SC6138A
用户手册

杭州士兰微电子股份有限公司
地址：杭州市黄姑山路 4 号
邮编：310012
主页：www.silan.com.cn

声明：

- 士兰保留本文档的更改权，恕不另行通知！
- 产品提升永无止境，我公司将竭诚为客户提供更优秀的产品！

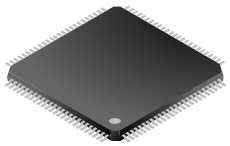
目　　录

高性能的数字音频处理器

1. 描述

SC6138A 是集成 APU/USB/SD/以太网控制器的音频编解码 SoC 电路，具有低成本、低功耗、高性能、高集成度的特点。芯片采用 32bit RISC 作为主控，辅以 APU（Audio Processing Unit）来完成音频编解码和后处理的架构。

SC6138A 结合外部 WiFi 和蓝牙模块，实现无线音乐推送播放和其他联网功能。



LQFP-100-14x14-0.5

2. 芯片架构

- RISC 核
  - 中天微 CK610 核心，精简指令集计算架构；

- 32 位数据长度，16 位指令长度，8 级流水线，300MHz 工作频率；
- 双发射指令，乱序执行，Dhrystone MIPS：1.6+/MHz；
- 支持 DSP 指令扩展，32 位乘法单元；
- 16KB 指令 Cache，16KB 数据 Cache，支持直写和写回操作；
- 支持 MMU，支持 linux 系统；

- APU 模块
  - 5 级流水线，工作频率 150MHz；
  - 16x16/32x32 乘法器，DSP 指令；
  - 专用硬件音频加速引擎：
    - 双乘法器 MAC 单元，32x16 或者 24x24；
  - 16KB 数据 Cache，16KB 指令 Cache；

时钟和电源管理

- 外接 32.768KHz 晶振和 12MHz 晶振；
- 内置以 12MHz 晶振为参考时钟的 System PLL，Audio PLL 和 USB1.1 PLL；
- 上电 System PLL 频率可外部配置，芯片运行中可以动态分频切换和改变 PLL 频率；
- 双电源供电，IO 和模拟模块 3.3V 供电，内核 1.2V；
- 集成单独供电 RTC，带万年历功能；
- 支持降频、STOP 等多种低功耗工作模式；

存储控制器

- SPI Flash 控制器
  - 支持 SPI Flash 1/2 线模式；
  - 支持 RISC 直接在 SPI Flash 上运行；
- SRAM
  - 4KB 内置 SRAM，启动时作为指令 RAM；
  - 支持 Byte，Half-word，Word 读写；
- SDRAM 控制器
  - 高效 SDRAM 控制器，支持 16bit 位宽操作；
  - 最高单颗支持 64MB；
  - 支持 Self-refresh，Power Down 等 SDRAM 低功耗模式；
  - 电路内部叠封 16M*16bit SDRAM 颗粒；

外设接口

- I^2S_I
  - 支持 5.1 声道输入；
  - 支持 24bit@192KHz；
  - 支持时钟 Slave 模式；
  - 支持 PDMA 模式；
- I^2S_O
  - 支持 5.1 声道输出；

