the specified period of waveforms, it will automatically reload the period number, thereby continuing to generate PWM waves.

23.3.4 Multiple Output Modes

The PWM controller supports independent output mode, that is, each channel outputs independently without interfering with each other; it supports dual-channel synchronous mode, that is, one channel's The output is completely consistent with the output of another channel; supports five-channel synchronous mode, the output of channel 1 to channel 4 is completely consistent with the output of channel 0 consistent; support dual-channel complementary output, that is, the waveform output by one channel is completely opposite to the waveform output by the other channel; support complementary mode Commonly used dead zone setting, the dead zone length can be set up to 256 clock cycles; support braking mode, when the braking port detects the specified voltage After leveling, the output channel will output the brake level that has been set.

A variety of output modes are flexible and configurable to meet various application scenarios related to PWM by users.

23.4 Register Description

23.4.1 PWM Register List

Table 185 PWM register list

offset address name	abbreviation	access	describe	reset value
0X0000 Clock frequency division register_01	PWM_CLKDIV01	RW	Divide the clock frequency of channel 0 and channel 1	0X0000_0000
0X0004 Clock frequency division register_23	PWM_CLKDIV23	RW	Divide the clock frequency of channel 2 and channel 3	0X0000_0000
0X0008 Control register	PWM_CTL	RW	is used to configure or control some configurable items	0X0000_0000
0X000C period register	PWM_PERIOD	RW	is used to set the period of channel 0 to channel 4	0X0000_0000
0X0010 Cycle number register	PWM_PNUM	RW	is used to set the signal generation of channel 0 to channel 4 Number of cycles	0X0000_0000
0X0014 compare register	PWM_CMPDAT	RW	is used to store the comparison value of channel 0 to channel 4 to produce different duty cycles	0X0000_0000
0X0018 Dead zone control register	PWM_DTCTL	RW	is used to configure or control the configurable set item	0X0000_0000
0X001C interrupt control register	PWM_INTEN	RW	is used to enable and control related interrupts	0X0000_0000
0X0020 interrupt status register	PWM_INTSTS	RW	is used to query the status of related interrupts	0X0000_0000
0X0024 channel 0 capture register	PWM_CAPDAT	RO	is used to capture and count the rising edge and falling edge	0X0000_0000
0X0028 Brake control register	PWM_BRKCTL	RW	is used to control the brake mode	0X0000_0000
0X002C Clock frequency division register_4	PWM_CH4_reg1	RW	Frequency division of channel 4 clock	0X0000_0000

0X0030 Channel 4 control register_1 PWM_CH4_reg2 RW Set the relevant configuration items of channel 4 0X0000_0000					
0X0034 channel 4 capture register PWM_CAP2DAT			RO is used to capture and count the rising edge and falling edge		0X0000_0000
0X0038 Channel 4 control register_2 PWM_CAP2CTL RW Set related configuration items of channel 4 0X0000_0000					

23.4.2 Clock Divider Register_01

Table 186 PWM clock frequency division register_01

bit access		Instructions	reset value
[31:16] RW		CLKDIV1 CH1 frequency division counter The frequency division number is determined by the counter value Note: The frequency division range is (0~65535), if no frequency division is required, input 0 or 1.	16'h0
[15:0] RW		CLKDIV0 CH0 frequency division counter Same as CH1	16'h0

23.4.3 Clock frequency division register_23

Table 187 PWM clock frequency division register_23

bit access		Instructions	reset value
[31:16] RW		CLKDIV3 CH3 frequency division counter Same as CH1	16'h0
[15:0] RW		CLKDIV2	16'h0

		CH2 frequency division counter Same as CH1	

23.4.4 Control Registers

Table 188 PWM Control Register

bit	access	Instructions	reset value
[31:27]	RW	CNTEN Counter count enable 1'b0: stop counting 1'b1: start counting Note: Each bit controls each channel separately, from high to low to control CH4, CH3, CH2, CH1 and CH0	5'b0
[26]		reserve	1'b0
[25]	RW	CAPINV Capture reverse enable flag 1'b0: capture mode input signal reverse invalid 1'b1: The reverse of the input signal in capture mode is valid, and the input signal is reversed	1'b0
[24]	RW	CPEN Capture function enable flag 1'b0: The capture function of CH0 is invalid, and the values of RCAPDAT and FCAPDAT will not be updated; 1'b1: CH0 capture function is valid, capture and latch the PWM counter, respectively stored in RCAPDAT (rising edge latch) and FCAPDAT (falling edge latch)	1'b0
[23:22]	RW	CNTTYPE3	2'b0

		CH3 counter counting mode 2'b00: Edge-aligned mode (the counter counts up, only for capture mode) 2'b01: Edge-aligned mode (the counter counts down, only for PWM mode) 2'b10: Center-aligned mode (PWM mode only) Note: In PWM mode, when the counter is set to edge-aligned mode, it is necessary to set the counting mode to decrement Way.	
[21:20] RW		CNTTYPE2 CH2 counter counting mode Same as CH3	2'b0
[19:18] RW		CNTTYPE1 CH1 counter counting mode Same as CH3	2'b0
[17:16] RW		CNTTYPE0 CH0 counter counting mode Same as CH3	2'b0
[15:14] RW		TWOSYNCEN 2-channel synchronous mode enable signal 1'b0: 2-channel synchronization not allowed 1'b1: allow 2-channel synchronization, PWM_CH0 and PWM_CH1 have the same phase, and the phase is determined by PWM_CH0; PWM_CH2 Has the same phase as PWM_CH3, and the phase is determined by PWM_CH2 15bit control CH3 and CH2	2'b0

		14bit control CH1 and CH0	
[13]	-- reserve		1'b0
[12]	RW	PAIN PWM pin output enable bit 1'b0: PWM pin is set to output state 1'b1: PWM pins are tri-stated Note: only for CH0	1'b0
[11:8] RW		CNTMODE PWM Generation Cycle Mode 1'b0: one-shot mode 1'b1: autoload mode Note: During the change of CNTMODE, PWM_CMPDAT is reset to zero; each bit controls each channel separately, from high to low to control PW3, PW2, PW1 and PW0 in turn	4'h0
[7]	-- reserve		1'b0
[6]	RW	THE ALLSYNC All channel synchronous mode enable signal 1'b0: Synchronization of all channels not allowed 1'b1: Allow synchronization of all channels, PWM_CH0, PWM_CH1, PWM_CH2 and PWM_CH3 have The same phase, and the phase is determined by PWM_CH0	1'b0
[5:2] RW		PINV PWM output signal polarity enable 1'b0: PWM output polarity inversion disabled	4'h0

		1'b1: PWM output polarity inversion enable Note: Each bit controls each channel separately, from high to low to control PW3, PW2, PW1 and PW0	
[1:0] RW		OUTMODE output mode 1'b0: Non-complementary mode every two channels 1'b1: Every two channels form a complementary pattern BIT1 controls CH2 and CH3 BIT0 controls CH0 and CH1	2'b0

23.4.5 Period Register

Table 189 PWM period register

bit access		Instructions	reset value
[31:24] RW		PERIOD3 CH3 period register value (note: period cannot be greater than 255) "Edge-aligned mode (counter counting method is decrementing)": ÿ PERIOD register value, the period value is (PERIOD + 1) ÿ Duty cycle = (CMP+1)/(PERIOD + 1) ÿ CMP>=PERIOD: PWM output is fixed at high ÿ CMP<PERIOD: PWM low level width is (PERIOD-CMP), high level width is (CMP+1) ÿ CMP=0: PWM low level width is PERIOD, high level width is 1; "Middle Alignment Mode": ÿ PERIOD register value: the period is 2*(PERIOD+1)	8'h0

		ÿ Duty cycle=(2*CMP+1)/(2*(PERIOD+1)) ÿ CMP>PERIOD: PWM keeps high ÿ CMP<=PERIOD: PWM low level=2*(PERIOD-CMP)+1, High level = (2*CMP) +1 ÿ CMP=0: PWM low level width is 2*PERIOD+1, high level width is 1. Note: In "middle aligned mode", the number of cycles should not be 255. No matter which alignment mode is selected, the channel period is determined by the frequency division number (N) and the number of periods (P), That is: the input clock is 40MHz, the clock frequency f_div after frequency division is: f_div = 40MHz/N, N is divided Frequency (16bit). The output frequency f_output is: f_output = f_div / P, where P is the number of cycles. Note: In PWM mode, when the counter is set to edge-aligned mode, it is necessary to set the counting mode to decrement Way.	
[23:16] RW		PERIOD2 CH2 period register value (note: period cannot be greater than 255) same PERIOD3	8'h0
[15:8] RW		PERIOD1 CH1 period register value (note: period cannot be greater than 255) same PERIOD3	8'h0
[7:0] RW		PERIOD0 CH0 period register value (note: period cannot be greater than 255) same PERIOD3	8'h0

23.4.6 Cycle Count Register

Table 190 PWM Cycle Number Register

bit access		Instructions	reset value
[31:24] RW		PNUM3 PWM3 generation cycle number Set PWM3 cycle number PNUM3, when PWM generates PNUM3 PWM signals, stop generating signals number, trigger the interrupt and set the interrupt status word at the same time	8'h0
[23:16] RW		PNUM2 PWM2 Generation Cycles same PNUM3	8'h0
[15:8] RW		PNUM1 PWM1 generation cycle number same PNUM3	8'h0
[7:0] RW		PNUM0 PWM0 generation cycle number same PNUM3	8'h0

23.4.7 Compare Register

Table 191 PWM compare register

bit access		Instructions	reset value
[31:24] RW		CMP3 PWM3 compare register value	8'h0

		"Edge-aligned mode (counter counting method is decrementing)": ÿ PERIOD register value, the period value is (PERIOD + 1) ÿ Duty cycle = (CMP+1)/(PERIOD + 1) ÿ CMP>=PERIOD: PWM output fixed bit high ÿ CMP<PERIOD: PWM low level width is (PERIOD-CMP), high level width is (CMP+1) ÿ CMP=0: PWM low level width is PERIOD, high level width is 1; "Middle Alignment Mode": ÿ PERIOD register value: the period is 2*(PERIOD+1) ÿ Duty cycle=(2*CMP+1)/2*(PERIOD+1) ÿ CMP>PERIOD: PWM keeps high ÿ CMP<=PERIOD: PWM low level=2*(PERIOD-CMP)+1, high level=(2*CMP)+1 ÿ CMP=0: PWM low level width is 2*PERIOD+1, high level width is 1. No matter which alignment mode is selected, the channel period is determined by the frequency division number (N) and the period number (P), namely: The input clock is 40MHz, the clock frequency f_div after frequency division is: f_div = 40MHz/N, N is the frequency division number (16bit). The output frequency f_output is: f_output = f_div / P, where P is the number of cycles. Note: In PWM mode, when the counter is set to edge-aligned mode, it is necessary to set the counting mode to decrement Way.	
[23:16] RW		CMP2 PWM2 compare register value Same as CMP3	8'h0
[15:8] RW		CMP1 PWM1 compare register value	8'h0