3. 应用

无线（WiFi 和蓝牙）音箱
Soundbar
音效

- 支持 24bit@192KHz；
- 支持时钟 Master 和 Slave 模式；
- 支持 PDMA 模式；

- SPDIF_I
  - 支持 SPDIF 标准协议，4 通道输入选择；
  - 支持 Non-PCM 数据流检测；
  - 支持多种采样率检测；
  - 支持 Non-PCM 数据流中的 PC 和 PD 前置位接收；
  - 支持 CD 应用中的 Q-subcode 缓冲；
  - 支持 PDMA 模式；
- SPDIF_O
  - 支持 2.0 声道，24bit@192KHz；
  - 支持 PDMA 模式；
- Audio CODEC
  - 噪音抑制系统，软件静音模式；
  - 支持 24/16bit 数据位宽；
  - 支持 8/11.025/12/16/22.05/24/32/44.1/48/96KHz；
  - 输出短路保护；
  - 支持 MIC/LINE 输入，耳机和 LINE 输出；
- GMAC
  - 符合 IEEE802.3-2002 标准；
  - 支持 10/100Mbit/s 全双工、半双工模式；
  - 支持 RMII 接口；
  - 支持多样灵活的地址过滤模式；
  - 支持 MDIO 接口；
  - 内置 Normal 和链式 DMA 传输模式；
- USB_HS
  - 内置 USB2.0 高速 PHY；
  - 符合 USB 标准 2.0；
  - 支持控制、批量、中断和同步传输；
  - 内置 2KB SRAM；
  - 内置 Normal 和 Scatter-Gatter DMA 传输；
- USB_FS
  - 内置 USB1.1 全速 PHY；
  - 符合 USB 标准 1.1；
  - 支持控制、批量、中断和同步传输；
  - 内置 1KB SRAM；
  - 内置 DMA 传输；
- SD/MMC/SDIO
  - 符合 SD2.0，MMC4.3 和 SDIO2.0 标准规范；
  - 支持 1 和 4-bit 模式；
  - 内置 Normal 和链式 DMA 传输；

- UARTx4
  - 提供 4 个 UART 模块；
  - 高速 UART1 内置 64 深度 FIFO，UART2 深度 8，UART3/4 深度 16；
  - 支持数据位和停止位可编程；
  - 支持奇偶检验或者无校验；
  - 支持接收、发送 FIFO 中断；
  - UART1、UART2 支持 PDMA 模式；
- I^2Cx2
  - 支持标准、快速和高速三种模式；
  - 支持 Master 和 Slave 模式；
- SPI
  - 支持 SPI 标准 4 线协议；
  - 内建独立 8x32 发送和接收缓存；
  - 支持 PDMA 模式；
- PWM
  - 支持预置分频；
  - 8 路独立 PWM 输出；
  - 16 位计数精度；
- ADC
  - 7 路模拟输入通道，12bit 精度；
  - 支持参考电压可选，供电电压或者 2.4V 基准；
  - 支持手动，自动定时转换；
  - 支持两路定时转换；
  - 支持定时转换结果差值中断；
- GPIO
  - 两组 GPIO 控制器，51 GPIOs；
  - 独立上拉电阻使能；
  - 驱动能力选择（2/4/8/24mA）；
  - 支持每个 IO 的中断可配，沿或者电平触发；