		Same as CMP3	
[7:0] RW		CMP0 PWM0 compare register value Same as CMP3	8'h0

23.4.8 DEAD-TIME CONTROL REGISTER

Table 192 PWM dead zone control register

bit access		Instructions	reset value
[31:22]	-- reserve		10'h0
[21]	RW	DTEN23 Whether channel 2 and channel 3 can be inserted into the dead zone valid mark The effective signal of inserting dead zone is valid only after the complementary mode of the channel is turned on. And, if a valid signal is inserted If it is 0, there is no dead zone insertion for the complementary signals output by the two channels 1'b0: Insert dead zone invalid 1'b1: Insert dead zone valid	1'b0
[20]	RW	DTEN01 Whether channel 0 and channel 1 can be inserted into the effective flag of the dead zone Same as DTEN23	1'b0
[19:18]	-- reserve		2'b0
[17:16] RW		DTDIV Dead-band clock frequency division control 2'b00: dead zone clock is equal to base clock (40MHz)	2'b0

		2'b01: The dead zone clock is equal to the reference clock (40MHz) divided by two 2'b10: The dead zone clock is equal to the reference clock (40MHz) divided by four 2'b11: The dead zone clock is equal to the reference clock (40MHz) divided by eight	
[15:8] RW		DTCNT23 Dead-band interval for channel 3 and channel 2 8bit determines the dead zone interval value, and the dead zone clock is determined by DTDIV	8'h0
[7:0] RW		DTCNT01 Dead-band interval between channel 1 and channel 0 8bit determines the dead zone interval value, and the dead zone clock is determined by DTDIV	8'h0

23.4.9 Interrupt Control Register

Table 193 PWM interrupt control register

bit	access	Instructions	reset value
[31:8]	-- reserve		24'h0
[7]	RW	DMA_request_EN DMA_request enable 1'b0: DMA_request invalid 1'b1: DMA_request valid	1'b0
[6]	RW	FLY Falling edge buffer interrupt enable bit 1'b0: Falling edge buffer interrupt is invalid 1'b1: Falling edge buffer interrupt is valid	1'b0

		Note: For CH0	
[5]	RW	RLIEN Rising edge buffer interrupt enable bit 1'b0: Rising buffer interrupt invalid 1'b1: Rising edge buffer interrupt is valid Note: For CH0	1'b0
[4:0]	RW	SMALL PWM Period Interrupt Enable Bit 1'b0: Periodic interrupts are inactive 1'b1: Periodic interrupt active Note: When the counter counts to 0 and the number of PWM periods satisfies PWM_PNUM, an interrupt is triggered.	5'b0

23.4.10 Interrupt Status Register

Table 194 PWM Interrupt Status Register

bit access		Instructions	reset value
[31:10]	-- reserve		12'h0
[9]	RW	OVERFL counter overflow flag 1'b0: capture mode, the counter does not overflow during the counter counting process 1'b1: capture mode, counter overflow during counter counting Note: When the user clears CFLIF or CRLIF, this bit will also be cleared at the same time	1'b0
[8]	RW	FLIFOV	1'b0

		Falling edge delay interrupt flags overrun status 1'b0: When CFILF is 1, no falling edge delay interrupt will be generated 1'b1: When CFILF is 1, falling edge delay interrupt occurred again Note: When the user clears CFILF, this bit is also cleared at the same time	
[7]	RW	RLIFOV Rising edge delay interrupt flags overrun status 1'b0: When CRILF is 1, no rising edge delay interrupt will be generated 1'b1: When CRILF is 1, another rising edge delay interrupt occurs Note: When the user clears CRILF, this bit is also cleared at the same time	1'b0
[6]	RW	CFLIF Capture falling edge interrupt flag 1'b0: No falling edge captured 1'b1: When a falling edge is captured, this bit is set to 1 Note: By writing 1, clear this flag; Note: For CH0	1'b0
[5]	RW	CRLIF Capture rising edge interrupt flag 1'b0: rising edge not captured 1'b1: When a rising edge is captured, this bit is set to 1 Note: By writing 1, clear this flag; Note: For CH0	1'b0
[4:0] RW		PIF	5'b0

		PWM Period Interrupt Flag When the PWM generates a PWM signal with a specified period, the flag is set to 1; write 1 by software to clear the flag Note: Each bit identifies each channel separately, and controls PW4, PW3, PW2, PW1 and PW0 in sequence from high to low	

23.4.11 Channel 0 Capture Register

Table 195 PWM channel 0 capture register

bit access		Instructions	reset value
[31:16] RO		PWM_FCAPDAT capture falling edge register When there is a falling edge on the input signal, store the current counter value	16'h0
[15:0] RO		PWM_RCAPDAT capture rising edge register When there is a rising edge on the input signal, store the current counter value	16'h0

23.4.12 Brake Control Register

Table 196 PWM brake control register

bit access		Instructions	reset value
[31:16]	-- reserve		16'h0
[15:11] RW		BRKCTL brake mode enable 1'b0: Brake mode disabled 1'b1: Brake mode active	5'b0

		[7:3] corresponds to CH4, CH3, CH2, CH1 and CH0 respectively	
[10:8]	-- reserve		3'b0
[7:3] RW		BKOD Brake Output Control Register 1'b0: When the brake mode is valid, the PWM output is low 1'b1: When the braking mode is valid, the PWM output is high [7:3] corresponds to CH4, CH3, CH2, CH1 and CH0 respectively	5'b0
[2:0]	-- reserve		3'b0

23.4.13 Clock Divider Register_4

Table 197 PWM clock frequency division register_4

bit access		Instructions	reset value
[31:16] RW		CLKDIV4 CH4 frequency division counter The frequency division number is determined by the counter value Note: The frequency division range is (0~65535), if no frequency division is required, input 0 or 1.	16'h0
[15:8] RW		PERIOD4 CH4 period register value (note: period cannot be greater than 255) "Edge-aligned mode (counter counting method is decrementing)": ÿ PERIOD register value, the period value is (PERIOD + 1) ÿ Duty cycle = (CMP+1)/(PERIOD + 1) ÿ CMP>=PERIOD: PWM output fixed bit high	8'h0

		ÿ CMP<PERIOD: PWM low level width is (PERIOD-CMP), high level width is (CMP+1) ÿ CMP=0: PWM low level width is PERIOD, high level width is 1; "Middle Alignment Mode": ÿ PERIOD register value: the period is 2*(PERIOD+1) ÿ Duty cycle=(2*CMP+1)/(2*(PERIOD+1)) ÿ CMP>PERIOD: PWM keeps high ÿ CMP<=PERIOD: PWM low level=2*(PERIOD-CMP)+1, high level=(2*CMP)+1 ÿ CMP=0: PWM low level width is 2*PERIOD+1, high level width is 1. Note: In "middle aligned mode", the number of cycles should not be 255. No matter which alignment mode is selected, the channel period is determined by the frequency division number (N) and the number of periods (P), That is: the input clock is 40MHz, the clock frequency f_div after frequency division is: f_div = 40MHz/N, N is divided Frequency (16bit). The output frequency f_output is: f_output = f_div / P, where P is the number of cycles. **Note: In** PWM mode, when the counter is set to edge-aligned mode, it is necessary to set the counting mode to decrement Way.	
[7:0] RW		CH4 generation cycle number Set the number of PWM4 cycles to PNUM4, when the PWM generates PNUM4 PWM signals, Stop generating signals while triggering an interrupt and setting the interrupt status word	8'h0

23.4.14 Channel 4 Control Register_1

Table 198 PWM channel 4 control register_1

bit access		Instructions	reset value
[31:16]	-- reserve		16'h0

[15:8] RW		CMP4 CH4 period register value "Edge-aligned mode (counter counting method is decrementing)": ÿ PERIOD register value, the period value is (PERIOD + 1) ÿ Duty cycle = (CMP+1)/(PERIOD + 1) ÿ CMP>=PERIOD: PWM output fixed bit high ÿ CMP<PERIOD: PWM low level width is (PERIOD-CMP), high level width is (CMP+1) ÿ CMP=0: PWM low level width is PERIOD, high level width is 1; "Middle Alignment Mode": ÿ PERIOD register value: the period is 2*(PERIOD+1) ÿ Duty cycle=(2*CMP+1)/2*(PERIOD+1) ÿ CMP>PERIOD: PWM keeps high ÿ CMP<=PERIOD: PWM low level=2*(PERIOD-CMP)+1, high level=(2*CMP)+1 ÿ CMP=0: PWM low level width is 2*PERIOD+1, high level width is 1. No matter which alignment mode is selected, the channel period is determined by the frequency division number (N) and the period number (P), namely: The input clock is 40MHz, the clock frequency f_div after frequency division is: f_div = 40MHz/N, N is the frequency division number (16bit). The output frequency f_output is: f_output = f_div / P, where P is the number of cycles. **Note: In** PWM mode, when the counter is set to edge-aligned mode, it is necessary to set the counting mode to decrement Way.	8'h0
[7:5]	-- reserve		3'b0
[4:3] RW		CNTTYPE4 CH4 counter counting mode	2'b0

		2'b00: Edge-aligned mode (the counter counts up, only for capture mode) 2'b01: Edge-aligned mode (the counter counts down, only for PWM mode) 2'b10: Center-aligned mode (PWM mode only) **Note: In** PWM mode, when the counter is set to edge-aligned mode, it is necessary to set the counting mode to decrement Way.	
[2]	-- reserve		1'b0
[1]	RW	CNTMODE4 CH4 generation cycle mode 1'b0: one-shot mode 1'b1: autoload mode Note: During the change of CNTMODE, PWM_CMPDAT is reset to zero	1'b0
[0]	RW	PINV4 CH4 output signal polarity enable 1'b0: PWM output polarity inversion disabled 1'b1: PWM output polarity inversion enable	1'b0

23.4.15 Channel 4 Capture Register

Table 199 PWM channel 4 capture register

bit access		Instructions	reset value
[31:16] RO		PWM_FCAP2DAT capture falling edge register When there is a falling edge on the input signal, store the current counter value	16'h0

[15:0] RO		PWM_RCAP2DAT capture rising edge register When there is a rising edge on the input signal, store the current counter value	16'h0

23.4.16 Channel 4 Control Register_2

Table 200 PWM channel 4 control register_2

bit access		Instructions	reset value
[31:11]	-- reserve		21'h0
[10]	RW	DMA_request2_mask DMA_request2 enable 1'b0: DMA_request2 invalid 1'b1: DMA_request2 valid Note: only for CH4	1'b0
[9]	RW	FLIEN2 Falling edge buffer interrupt enable bit 1'b0: Falling edge buffer interrupt is invalid 1'b1: Falling edge buffer interrupt is valid Note: only for CH4	1'b0
[8]	RW	RLIEN2 Rising edge buffer interrupt enable bit 1'b0: Rising buffer interrupt invalid 1'b1: Rising edge buffer interrupt is valid Note: only for CH4	1'b0

[7]	RW	SURFACE2 counter overflow flag 1'b0: capture mode, the counter does not overflow during the counter counting process 1'b1: capture mode, counter overflow during counter counting Note: When the user clears CFLIF or CRLIF, this bit will also be cleared at the same time Note: only for CH4	1'b0
[6]	RW	FLIFOV2 Falling edge delay interrupt flags overrun status 1'b0: When CFILF is 1, no falling edge delay interrupt will be generated 1'b1: When CFILF is 1, falling edge delay interrupt occurred again Note: When the user clears CFILF, this bit is also cleared at the same time Note: only for CH4	1'b0
[5]	RW	RLIFOV2 Rising edge delay interrupt flags overrun status 1'b0: When CRILF is 1, no rising edge delay interrupt will be generated 1'b1: When CRILF is 1, another rising edge delay interrupt occurs Note: When the user clears CRILF, this bit is also cleared at the same time Note: only for CH4	1'b0
[4]	RW	CFLIF2 Capture falling edge interrupt flag 1'b0: No falling edge captured 1'b1: When a falling edge is captured, this bit is set to 1	1'b0

		Note: By writing 1, this flag is cleared Note: only for CH4	
[3]	RW	CRLIF2 Capture rising edge interrupt flag 1'b0: rising edge not captured 1'b1: When a rising edge is captured, this bit is set to 1 Note: By writing 1, this flag is cleared Note: only for CH4	1'b0
[2]	RW	POEN2 PWM pin output enable bit 1'b0: PWM pin is set to output state 1'b1: PWM pins are tri-stated Note: only for CH4	1'b0
[1]	RW	CPEN2 Capture function enable flag 1'b0: CH4 capture function is invalid, RCAPDAT and FCAPDAT values will not be updated; 1'b1: CH4 capture function is valid, capture and latch the PWM counter, store in RCAPDAT (rising edge latch) and FCAPDAT (falling edge latch) Note: only for CH4	1'b0
[0]	RW	CAPINV2 Capture reverse enable flag 1'b0: capture mode input signal reverse invalid	1'b0

		1'b1: The reverse of the input signal in capture mode is valid, and the input signal is reversed Note: only for CH4	

24 QFLASH controller

24.1 Function overview

W800 has a built-in QFLASH controller, which provides bus-based QFLASH read, write, and erase operations, and provides access to the system bus and data bus

Arbitration, realize QFLASH read operation in CACHE mode.

24.2 Main Features

- ÿ Provide bus access FLASH interface
- ÿ Support access to FLASH by directly sending SPI commands through register configuration
- ÿ Support 24-bit or 32-bit address access to FLASH
- ÿ Supports automatic decryption and reading of codes and data stored in FLASH

24.3 Functional Description

24.3.1 Bus Access

The address of FLASH in the system starts from 0x08000000, and the address space can be read directly.

24.3.2 Register Access

By configuring the registers, you can directly send SPI commands to FLASH to achieve more flexible access to FLASH, and support most

Divide the control command of FLASH.

24.3.3 Command Configuration and Startup

Write the control command code of Flash into the command bit of the register FLASH_CMD, and configure the register according to the command format

Other bits, for example, if there are address bits in the command, set the address bit to 1. If there is data in the command, the DAT bit must be set to 1, with

Body reference register description.

After the command configuration is completed, set the command_b bit of the CMD_START register to 1, and the controller starts to send the command to the FLASH.

24.3.3.1 Communication mode and configuration

First send commands to FLASH to configure FLASH as QIO or QPI mode, and then configure QIOM or QPIM of FLASH controller

When the bit is set to 1, it can use QIO mode or QPI mode to communicate with FLASH.

24.3.3.2 Data Caching

When reading and writing data to FLASH through registers, use the storage space at address 0x4000 0000 ~ 0x4000 03ff as the data

Data cache, this section of address space is also used as the data cache used by the RSA encryption and decryption module. When reading data, the controller will read from FLASH

The output data is sequentially stored in the address space starting from 0x4000 0000 for caching. When writing data to FLASH, it is also from 0x4000

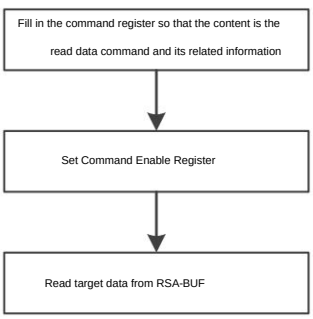
0000 start to read data and send to flash.

24.3.3.3 Read Operation

Each read operation can read up to 256 bytes of data, and can read data in any position of QFlash.

After the read operation command is started, the data is stored in the cache without waiting or querying.

The operation process is shown in the figure.



24.3.3.4 Write Operations

The following commands are all write operations:

WRSR: Write Status Register

PP: page programming

SE: sector erase

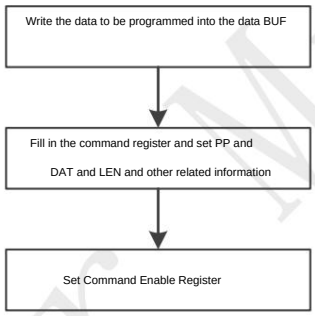
BE: 32K block erase

HBE: 64K block erase

CE: Full Chip Erase

The erase command does not need data, directly configure the command register as the corresponding command, and then start it.

The programming command process is as follows



24.4 Register Description

24.4.1 Register List

Table 201 QFLASH controller register list

offset address name		abbreviation	describe	Defaults
0X0000_0000 command information register		CMD_INFO	Command and data words required for QFLASH operation Festival	0X0000_0000
0X0000_0004 Command start register		CMD_START command start		0X0000_0000
0X0000_0008 Flash control register Flash_CR			QFLASH operation mode control	0X0000_0000
0X0000_000C Remap Register		Remap	Remap option control	0X0000_0000
0X0000_0010 ADDR register		Flash_ADDR operation address		0X0000_0000
0X0000_0014 decryption control register		DECRYPT_CR Firmware confidential mode control		0X0000_0000
0X0000_0018 decryption status register		DECRYPT_STA firmware decryption status;		0X0000_0000
0X0000_0200 ··· 0X0000_0600	Read and write data cache	RSA_BUF	Used to cache QFLASH read and write data or Or RSA encryption and decryption data. This memory is **The QFLASH controller is shared with the RSA module, Therefore, QFLASH read and write and RSA operation** **Cannot be used at the same time, need to be implemented in the driver** **Add mutual exclusion protection**	0X0000_0000

24.4.2 Command Information Register

Table 202 QFLASH command information register

bit access		Instructions	reset value

[31]	RW	1: Indicates that CMD_START carries Address, and the lower 20 bits of the Address are stored in the CMD_START register bit, the upper 4 bits of the Address are fixed at 0, a total of 24 bits	1'b0
[30]	RW	1: Indicates that the CRM in the CMD_START register carries content	1'b0
[29]	RW	1: Instruct CMD_INFO to carry Dummy cycle	1'b0
[28:26]	RW	indicates how many Dummy cycles CMD_INFO carries after its content +1	3'b0
[25:16]	RW	indicates how many bytes of data CMD_INFO carries after its content +1	10'b0
[15]	RW	1: Indicates that CMD_INFO carries data	1'b0
[14]	RW	1: Indicates that the command filled in the current CMD_INFO command field is a read command, and the command includes RDSR, FR, QIOFR, RDPDRID, RSR, etc.	1'b0
[13]	RW	1: Indicates that the command filled in the current CMD_INFO command field is a WRSR command	1'b0
[12]	RW	1: Indicates that the command filled in the current CMD_INFO command field is a PP command	1'b0
[11]	RW	1: Indicates that the command filled in the current CMD_INFO command field is an SE command	1'b0
[10]	RW	1: Indicates that the command filled in the current CMD_INFO command field is a BE command	1'b0
[9]	RW	1: Indicates that the command filled in the current CMD_INFO command field is an HBE command	1'b0
[8]	RW	1: Indicates that the command filled in the current CMD_INFO command field is a CE command	1'b0
[7 : 0]	RW	CMD_INFO command field, fill in the QFlash command, please refer to the QFlash chip manual.	8'b0

24.4.3 Command Enable Register

Table 203 QFLASH command start register

bit access		Instructions	reset value
[31:29]	RO	Reserved	3'b0

[28]	RW	is set to 1 to start the command, after the command operation is completed, it will be cleared to 0 by hardware.	1'b0
[27: 8]	RW	Address[19:0]: If the A flag in the command register is 1, the lower 20 bits of Address are stored here. 20'b0	
[7 : 0]	RW	CRM[7:0]: If the CRM flag in the command register is 1, store the CRM content here.	8'b0

24.5 Common Commands of QFLASH

Table 204 QFALSH common commands

command name	Command word	command abbreviation
Write Enable	06H	WREN
Write Disable	04H	WURDI
Read Status	05H	RDSR
Write Status	01H	WRSR
Fast Read	0BH	FR
Quad IO Fast Read	EBA	QIOFR
Page Program	02H	PP
Sector Erase	20H	SE

For other commands, refer to the relevant manuals of QFLASH.

25 PSRAM Interface Controller

25.1 Function overview

W800 has a PSRAM controller with built-in SPI/QSPI interface, supports external PSRAM device access, and provides PSRAM read write erase operation. The highest read and write speed is 80MHz.

25.2 Main Features

ÿ Supports read and write access to external PSRAM

ÿ Configurable as SPI and QSPI

ÿ SPI/QSPI clock frequency can be configured

ÿ Support BURST INC mode access

ÿ Support PSRAM semi-sleep mode

25.3 Functional description

The full name of PSRAM is Pseudo static random access memory, which refers to pseudo-static random access memory. Comparable to traditional SRAM It has the advantages of small package, large capacity, and low cost. It is mainly used for data caching in IoT applications. The interface is mostly SPI, QSPI Wait. The interface pins mainly include clock signal SCK, chip select signal CS and 4 bidirectional data IOs. The PSRAM controller provided by W800 can It is accessed by the bus mode of PSRAM supporting SPI/QSPI interface, the maximum working clock rate is 80MHz, and the maximum capacity supports 64Mb.

25.3.1 **Pin Description**

SCLK: SPI interface clock, the SCLK period is set by PSRAM_CTRL[7:4], the minimum frequency division value that can be set is 3, and the default is AHB Clock divided by 4.

CS: SPI interface chip select signal

SIO0: SPI mode data input, QSPI mode SD[0]

SIO1: SPI mode data output, QSPI mode SD[1]

SIO2: QSPI mode SD[2]

SIO3: QSPI mode SD[3]

25.3.2 **Access Mode Settings**

The PSRAM controller supports two access modes of external PSRAM, SPI and four-wire QSPI, and the default is SPI. by setting

PSRAM_CTRL[1] configures the SPI mode.

PSRAM_CTRL[1] defaults to 0, that is, SPI mode. At this time, it takes 64 SCLK cycles to complete a write operation, and it takes 64 SCLK cycles to complete a read operation.

It takes 72 SCLK cycles.

If PSRAM works in SPI mode, when writing 1 to PSRAM_CTRL[1], the controller will send command 35H to PSRAM,

When reading PSRAM_CTRL[1] is 1, it means that the command is sent and PSRAM enters QPI mode. At this time, a write operation needs to be completed

16 SCLK cycles, 22 SCLK cycles are required to complete a read operation.

If PSRAM works in QPI mode, when writing 0 to PSRAM_CTRL[1], the controller will send command F5H to PSRAM,

When reading PSRAM_CTRL[1] is 0, it means that the command is sent and PSRAM enters SPI mode.

Setting the PSRAM working mode must be done after the initialization operation is completed, and cannot be set at the same time.

25.3.3 **PSRAM Initialization**

Before the first use, after the PSRAM is powered on and stabilized, write 1 to the register PSRAM_CTRL[0] to start the PSRAM reset initialization

Operation, that is, send 66H and 99H commands to PSRAM. By default, SPI mode is used to send, that is, 8 SCLK + tCPH + 8 are required

SCLK time. After the initialization is completed, the hardware automatically clears PSRAM_CTRL[0].