内置硬件模块

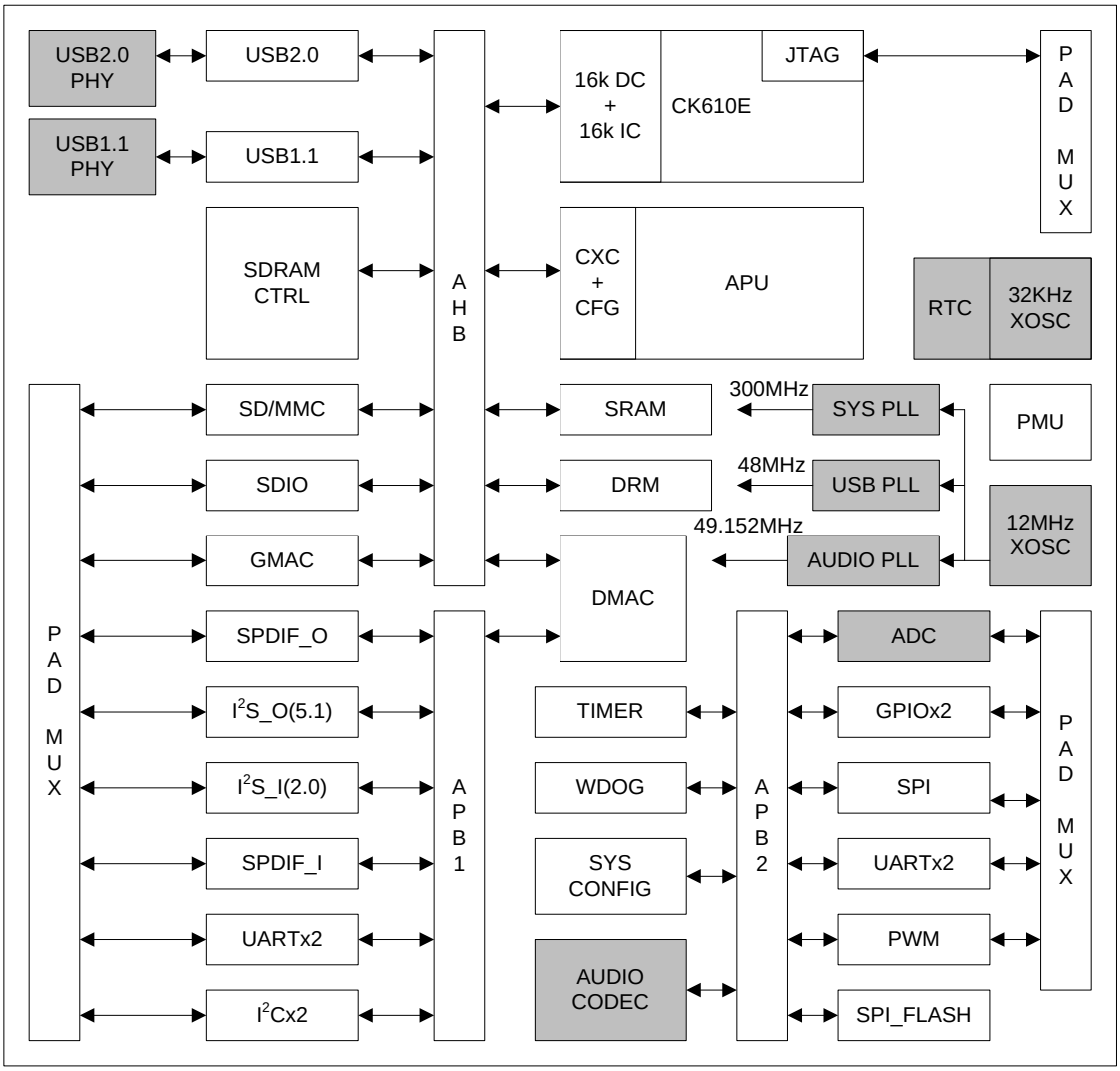
- DRM
  - AES（128），DES（64）和 SMS4（128）加解密，模式 CBC/ECB 等；
  - RSA（2048）加解密；
  - 内置 DMA，完成数据输入输出；
- PDMA
  - 支持 5 路独立可配置通道；
  - 支持 Normal 和 Scatter-Gatter 传输模式；
  - 支持 Byte-，Half-word-和 Word-存取；
  - 支持源/目标地址自增；
  - 支持 16 路数据请求通道；