Initialization restores the PSRAM to SPI mode.

The recommended initialization process is:

ÿ Write PSRAM_CTRL[0] to 1

ÿ Wait until PSRAM_CTRL[0] is automatically cleared

(3) Reset the PSRAM controller by soft reset

ÿ Reset other required parameters of PSRAM_CTRL

25.3.4 PSRAM access method

The way to read and write PSRAM is the same as that of ordinary SRAM, that is, write/read data to the corresponding bus address.

25.3.5 BURST Function

By setting PSRAM_CTRL[2], the controller can support the BURST initiated by the AHB bus, that is, when the HBURST on the AHB bus is

When 1/3/5/7, it means that the AHB bus starts a continuous read/write address increment. At this time, in order to improve the access speed of PSRAM, the control

After reading/writing a word, the device does not pull CS high, but directly reads/writes the data of the next word.

The OVERTIMER register is used to set the maximum time for CS to be low, and the unit is the number of cycles of HCLK. Every time a BURST operation is started,

The internal counter starts counting from 0. When the counter value is greater than the set value, the controller will automatically stop after reading/writing the current word.

BURST operation, directly change CS to high level, if there is still data on the AHB bus that needs to be read/written at this time, it will be regarded as a separate WORD

conduct.

PSRAM controller does not support BURST in WRAP form, if accessing PSRAM will use WRAP BURST, please set

PSRAM_CTRL[2] is set to 0.

25.4 Register Description

25.4.1 Register List

Table 205 PSRAM controller register list

offset address name		abbreviation	describe	Defaults
0X0000_0000 Control Register		PSRAM_CTRL	PSRAM controller settings	0X0000_0000
0X0000_0004 Timeout Control Register		OVERTIMER	CS timeout control	0X0000_0000

25.4.2 Command Information Register

Table 206 PSRAM Control Setting Register

bit access		Instructions	reset value
[31:12] RO		RSV	'b0
[11]	RW HSM,	Halfsleep mode enabled 1: Enable PSRAM semi-sleep mode 0: Clear semi-sleep mode	1'b0
[10:8] RW tCPH,		CS High level minimum time setting, the unit is the number of AHB clock cycles, must be greater than 1, the specific time depends on According to the instructions in the same psram manual, it is not clear that the default value can not be modified	3'd6
[7:4]	RW SPI divider setting	It can only be configured as a value above 2, and the written value is the multiple of the frequency division	4'd4
[3]	RO	RSV	
[2]	RW INC_EN	BURST function enable 1: Support the BURST function on the AHB bus 0: does not support the BURST function	1'b0

[1]	RW QUAD	Write 1 to enable QPI mode of PSRAM, write 0 to enable SPI mode Read this flag bit to know which mode the current PSRAM is in.	1'b0
[0]	RW PSRAM reset	Writing 1 starts the reset operation of PSRAM, and it is automatically cleared after reset.	1'b0

25.4.3 Timeout Control Register

Table 207 CS timeout control register

bit access		Instructions	reset value
[11:0]	RW	timeout register setting, set the maximum time of CS low for BURST mode	12'd0

26 ADC

26.1 ADC Functional Overview

The acquisition module based on Sigma-Delta ADC completes the acquisition of up to 4 channels of analog signals, and the sampling rate is controlled by an external input clock.

It can collect input voltage and chip temperature, and supports input calibration and temperature compensation calibration.

26.2 ADC Main Features

ÿ Support up to 4 channels of data acquisition

ÿ Support DMA module for data buffering;

ÿ Support interrupt interactive mode;

ÿ Supports the comparison function between collected data and input data

ÿ Maximum sampling frequency 1KHz;

ÿ Support single-ended input mode and differential input mode;

26.3 ADC Functional Description

26.3.1 Basic module structure

The ADC module is a PGA+SDADC structure. After the input signal is pre-amplified by the PGA, it is input to the SDADC for processing. The SDADC design

It is 16bit ADC. It should be noted that the on-chip ADC is powered by 2.5V LDO. In order to protect the internal circuit, the input signal must be fully

Vinmax < 2.5V, other restrictions, see section 26.3.8 for details.

26.3.2 Channel Selection

Register 0x04[11:8] provides the channel selection function. The ADC of W800 provides the acquisition function of 4-channel signals. By setting the register

0x04[11:8] can select any of the 4 channels for single-ended signal data acquisition, or select 2 pairs of differential pairs that are pre-paired among the 4 channels

Separate channels for differential signal data acquisition. For specific channel selection, please refer to the register description.

In addition to 4 channels, the ADC module also provides temperature detection, voltage detection and offset calibration functions. Corresponding to the channel selection respectively

4'b1100ÿ4'b1101ÿ4'b1110.

If you need to use the corresponding function, configure the channel selection as the corresponding channel, and then follow the procedure described above.

26.3.3 Data Comparison

The ADC module provides the data comparison function, and the switch of this function can be set through BIIT[5] of the configuration register 0x10.

The user can configure the value to be compared through BIT[17:0] of the result register 0x18. This value needs to be converted to 18bit signed number, the most

The high bit is the sign bit. BIT[6] of ADC configuration register 0x10 can set the direction of data comparison. When this bit is 0, if ADC

If the collected value is greater than or equal to Comp_data, INT_CMP will be set to 1, and an interrupt will be generated; when this bit is 1, only

When the value collected by ADC is less than Comp_data, INT_CMP will be set to 1 and an interrupt will be generated.

26.3.4 PGA Gain Adjustment Description

BIT[8:4] of register address 0X08, 5-bit gain control signal, where

0X08[8:7]=GAIN_CTRL_PGA<4:3> (corresponding to GAIN2)

0X08[6:4]=GAIN_CTRL_PGA<2:0> (corresponding to GAIN1, 110 and 111 are not used)

PGA overall gain GAIN_PGA=GAN1*GAIN2

PGA Gain Configuration Table (expressed in magnification)

[6:4] \ [8:7]	00	01	10	11
000	1	2	3	4
001	16	32	48	64
010	32	64	96	128
011	64	128	192	256
100	128	256	384	512
101	256	512	768	1024

According to the simulation results, for the same magnification, it is recommended to use the configuration with a larger GAIN2 magnification first.

26.3.5 VCMIN Adjustment Description for Single-Ended Mode

The internal VCMIN voltage of PGA, in single-ended mode, is used as the input negative terminal of the op amp.

The purpose of adjusting VCMIN is to adapt to various signal sources with different bias points, and to prevent the PGA output from being saturated at certain bias points that are too high or too low.

and.

BIT[16:11] of register address 0X40000D20, the default is 100000, that is, VCMIN default=1.311V, stepÿ39.1mV

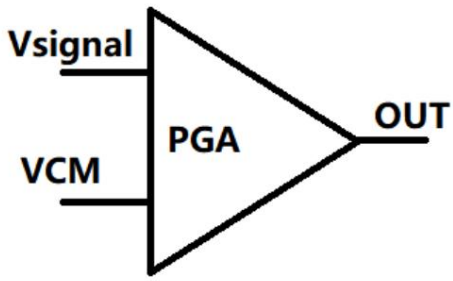


Figure 37 Principle of single-ended working mode

26.3.6 Bypass mode

Bypass_pga register address 0X08 BIT[3], default 0, no bypass, PGA works normally; if it is 1, PGA is bypassed at this time,

The external signal source is directly connected to the SDADC input terminal, which is mainly used for internal module testing, and no bypass is required for normal use.

Bypass_ref Register address 0X08 BIT[2], default 0, no bypass, SDADC works normally; if it is 1, SDADC internal

The internal ref is bypassed, and an external reference voltage needs to be added through the test pin, which is mainly used for internal module testing, and bypass is not required for normal use.

26.3.7 Supplementary Description of SDADC Output Code

The SDADC output result is stored in the ADC_RESULT register, register address 0X00 BIT[17:0].

The output code of SDADC itself is an unsigned number, because the number has processed the output code of SDADC (the highest bit is reversed), it becomes signed

number, and the 0 code of the SDADC output code is not fixed (and not exactly 10 0000 0000 0000 0000), so directly use the register

It is wrong to use the value of the device for subsequent calculations.

When calculating the output result of SDADC, the value of the register ADC_RESULT[17:0] (the read value is a hexadecimal number), the software needs to first

Convert back to an unsigned number (invert the highest bit of the binary number ADC_RESULT[17]), and then perform subsequent calculations.

The SDADC designed this time is a 12bit ADC. The ADC conversion data is processed in the design, and the actual effective number of registers is 16bit.

The register data is 18bit, and the lowest two bits are added through the digital filter, and the 16bit number is converted into an 18bit number (18bit numberÿ16bit number*4),

So the effective number of digits is the high 16 bits (that is, ADC_RESULT[17:2]).

After converting the read value back to an unsigned number, the calculation involving LSB also requires software to convert 18bit data into 16bit data, that is, discard the last

The lower two bits of data (ADC_RESULT[1:0]), only retain the upper 16bit (turn back to ADC_RESULT[17:2] of the unsigned number), and then use 16bit

The data is used for subsequent calculations. At this time, the simulation result of LSB is approximately equal to 68.5uV (it needs to be tested to determine the actual LSB of the chip).

26.3.8 Input Signal Voltage Range

Because the NTO version PGA and SDADC are powered by 2.5V LDO, in order to prevent internal circuit saturation, the input signal voltage range is limited

system.

Assuming that the input differential signals are Vinp and Vinn (in single-ended mode, the signal Vinn is VCMIN), then the input differential voltage Vdiff=Vinp

Vinn, input common mode voltage Vcm=(Vinp+Vinn)/2. PGA first-stage gain GAIN1, second-stage gain GAIN2, PGA overall gain

Yield GAIN_PGA=GAN1*GAIN2 (see Section 2.2.2 for details).

The input signal must meet the following constraints at the same time:

0V<Vinp<2.5Vÿ0V<Vinn<2.5Vÿ

0V<Vcm-(Vdiff*GAIN1)/2ÿ

Vcm+(Vdiff*GAIN1)/2<2.5Vÿ

0V<1.2V-(Vdiff*GAIN_PGA)/2ÿ

1.2V+(Vdiff*GAIN_PGA)/2<2.5Vÿ

If the approximate range of the input signal is known, it can be directly substituted into the above formula to calculate and select the available gain setting.

For unknown input signals, you can first use the x1 gain setting to determine the approximate range, and then substitute into the above formula for calculation to select the available gain setting.

26.4 Register Configuration for Typical Use Cases

The following configuration words are all hexadecimal numbers, and the high-order 0 has been omitted.

26.4.1 Offset measurement

1. Set the working state of the module, set the register 0X04 to 0xE03, open the channel selection (offset detection) / SDADC_CHOP enable

enable/SDADC enable/LDO enable

2. Set module register 0X08=0x3, PGA is x1 gain configuration/PGA_CHOP enable/PGA enable. Among them, the gain configuration

0X08 BIT[8:4] should be modified according to the requirements of the test (that is, what gain configuration is used for the subsequent test, offset measurement

also use the same gain configuration when

3. Set the clock status 0X40000E14 BIT[15:8]=0x28, and confirm that the SDADC clock configuration is 1MHz. The default value after the chip is powered on

It is 0x28, if there is no change, this step can be omitted

4. Wait for 2ms, read the output result of the ADC_RESULT register, refer to Section 26.3.7, the software needs to convert back to unsigned number (most

The high bit ADC_RESULT[17] is negated), and then the lowest two bits are discarded, and only the high 16bit data is kept. Repeat the reading a total of 10 times, the reading interval is 2ms,

Calculate the average number of codes and record it as code_offset

26.4.2 Differential Input Mode

1. First measure the offset according to Section 26.4.1 to get the code_offset

2. Set module register 0X04=0x803, channel selection (positive channel 0, negative channel 1)/SDADC_CHOP enable

/SDADC enable/LDO enable. The channel selection 0X04[11:8] can be configured as differential input of other channels, refer to 26.3.2

Festival;

3. Set module register 0X08=0x3, PGA is x1 gain configuration/PGA_CHOP enable/PGA enable. Among them, the gain configuration

0X08[8:4] Select the appropriate configuration according to the input signal voltage range, refer to section 26.3.4 and 26.3.8

4. Set the clock status 0X40000E14[15:8]=0x28, and confirm that the SDADC clock configuration is 1MHz. After the chip is powered on, the default value is

28. If there is no change, this step can be omitted

5. Wait for 2ms, read the output result of ADC_RESULT register, register address 0X00[17:0]. Referring to Section 26.3.7, requires

The software first converts back to an unsigned number (the highest bit ADC_RESULT[17] is inverted), and then discards the lowest two bits, and only keeps the high 16bit data. 6.

Repeat reading a total of 10 times, the reading interval is 2ms, and the average value of the calculated code number is recorded as code_out

7. The differential input signal Vindiff=(code_out-code_offset)*LSB/GAIN_PGA, where LSB is approximately equal to

68.5uV

26.4.3 SINGLE-ENDED INPUT MODE

1. First measure the offset according to section 26.4.1 to get code_offset;

2. Set module register 0X04=0x3, channel selection (positive channel 0, negative VCMIN)/SDADC_CHOP enable/SDADC

Enable/LDO enable. The channel selection 0X04[11:8] can be configured as single-ended input of other channels, refer to section 26.3.2

Set module register 0X408=0x3, PGA is x1 gain configuration/PGA_CHOP enable/PGA enable. Among them, VCMIN configuration

0X40000D20[16:11] and gain configuration 0X08[8:4], select the appropriate configuration according to the input signal voltage range, refer to 26.3.4,

Sections 26.3.5 and 26.3.8

3. Set the clock status 0X40000E14[15:8]=0x28, and confirm that the SDADC clock configuration is 1MHz. After the chip is powered on, the default value is

28. If there is no change, this step can be omitted;

4. Wait for 2ms, read the output result of ADC_RESULT register, register address 0X00[17:0]. Referring to Section 26.3.7, requires

The software first converts back to an unsigned number (the highest bit ADC_RESULT[17] is inverted), and then discards the lowest two bits, and only keeps the high 16bit data. repeat

Read a total of 10 times, the reading interval is 2ms, and the average value of the calculated code number is recorded as code_out;

5. Single-ended mode input signal vin_single=(code_out-code_offset)*LSB/ GAIN_PGA+VCMIN, where LSB is based on

The simulation result is approximately equal to 68.5uV, and the default value of VCMIN is 1.311V;

26.4.4 Temperature Detection Mode

1. Set module register 0X04=0xC03, channel selection (temperature detection)/SDADC_CHOP enable/SDADC enable/LDO enable

can;

2. Set module register 0X08=0x183, PGA is x4 gain configuration/PGA_CHOP enable/PGA enable;

3. Set the clock status 0X40000E14[15:8]=0x28, and confirm that the SDADC clock configuration is 1MHz. After the chip is powered on, the default value is

28. If there is no change, this step can be omitted;

4. Set the working status of the tempsensor module to 0X0C=0x1, enable TempSensor/cal_offset_temp12=0,

TempSensor configured for x2 gain;

5. Wait for 2ms, read the output result of the ADC_RESULT register, refer to section 26.3.7, the software needs to convert back to unsigned number (most

The high bit ADC_RESULT[17] is inverted), recorded as the output code code_out1;

6. Set the working status of the tempsensor module to 0X0C=0x3, enable TempSensor/cal_offset_temp12=1,

TempSensor is configured for x2 gain;

7. Wait for 2ms, read the output result of the ADC_RESULT register, refer to Section 26.3.7, the software needs to convert back to unsigned number (most

The high bit ADC_RESULT[17] is reversed), recorded as the output code code_out2;

8. Repeat steps 4 to 7 for a total of 10 times to calculate the average values of code_out1 and code_out2 code_out1avg and

code_out2avgÿ

9. Calculate Vtemp=(code_out1avg-code_out2avg)/16 from the above steps, and TempSensor output voltage=Vtemp

Number of codes * (LSB/4), where LSB is approximately equal to 68.5uV according to the simulation result (this LSB is the simulation result of 16bit, the temperature detection above

In steps 5 and 7, the read is 18bit data, 18bit number ÿ 16bit number * 4, then the least significant bit corresponding to 18bit ÿ LSB/4)

10. According to the test results, a fitting formula has been sorted out, which can directly estimate the chip temperature

Vtemp code=(15.548*chip temperature)+4444.1

Substitute the Vtemp calculated in step 9 into the above formula to get the chip temperature. (Note that the fitting formula is in decimal)

In addition, add an optional measure to improve accuracy:

For different chips, assuming that the coefficient k in the fitting formula y=kx+b remains unchanged (both are 15.548), the coefficient b of different chips has certain differences.

According to the instructions above, first calculate the Vtemp code number at a known ambient temperature (for example, room temperature 25°C), and substitute it into the Vtemp code

=(15.548*(known ambient temperature+11.8))+b, get the coefficient b of different chips, save it in rom or flash, replace the above formula

4444.1. Then, the temperature is calculated according to the coefficients of the new formula, which can improve the temperature detection accuracy of different chips to a certain extent.

26.4.5 Voltage Detection Mode

1. First measure the offset according to section 26.4.1 to get code_offset;

2. Set module register 0X04=0xD03, channel selection (voltage detection)/SDADC_CHOP enable/SDADC enable/LDO enable

can

3. Set module register 0X08=0x83, PGA is x2 gain configuration/PGA_CHOP enable/PGA enable.

4. Set the clock status 0X40000E14[15:8]=28, and confirm that the SDADC clock configuration is 1MHz. The default value is 28 after the chip is powered on.

If there is no change, this step can be omitted

5. Wait for 2ms, read the output result of the ADC_RESULT register, refer to Section 26.3.7, the software needs to convert back to unsigned number (most

The high bit ADC_RESULT[17] is negated), and then the lowest two bits are discarded, and only the high 16bit data is kept. Repeat the reading a total of 10 times, the reading interval is 2ms,

Calculate the average value of the number of codes and record it as code_out;

6. Chip power supply voltage Vpower=(code_out-code_offset)*LSB/2+1.2V, where LSB is approximately equal to

68.5uV

26.5 ADC Register Description

26.5.1 Register List

Table 208 ADC register list

offset address name		abbreviation	describe	reset value
0X0000	ADC result register	ADC_RESULT	Store ADC acquisition value and data comparison value	0X0000_0000
0X0004	ADC Analog Registers	ADC_ANA_CTRL	Configure ADC related functions	0X0000_0004
0X0008	PGA configuration register	PGA_CTRL	Configure PGA related functions;	0x0000_0000
0X000c	TempSensor Configuration Registration device	TEMP_CTRL	Configure temperature sensor related functions; 0x0000_0000	
0x0010	ADC configuration register	ADC_CTRL	Configure ADC module functions;	0x0050_0500
0x0014	ADC Interrupt Register	ADC_INT_STATUS	ADC module interrupt status register; 0x0000_0000	
0x0018	compare value register	CMP_VALUE	Compare Threshold Settings	0x0000_0000

26.5.2 ADC Result Register

Table 209 ADC result register

bit access		Instructions	reset value
[17:0]	RW	ADC conversion result value, the effective bit width is 18bits. Signed number, the most significant bit is the sign bit.	1'b0

26.5.3 ADC Analog Configuration Register

Table 210 ADC configuration register

Bit	Access	Operation Instructions	reset value
[11 : 8]	RW	ADC working channel selection signal: 4'b0000: AIN0 channel works. In DMA mode, corresponding to DMA channel 0 work 4'b0001: AIN1 channel works. In DMA mode, corresponding to DMA channel 1 work 4'b0010: AIN2 channel works. In DMA mode, corresponding to DMA channel 2 work 4'b0011: AIN3 channel works. In DMA mode, corresponding to DMA channel 3 work 4'b0100ÿRSV 4'b0101ÿRSV 4'b0110ÿRSV 4'b0111ÿRSV 4'b1000: AIN0/AIN1 differential signal input. In DMA mode, corresponding to DMA channel 0 work 4'b1001: AIN2/AIN3 differential signal input. In DMA mode, corresponding to DMA channel 2 work 4'b1010ÿRSV 4'b1011ÿRSV 4'b1100: Temperature sensor input, corresponding to DMA channel 2 4'b1101: Voltage detection module input, corresponding to DMA channel 3 4'b1110: offset detection input; 4'b1111ÿRSV	4'b1000
[7]	RO	RSV	1'b0
[6:5]	RW	chop_enr	2'b00

		LDO chopping signal PD signal	
[4]	RW	chop_ens sdadc chopping signal PD signal 0: enable 1: disable	1'b0
[3]	RO	RSV	1'b0
[2]	RW	Pd_sdadc sdadc analog module power-down enable 1: power down 0: working	1'b1
[1]	RW	Rstn_sdadc Analog module digital logic part reset signal 0: reset 1: normal work	1'b0
[0]	RW	en_ldo_sdadc ADC LDO enable 0: power down 1: work	1'b0

26.5.4 PGA Configuration Register

Table 211 PGA configuration register

Bit	Access	Operation Instructions	reset value
[8:4]	RW	Gain_ctrl_pga PGA gain configuration; BIT[8:7] configure GAIN2, BIT[6:4] configure GAIN1, specific gain table refer to Section26.3.4	5'd0
[3]	RW	Bypass_pga pga bypass signal 1: bypass pga 0: no bypass	1'b0
[2]	RW	Bypass_ref Internal Reference Voltage Bypass Signal 1: Bypass internal reference voltage 0: no bypass	1'b0
[1]	RW	Chop_enp PGA chop enable signal 1: enable 0: disable	1'b0
[0]	RW	One_because Pga enable signal 1: enable 0: disable	1'b0

26.5.5 TEMP Configuration Register

Table 212 Temperature Sensor Configuration Register

Bit Access Operation Instructions			reset value
[5:4] RW		Gain_temp temp gain control Coding Gain 00 -- 2 01 -- 4 10 -- 6 11 -- 8	2'd0
[3ÿ2] RO		RSV	2'b0
[1]	RW	Cal_offset_temp12 TEMPSEN offset calibration function	1'b0
[0]	RW	ON_TEMP 1: temp enable 0: temp disabled	1'b0

26.5.6 ADC Function Configuration Register

Table 213 Temperature Sensor Configuration Register

Bit Access Operation Instructions			reset value
[29:20] RW		ana_swth_time After the software switches the data channel, the time required for the analog circuit to remain stable, the default is 80 pclks, which is 2us	10'h050