- Timer
  - 支持预置分频；
  - 四组独立计数器；
  - 32bit 计数精度；

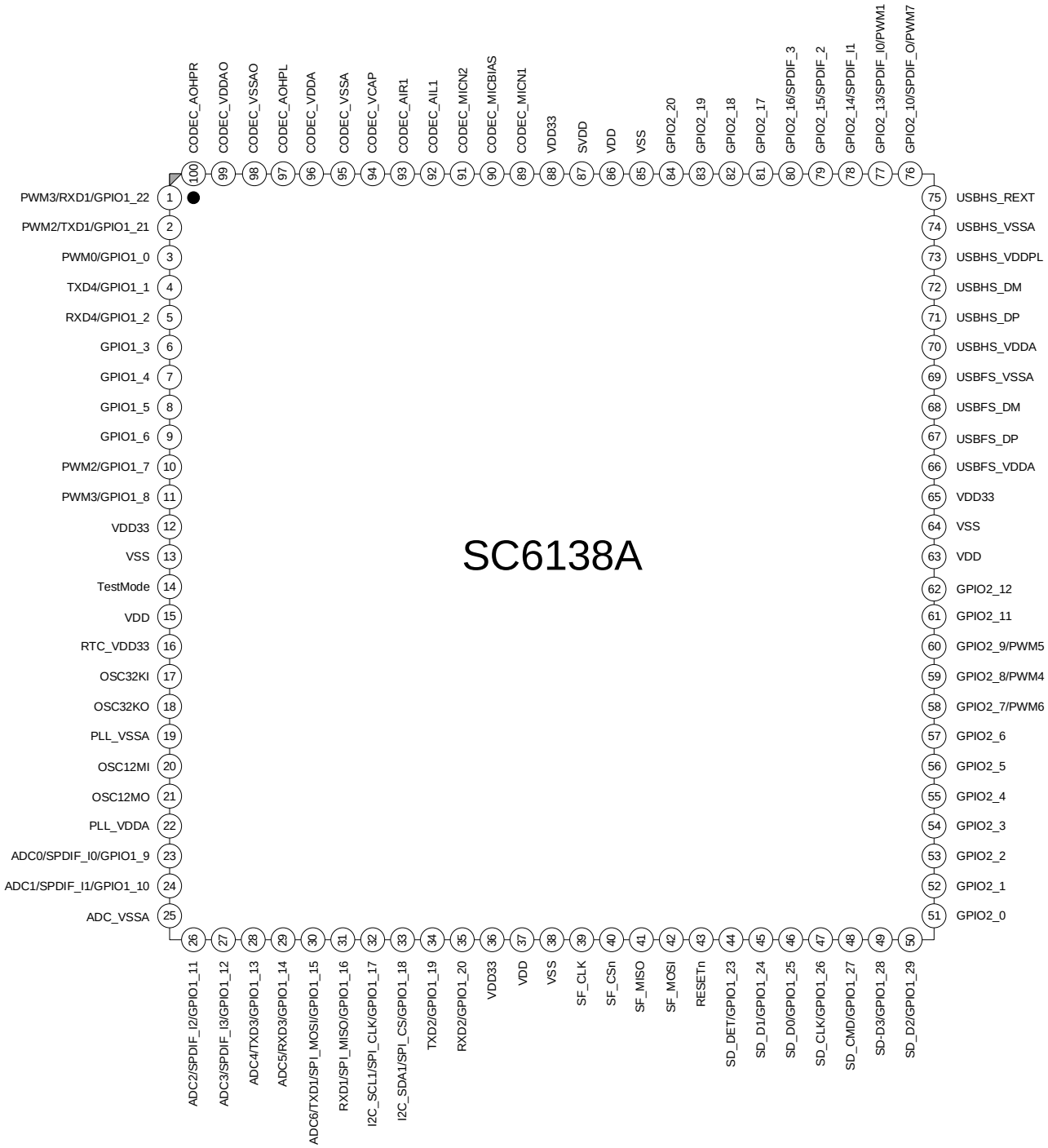
4. 产品规格分类

产品名称	封装形式	打印名称	材料	包装
SC6138A	LQFP-100-14×14-0.5	SC6138A	无卤	料盘

5. 内部框图



6. 管脚排列图



7. 管脚描述

管脚号	管脚名称	I/O	管 脚 描 述
1	GPIO1_22	IO	GPIO1[22]，复用ADC_SDI2/RXD1/PWM3
2	GPIO1_21	IO	GPIO1[21]，复用 ADC_SDI1/TXD1/PWM2
3	GPIO1_0	IO	GPIO1[0]，复用 ADC_SDI0/PWM0

管脚号	管脚名称	I/O	管 脚 描 述
4	GPIO1_1	IO	GPIO1[1]，复用 ADC_WS/TXD4
5	GPIO1_2	IO	GPIO1[2]，复用 ADC_SCLK/RXD4
6	GPIO1_3	IO	GPIO1[3]，复用 DAC_MCLK
7	GPIO1_4	IO	GPIO1[4]，复用 DAC_SCLK
8	GPIO1_5	IO	GPIO1[5]，复用 DAC_WS
9	GPIO1_6	IO	GPIO1[6]，复用 DAC_SDO0
10	GPIO1_7	IO	GPIO1[7]，复用 DAC_SDO1/PWM2
11	GPIO1_8	IO	GPIO1[8]，复用 DAC_SDO2/PWM3
12	VDD33	P3	数字 IO 电源 3.3V
13	VSS	G	数字地
14	TestMode	I	测试模式
15	VDD	P1	数字内核电源 1.2V
16	RTC_VDD33	P3	RTC 电源 3.3V
17	OSC32KI	AI	32.768KHz 晶振输入
18	OSC32KO	AO	32.768KHz 晶振输出
19	PLL_VSSA	GA	PLL 模拟地
20	OSC12MI	AI	12M 晶振输入
21	OSC12MO	AO	12M 晶振输出
22	PLL_VDDA	PA	PLL/ADC 模拟电源 3.3V
23	GPIO1_9	IO/AI	GPIO1[9]，复用 SPDIF_I0/按键 ADC_AIN0
24	GPIO1_10	IO/AI	GPIO1[10]，复用 SPDIF_I1/按键 ADC_AIN1/TRSTn
25	ADC_VSSA	GA	ADC 模拟模块地
26	GPIO1_11	IO/AI	GPIO1[11]，复用 SPDIF_I2/按键 ADC_AIN2/TCK
27	GPIO1_12	IO/AI	GPIO1[12]，复用 SPDIF_I3/按键 ADC_AIN3/TMS
28	GPIO1_13	IO/AI	GPIO1[13]，复用 TXD3/按键 ADC_AIN4/TDI
29	GPIO1_14	IO/AI	GPIO1[14]，复用 RXD3/按键 ADC_AIN5/TDO
30	GPIO1_15	IO/AI	GPIO1[15]，复用 TXD1/SPI_MOSI/按键 ADC_AIN6
31	GPIO1_16	IO/AI	GPIO1[16]，复用 RXD1/SPI_MISO/按键 ADC_VREFO
32	GPIO1_17	IO/AO	GPIO1[17]，复用 SCL1/SPI_CLK
33	GPIO1_18	IO	GPIO1[18]，复用 SDA1/SPI_CS
34	GPIO1_19	IO	GPIO1[19]，复用 TXD2/SF_MISO2
35	GPIO1_20	IO	GPIO1[20]，复用 RXD2/SF_MISO3
36	VDD33	P3	数字 IO 电源 3.3V
37	VDD	P1	数字内核电源 1.2V
38	VSS	G	数字地
39	SF_CLK	O	SPI Flash 时钟，复用 SPLL_SEL[2]
40	SF_CSn	O	SPI Flash 片选
41	SF_MISO	I	SPI Flash 数据输入，复用 4 线 IO1/SPLL_SEL[1]
42	SF_MOSI	O	SPI Flash 数据输出，复用 4 线 IO0/SPLL_SEL[0]
43	RESETn	I	复位