[19ÿ18] RO		RSV	2'b0
[17ÿ8] RW		ana_heat_time After the software starts adc_start, the time required for the analog circuit to remain stable, the default is 80 pclks, which is 2us	10'h050
[7]	RO	RSV	1'b0
[6]	RW	Cmp_pol 0: Interrupt when adc_result >=cmp_value 1: Interrupt when adc_result<cmp_value	1'b0
[5]	RW	Cmp_int_ena 1: compare interrupt enable 0: compare interrupt is not enabled	1'b0
[4]	RW	Adc_cmp_enable 1: ADC compare function enable 0: adc compare function disabled	1'b0
[3:2]	RO	Reserved	2'b0
[1]	RW	Adc_int_ena 1: ADC data conversion interrupt enable 0: ADC data conversion interrupt is not enabled	1'b0
[0]	RW	Adc_dma_enable 1: dma enable 0: DMA disabled	1'b0

26.5.7 ADC Interrupt Status Register

Table 214 ADC Interrupt Status Register

Bit	Access	Operation Instructions	reset value
[1]	RW	Cmp_int Compare interrupt flag bit, hardware set, software write 1 to clear	1'b0
[0]	RW	Adc_int Data conversion complete interrupt, hardware set, software write 1 to clear	1'b0

26.5.8 Compare Threshold Register

Table 215 Compare Threshold Register

Bit	Access	Operation Instructions	reset value
[17:0]	R/W	Cmp_value the value to compare	18'h00000

27 Touch Sensor

27.1 Overview of module functions

The basic functions of the module are as follows:

ÿ Support up to 16 channels of Touch Sensor scanning;

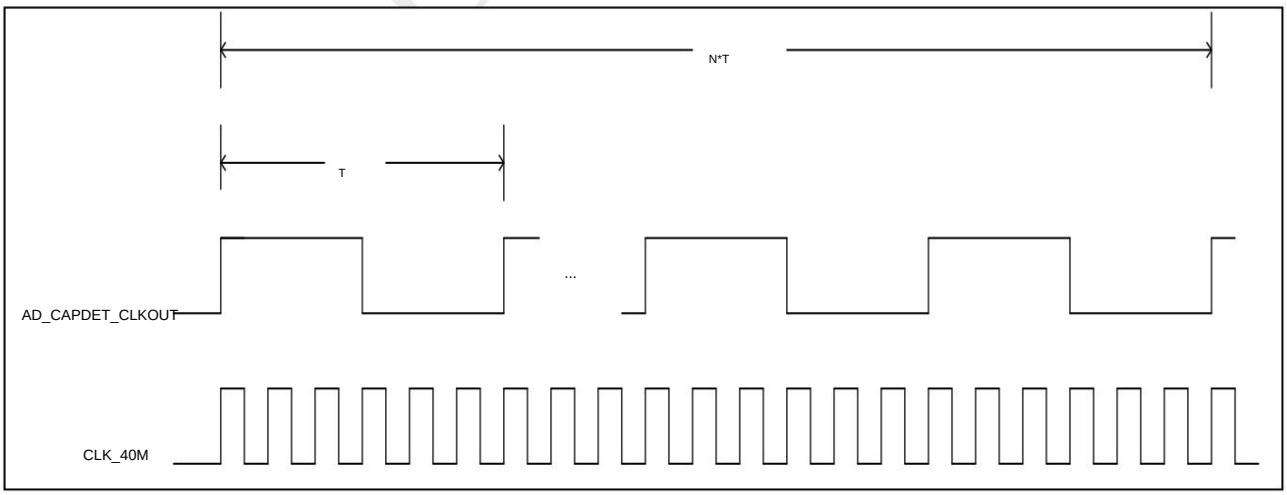
ÿ Record the scanning results of each Touch Sensor;

ÿ Report scan results through interrupts;

27.2 Function instructions

The touch button module is integrated in the W800 system. The basic principle is that when the button is touched, the capacitance value of the circuit on the button will change, thereby Affects the output clock frequency in the block. By detecting the frequency of the output clock of the module, it can be judged whether the capacitance value has changed, thereby judging The state of the key.

The basic function of this digital module is to scan the state of each touch button successively at regular intervals, count and record in the set time window The status of each key is displayed. If it exceeds the set threshold, it is judged that the button is touched and reported to the MCU system through an interrupt.



27.2.1 Basic workflow

1. Set the Touch_CR register, configure the scan cycle, scan window, and select the IO to be scanned. Touch_CR in

scan_period is the interval period of each scan, and the unit is 16ms. That is, if the register is set to 10, every 160ms will successively

Scan 16 touch keys. CAPDET_CNT is the window counted when scanning each IO state, that is, N in the above figure. It should be noted that for

To avoid the jitter caused by switching channels, the counting will start at the third pulse after each switching scan IO, so if this register is set

is N, the actual counting window is N-2.

2. Set the counting threshold corresponding to each Touch IO. If the result of the scan exceeds the base number + threshold, the button can be considered to be touched;

3. Enable Touch_CR[0], the module starts to work.

4. After the touch key module is enabled, the hardware will first scan the selected IOs one by one, and store the count value of each IO as the base.

Then scan the selected IO every other scan_period, and store the currently scanned value. After all IO scans are completed, the

The count value of each IO is compared with the base number. If the current value > base number + threshold, the button is considered to be touched. Register 0x44

The corresponding bit in PAD_STATUS will be set to 1 and reported to the system through an interrupt. PAD_STAUS Write 1 to clear 0.

27.3 Register list:

Table 216 Touch Sensor controller register list

offset address name		abbreviation	describe	Defaults
0X0000_0000 Control Register		Touch_CR	Touch Sensor controller settings	0X0000_0000

0X0000_0004 ~ 0X0000_0040	Touch button single control Touch_Sensor x		Threshold control and count value of each touch key 0X0000_0000	
0x0000_0044 interrupt controller		Int_Source	interrupt controller	0x0000_0000

27.3.1 Touch Sensor Control Register

Table 217 Touch Sensor Control Setting Register

bit access	Instructions	reset value
[31:26] RW Scan_period	Scan period, the unit is 16ms; for example, if scan_period is set to 6'd10, then the button will be scanned every 160ms key state;	6'd10
[25:20] RW CAPDET_CNT	Select the number of CAPDET output pulses as the counting window. Note: In order to avoid the jitter caused by switching channels, counting will start at the beginning of the third pulse, so this register setting If it is set to N, the actual counting window is N-2. For example, if it is set to 6'd20, the period of 18 CAPDET pulses is the counting window.	6'd20
[19:4] RW Touch Sensor button selection; scan the touch button status of the corresponding bit.	0x0000: do not scan; 0x0001: scan the first button; 0x0002: scan the second button;	16'd0

		0x0003: Scan the first and second keys; … 0xFFFF: scan all 16 keys;	
[3:1]	RW RSV		3'd0
[0]	RW touch button controller enable 1: Enable touch key scanning; 0: Disable touch button scanning;		1'b0

27.3.2 Touch key single channel control register

Table 218 Touch key single channel setting register

bit access		Instructions	reset value
[22:8] RO		Touch Sensor 1 count value	14'd0
[7]	RO	RSV	1'b0
[6:0]	RW Touch Sensor 1 Threshold		7'd50

27.3.3 Interrupt Control Register

Table 219 Touch key interrupt control register

bit access		Instructions	reset value
[31:16]	RW INT_EN When the corresponding bit is 1, it means that the corresponding IO is triggered and an interrupt will be generated; When the corresponding bit is 1, it means that the corresponding IO is triggered and no interrupt will be generated;		16'd0

[15:0]	RW	PAD_STATUS/INT_SOURCE When the bit is 1, it means that the corresponding PAD is triggered; When the bit is 0, it means that the corresponding PAD is not triggered; write 1 clear 0	16'd0

28 W800 security architecture design

28.1 Function overview

XT804 indicates the security feature of bus access through the HPROT signal

	0	1
HPRT[3]	uncacheable	cacheable
HPRT[2]	un-security	security
HPRT[1]	User	super
HPRT[0]	Code	data

W800's support for security architecture is divided into security access control for sram and access control for peripherals.

28.1.1 SRAM Secure Access Controller (SASC)

The first-level bus SRAM, the second-level bus SRAM and FLASH each have a security access controller, and the storage space controlled by each SASC have the following safety features

ÿ 8 configurable safe zones

ÿ Each security area can be configured with a minimum of 4 bytes

ÿ Security-super with unlimited access to all areas

ÿ If the CPU's access on the AHB bus is rejected, it will generate a false response signal (HRESP=1), from An access exception is thrown.

ÿ No exception will be generated if other master devices on the bus, such as DMA, are denied access.

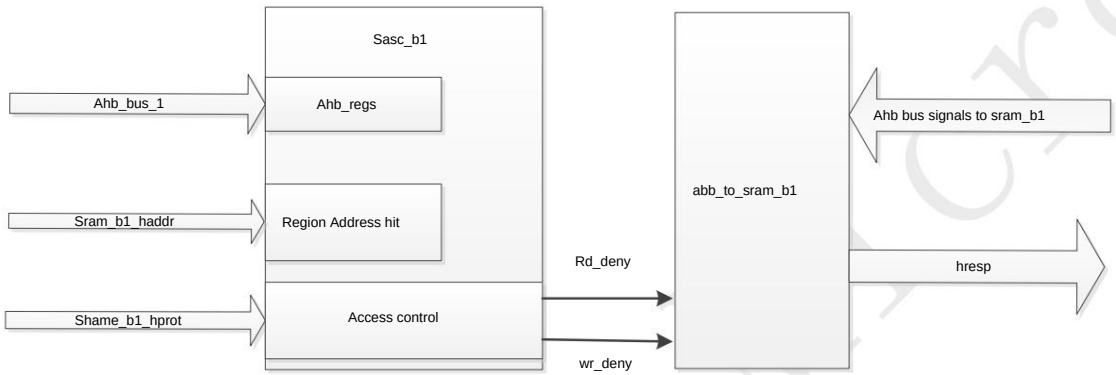
28.1.2 Trusted IP Controller (TIPC)

Hanging on the APB bus, it is used to configure the trust authority of all peripherals including the first-level bus, the second-level bus and the APB bus. If the peripheral

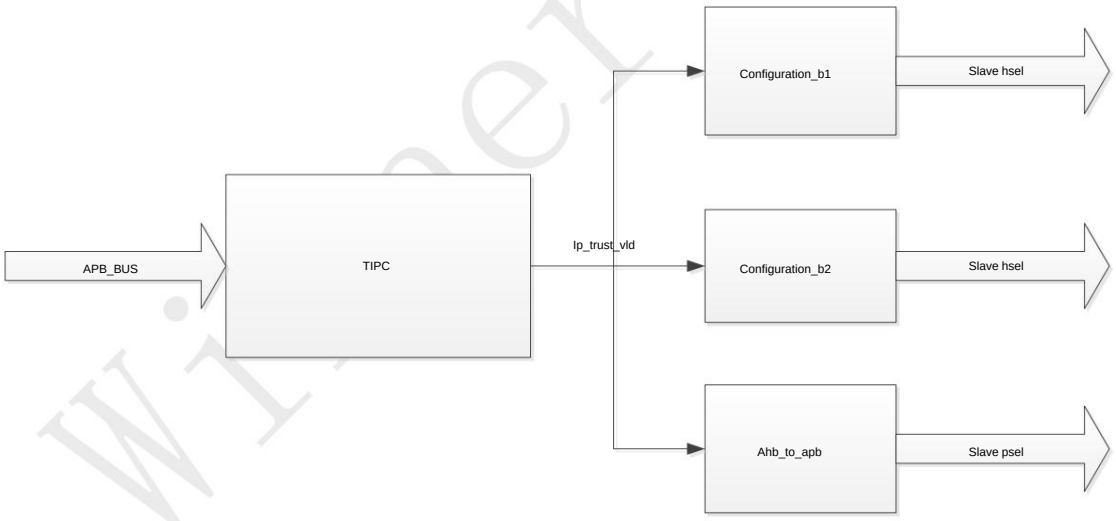
Configured as a trusted device (ip_trust_vld=1), only access on the bus is security (HPROT[2]=1 or PPROT[2]=1)

Only then can the peripherals be accessed normally, otherwise reading and writing will be rejected.

28.2 Security Architecture Block Diagram



Functional block diagram of Sasc_b1



TIPC functional block diagram

28.3 Register Description

28.3.1 SASC Register List

name offset	address	bit threshold	illustrate	reset value
CAR	0x0	[31:16] Leave unused		
		[15:14] sram region7 configuration properties, same as region0 2'b00		
		[13:12] sram region6 configuration properties, same as region0 2'b00		
		[11:10] sram region5 configuration attributes, same as region0 2'b00		
		[9:8] sram region4 configuration attributes, same as region0 2'b00		
		[7:6]	sram region3 configuration attribute, same as region0 2'b00	
		[5:4] sram region2 configuration attributes, same as region0 2'b00		
		[3:2] sram region1 configuration attributes, same as region0 2'b00		
		[1:0]	sram region0 configuration properties 00: un-security-user 01: un-security-super 10: security-user 11: security-super	2'b00
CR	0x4	31:8 unused reserved		
		7	region7 attribute control register, same as region0	
		6	region6 attribute control register, same as region0	
		5	region5 attribute control register, same as region0	
		4	region4 attribute control register, same as region0	
		3	region3 attribute control register, same as region0	

		2	region2 attribute control register, same as region0	
		1	region1 attribute control register, same as region0	
		0	region0 attribute control register 0: attribute is invalid 1: attribute is valid	
AP0	0x8	[31:16] Leave unused		
		[15:14] region7 AP configuration, suitable for un-security user		
		[13:12] region6 AP configuration, suitable for un-security user		
		[11:10] region5 AP configuration, suitable for un-security user		
		[9:8]	region4 AP configuration, suitable for un-security user	
		[7:6]	region3 AP configuration, suitable for un-security user	
		[5:4]	region2 AP configuration, suitable for un-security user	
		[3:2]	region1 AP configuration, suitable for un-security user	
		[1:0]	region0 AP configuration, suitable for un-security user 00: R/W 01: RO 10: WO 11: No	

			Access	
CD0	0xC	[31:16] Leave unused		
		[15:14] region7 CD configuration for un-security user		
		[13:12] region6 CD configuration, suitable for un-security user		
		[11:10] region5 CD configuration, suitable for un-security user		
		[9:8]	region4 CD configuration, suitable for un-security user	
		[7:6]	region3 CD configuration, suitable for un-security user	
		[5:4]	region2 CD configuration, suitable for un-security user	
		[3:2]	region1 CD configuration, suitable for un-security user	
		[1:0]	region0 CD configuration, suitable for un-security user 00: Date Access , Opcode Close 01: Data Access 10: Opcode Close 11: Data, Opcode all deny	

AP1	0x10	[31:16] Leave unused		
		[15:14] region7 AP configuration, suitable for un-security super		
		[13:12] region6 AP configuration, suitable for un-security super		
		[11:10] region5 AP configuration, suitable for un-security super		
		[9:8]	region4 AP configuration, suitable for un-security super	
		[7:6]	region3 AP configuration, suitable for un-security super	
		[5:4]	region2 AP configuration, suitable for un-security super	
		[3:2]	region1 AP configuration, suitable for un-security super	
		[1:0]	region0 AP configuration, suitable for un-security super 00: R/W 01: RO 10: WO 11: No Access	
CD1	0x14	[31:16] Leave unused		
		[15:14] region7 CD configuration for un-security super		

		[13:12]	region6 CD configuration, suitable for un-security super	
		[11:10]	region5 CD configuration, suitable for un-security super	
		[9:8]	region4 CD configuration, suitable for un-security super	
		[7:6]	region3 CD configuration, suitable for un-security super	
		[5:4]	region2 CD configuration, suitable for un-security super	
		[3:2]	region1 CD configuration, suitable for un-security super	
		[1:0]	region0 CD configuration, suitable for un-security super 00: Date Access , Opcode Close 01: Data Access 10: Opcode Close 11: Data, Opcode all deny	
AP2	0x18	[31:16]	Leave unused	
		[15:14]	region7 AP configuration, for security-user	
		[13:12]	region6 AP configuration, for security-user	
		[11:10]	region5 AP configuration, for security-user	

		[9:8]	region4 AP configuration, for security-user	
		[7:6]	region3 AP configuration, suitable for security-user	
		[5:4]	region2 AP configuration, for security-user	
		[3:2]	region1 AP configuration, for security-user	
		[1:0]	region0 AP configuration, for security-user 00: R/W 01: RO 10: WO 11: No Access	
CD2	0x1C	[31:16] Leave unused		
		[15:14] region7 CD configuration, for security-user		
		[13:12] region6 CD configuration, for security-user		
		[11:10] region5 CD configuration, for security-user		
		[9:8]	region4 CD configuration, for security-user	
		[7:6]	region3 CD configuration, for security-user	
		[5:4]	region2 CD configuration, for security-user	
		[3:2]	region1 CD configuration, for security-user	
		[1:0]	region0 CD configuration, for security-user 00: Date Access , Opcode Close 01: Data Access 10: Opcode Close 11: Data, Opcode all deny	
REGION0 0x20		31:n+1 Reserved		0
		n:8	BAddr0, base address of region0, n and sram are large	

			small related, sram=32kBÿn=20 sram=64kB, n=21	
		7:5 reserved		3'b000
		4:0	RSizeÿregion0 size 00101: 4B 00110: 8B …… 10001: 16KB 10010: 32KB ……	
REGION1 0x24		31:n+1 Reserved		0
		n:8	Baddr1, region1 base address	
		7:5 reserved		3'b000
		4:0	region1 size	
REGION2 0x28		31:n+1 Reserved		0
		n:8	region2 base address	
		7:5 reserved		3'b000
		4:0	region2 size	
REGION3 0x2C		31:n+1 Reserved		0
		n:8	region3 base address	
		7:5 reserved		3'b000

		4:0	region3 size	
REGION4 0x30		31:n+1 Reserved		0
		n:8	region4 base address	
		7:5 reserved		3'b000
		4:0	region4 size	
REGION5 0x34		31:n+1 Reserved		0
		n:8	region5 base address	
		7:5 reserved		3'b000
		4:0	region5 size	
REGION6 0x38		31:n+1 Reserved		0
		n:8	region6 base address	
		7:5 reserved		3'b000
		4:0	region6 size	
REGION7 0x3C		31:n+1 Reserved		0
		n:8	region7 base address	
		7:5 reserved		3'b000
		4:0	region7 size	

28.3.2 TIPC Register

name offset	address	bit threshold	illustrate	reset value
ip_trust_vld0 0x0		31:18 unused		0
		17	BT modem trusted attributes 1: trusted 0: not trusted	0
		16	I2S Trusted Attributes	0
		15	PWM Trusted Attributes	0
		14	LCD driver trusted attributes	0
		13	RF controller trusted attributes	0
		12	timer trusted attribute	0
		11	watch dog trusted attributes	0
		10	PORTB trusted attribute	0
		9	PORTA trusted attribute	0
		8	UART5 Trusted Attributes	0
		7	UART4 Trusted Attributes	0
		6	UART3 Trusted Attributes	0
		5	UART2 Trusted Attributes	0
		4	UART1 Trusted Attributes	0
		3	UART0 Trusted Attributes	0

		2	SPI MASTER TRUSTED ATTRIBUTES	0
		1	SAR ADC Trusted Attributes	0
		0	I2C Trusted Attributes	0
ip_trust_vld1 0x4		31:18 Unused		0
		17	RF BIST trusted attribute configuration 1: trusted 0: not trusted	0
		16	SDIO Wrapper Trusted Attributes	0
		15	SPI_HS trusted attributes	0
		14	SDIO Trusted Attributes	0
		13	unused	0
		12	SEC Credible Attributes	0
		11	MAC Trusted Attributes	0
		10	BBP Trusted Attributes	0
		9	MMU trusted attributes	0
		8	Clock reset control module trusted attribute 0	
		7	PMU Trusted Attributes	0
		6	BT Trusted Attributes	0
		5	GPSEC Trusted Attributes	0
		4	DMA Trusted Attributes	0
		3	RSA Trusted Attributes	0
		2	PSRAM Controller Trusted Attributes	0
		1	FLASH Controller Trusted Attributes	0

		0	SDIO HOST Trusted Attributes	0

28.4 Instructions for use

28.4.1 Memory Safe Access (SASC)

28.4.1.1 Register Access Rights

Access to SASC registers follows the following principles:

ÿ The CAR register can only be read and written when the cpu is security-super.

ÿ When the cpu is in un-security-super, it can access the region-related registers configured as un-security-user

device position.

ÿ When the cpu is security-super, all registers can be read and written.

ÿ Registers are not accessible in other cases

For example, when CAR is set to 16'he4e4, it means that region0 and region4 are set as un-security-user, and if cpu

In un-security-super, the cpu can read and write registers REGION0 and REGION4, as well as CR[0], CR[4], APx[1:0],

APx[9:8], CDx[1:0], CDx[9:8]ÿ

28.4.1.2 Setting of Protection Interval Address

Each SASC supports 8 configurable memory protection interval regions, and the base address in REGIONx[31:8] is the one you want to configure

25 to 2 bits of the actual physical address of the storage area. At the same time, the size and base address of Size need to meet the following requirements:

SIZE

00101: 4B;

00110: 8B; Baddrx[0] = 0

00111: 16B; Baddrx[1:0] = 0

01000: 32B; Baddrx[2:0] = 0

01001: 64B; Baddrx[3:0] = 0

01010: 128B; Baddrx[4:0] = 0

01011: 256B; Baddrx[5:0] = 0

01100: 512B; Baddrx[6:0] = 0

01101:1KB; Baddrx[7:0] = 0

01110: 2KB; Baddrx[8:0] = 0

01111:4KB; Baddrx[9:0] = 0

10000: 8KB; Baddrx[10:0] = 0

10001: 16KB; Baddrx[11:0] = 0

For example, if 20000100 is used as the base address of region0, then region0 can be set to a maximum of 256B. If you want to set

The region0 is set to 128B, and the REGION0 register should be filled with 0x0000400a. If 20040000 is used as the base address, the region

The maximum can be set to 64KB. If a 64KB region is to be defined, this register should be filled with 0x01000013.

28.4.1.3 Memory Access Rights

Priority of the four authority for mem:

request mem	Security super	Security user	Un-Security super	Un-Security user
Security super	√ +	√ +	√ +	√ +
Security user	X	√	X	X
Un-Security super	X	X	√	√ +
Un-Security user	X	X	X	√

Note: √ + all access, include read, write, data access, opcode fetch

√ access based on the current attribute

X no access

According to the picture above

ÿ The bus access of security-super can access sram at will

ÿ un-security-super's bus access can be freely accessed is configured as un-security-user

the region

ÿ The bus access of security-user can only access the security-user according to the current AP and CD attributes

the region

ÿ The bus access of un-security-super can access un

security-super ÿ region

ÿ The bus access of un-security-user can only access un

security-user ÿ region

The APx and CDx registers are used to set the access attributes of each region corresponding to different permissions, where CDx registers

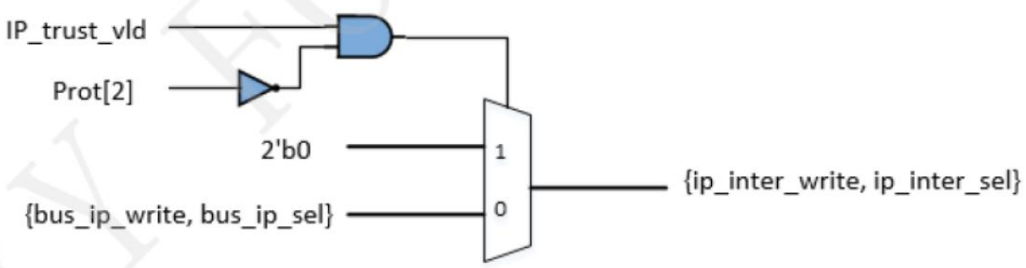
The register is only valid during the read operation during bus access, that is, to set whether to allow reading data and reading instructions. The APx registers are used to Set whether to allow read/write operations.

For example, if CAR[1:0] is set to 00, and CR[0] is 1, it means that REGION 0 is un-security-user, and And enable permission protection, if the bus access is security-super or un-security-super, you can access to REGION 0; if the bus access is Security-user, any access will be denied; if the bus access Q is un-security-user, AP0[1:0] is 01, indicating read-only, CD0[1:0] is 01, indicating data access, then REGION 0 allows the bus to read data, and all other operations are denied.

If the AHB bus accesses an inaccessible area or the permissions are wrong, the sasc module will give read deny or write deny signal, so as to return the wrong HRESP response signal, if the CPU initiates the bus access, then A hardware exception will be generated, and if it is another device, it will only deny access.

28.4.2 Trusted Access to Peripherals

The TIPC module can only be written when the PROT[2] signal is 1, that is, only in the trusted world can the Letter attribute configuration register. The Ip_trust_vld register is 0 by default, that is, all peripherals are untrusted. At this time, the peripherals It can be accessed at will. If the ip_trust_vld corresponding to the peripheral is set, that is, the peripheral is set as a trusted device, only Trusted world commands can access this peripheral.



29 Appendix 1. Definition of chip pins

29.1 Chip pin distribution

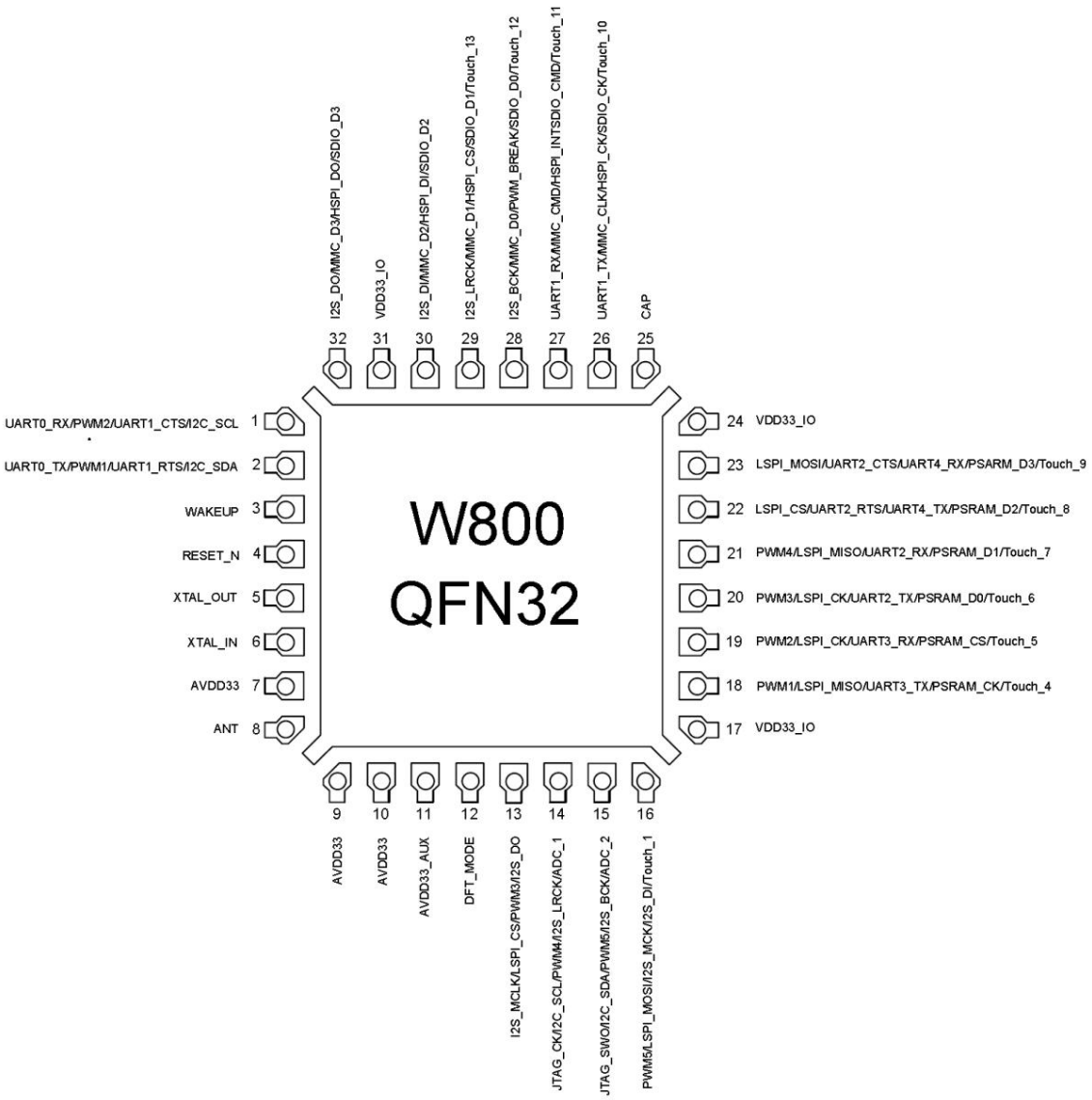


Figure 38 W800 chip pin distribution

29.2 Chip pin multiplexing relationship

Table 220 Chip pin multiplexing relationship

number	name	type	Pin function after reset	multiplexing function	Highest frequency	pull-up and pull-down	capability drive capability
1	PB_20	I/O	UART_RX	UART0_RX/PWM1/UART1_CTS/I²C_SCL	10MHz	UP/DOWN	12mA
2	PB_19	I/O	UART_TX	UART0_TX/PWM0/UART1_RTS/I²C_SDA	10MHz	UP/DOWN	12mA
3	WAKEUP		WAKEUP wake up function			DOWN	
4	RESET		RESET reset			UP	
5	XTAL_OUT	O	External crystal oscillator output				
6	XTAL_IN	I	External crystal oscillator input				
7	AVDD33	P	chip power supply, 3.3V				
8	ON	I/O	RF Antenna				
9	AVDD33	P	chip power supply, 3.3V				
10	AVDD33	P	chip power supply, 3.3V				
11	AVDD33_AUX	P	chip power supply, 3.3V				
12	TEST	I	Test function configuration pin				
13	BOOTMODE	I/O	BOOTMODE	I²S_MCLK/LSPI_CS/PWM2/I²S_DO	20MHz	UP/DOWN	12mA
14	PA_1	I/O	JTAG_CK	JTAG_CK/I²C_SCL/PWM3/I²S_LRCK/ADC0	20MHz	UP/DOWN	12mA
15	PA_4	I/O	JTAG_SWO	JTAG_SWO/I²C_SDA/PWM4/I²S_BCK/ADC1	20MHz	UP/DOWN	12mA
16	PA_7	I/O	GPIO, input, high impedance	PWM4/LSPI_MOSI/I²S_MCK/I²S_DI /Touch0	20MHz	UP/DOWN	12mA

17	VDD33IO	P	IO power supply, 3.3V				
18	PB_0	I/O GPIO, input, high impedance		PWM0/LSPI_MISO/UART3_TX/PSRAM_CK/Touch3	80MHz UP/DOWN		12mA
19	PB_1	I/O GPIO, input, high impedance		PWM1/LSPI_CK/UART3_RX/PSRAM_CS/Touch4	80MHz UP/DOWN		12mA
20	PB_2	I/O GPIO, input, high impedance		PWM2/LSPI_CK/UART2_TX/PSRAM_D0/Touch5	80MHz UP/DOWN		12mA
21	PB_3	I/O GPIO, input, high impedance		PWM3/LSPI_MISO/UART2_RX/PSRAM_D1/Touch6 80MHz UP/DOWN			12mA
22	PB_4	I/O GPIO, input, high impedance		LSPI_CS/UART2_RTS/UART4_TX/PSRAM_D2/Touch7 80MHz UP/DOWN			12mA
23	PB_5	I/O GPIO, input, high impedance		LSPI_MOSI/UART2_CTS/UART4_RX/PSARM_D3/Tou ch8	80MHz	UP/DOWN	12mA
24	VDD33IO	P	IO power supply, 3.3V				
25	CAP	I External capacitor, 1μF					
26	PB_6	I/O GPIO, input, high impedance		UART1_TX/MMC_CLK/HSPI_CK/SDIO_CK/Touch9	50MHz UP/DOWN		12mA
27	PB_7	I/O GPIO, input, high impedance		UART1_RX/MMC_CMD/HSPI_INT/SDIO_CMD/Touch 10	50MHz	UP/DOWN	12mA
28	PB_8	I/O GPIO, input, high impedance		I²S_BCK/MMC_D0/PWM_BREAK/SDIO_D0/ Touch11 50MHz UP/DOWN			12mA
29	PB_9	I/O GPIO, input, high impedance		I²S_LRCK/MMC_D1/HSPI_CS/SDIO_D1/ Touch12	50MHz UP/DOWN		12mA
30	PB_10	I/O GPIO, input, high impedance		I2S_DI/MMC_D2/HSPI_DI/SDIO_D2	50MHz UP/DOWN		12mA
31	VDD33IO	P	IO power supply, 3.3V				
32	PB_11	I/O GPIO, input, high impedance		I²S_DO/MMC_D3/HSPI_DO/SDIO_D3	50MHz UP/DOWN		12mA
33	GND	P ground					

statement

Beijing Lianshengde Microelectronics Co., Ltd. reserves the right to modify and update Lianshengde Microelectronics products or various documents and materials at any time. Place

Updates will be released through the official channel of Lianshengde Microelectronics. Users must ensure that they obtain correct information through official channels. Lian Sidley

Microelectronics does not promise to notify everyone of updated information.