管脚号	管脚名称	I/O	管 脚 描 述
44	GPIO1_23	IO	GPIO1[23]，复用 SD_DET/PWM1
45	GPIO1_24	IO	GPIO1[24]，复用 SD_D1
46	GPIO1_25	IO	GPIO1[25]，复用 SD_D0
47	GPIO1_26	IO	GPIO1[26]，复用 SD_CLK
48	GPIO1_27	IO	GPIO1[27]，复用 SD_CMD
49	GPIO1_28	IO	GPIO1[28]，复用 SD_D3
50	GPIO1_29	IO	GPIO1[29]，复用 SD_D2
51	GPIO2_0	IO	GPIO2[0]，复用 ENET_TXD1/SDIO_DET
52	GPIO2_1	IO	GPIO2[1]，复用 ENET_TXD0/SDIO_D1
53	GPIO2_2	IO	GPIO2[2]，复用 ENET_TX_EN/SDIO_D0
54	GPIO2_3	IO	GPIO2[3]，复用 ENET_MDC/SDIO_CLK
55	GPIO2_4	IO	GPIO2[4]，复用 ENET_MDIO/SDIO_CMD
56	GPIO2_5	IO	GPIO2[5]，复用 ENET_RX_DV/SDIO_D3
57	GPIO2_6	IO	GPIO2[6]，复用 ENET_RX_ER/SDIO_D2
58	GPIO2_7	IO	GPIO2[7]，复用 ENET_RXD0/PWM6
59	GPIO2_8	IO	GPIO2[8]，复用 ENET_RXD1/TXD4/PWM4
60	GPIO2_9	IO	GPIO2[9]，复用 ENET_CLK/RXD4/PWM5
61	GPIO2_11	IO	GPIO2[11]，复用 ADC_SDI1/TXD3/SCL2/TEST_CFG[0]
62	GPIO2_12	IO	GPIO2[12]，复用 ADC_SDI2/RXD3/SDA2/TEST_CFG[1]
63	VDD	P1	数字内核电源 1.2V
64	VSS	G	数字地
65	VDD33	P3	数字 IO 电源 3.3V
66	USBFS_VDDA	PA	USBFS PHY 模拟电源 3.3V
67	USBFS_DP	A	USBFS 数据线 D+
68	USBFS_DM	A	USBFS 数据线 D-
69	USBFS_VSSA	GA	USBFS PHY 模拟地
70	USBHS_VDDA	PA	USBHS PHY 模拟电源 3.3V
71	USBHS_DP	A	USBHS 数据线 D+
72	USBHS_DM	A	USBHS 数据线 D-
73	USBHS_VDDPL	PA1	USBHS PHY 模拟电源 1.2V
74	USBHS_VSSA	GA	USBHS PHY 模拟地
75	USBHS_REXT	A	USBHS PHY 参考电阻，11KΩ
76	GPIO2_10	IO	GPIO2[10]，复用 SPDIF_O/PWM7
77	GPIO2_13	IO	GPIO2[13]，复用 SPDIF_I0/PWM1
78	GPIO2_14	IO	GPIO2[14]，复用 SDIO_DET/SPDIF_I1
79	GPIO2_15	IO	GPIO2[15]，复用 SDIO_D1/SPDIF_I2
80	GPIO2_16	IO	GPIO2[16]，复用 SDIO_D0/SPDIF_I3
81	GPIO2_17	IO	GPIO2[17]，复用 SDIO_CLK
82	GPIO2_18	IO	GPIO2[18]，复用 SDIO_CMD
83	GPIO2_19	IO	GPIO2[19]，复用 SDIO_D3

管脚号	管脚名称	I/O	管 脚 描 述
84	GPIO2_20	IO	GPIO2[20]，复用 SDIO_D2
85	VSS	G	数字地
86	VDD	P1	数字内核电源 1.2V
87	SVDD	P3	SDRAM 颗粒 VDD3.3V
88	VDD33	P3	数字 IO 电源 3.3V
89	CODEC_MICN1	AI	CODEC 麦克风单端输入 1
90	CODEC_MICBIAS	AO	CODEC 麦克风偏置
91	CODEC_MICN2	AI	CODEC 麦克风单端输入 2
92	CODEC_AIL1	AI	CODEC 左声道 LINE 输入
93	CODEC_AIR1	AI	CODEC 右声道 LINE 输入
94	CODEC_VCAP	A	CODEC 外接电容
95	CODEC_VSSA	GA	CODEC 模拟地
96	CODEC_VDDA	PA	CODEC 模拟电源 3.3V
97	CODEC_AOHPL	AO	CODEC 左声道输出
98	CODEC_VSSAO	GA	CODEC 模拟地
99	CODEC_VDDAO	PA	CODEC 模拟电源 3.3V
100	CODEC_AOHPR	AO	CODEC 右声道输出

注：

1） I/O 类型说明：

I – 输入管脚

O – 输出管脚

IO – 双向管脚

A – 表示是模拟管脚，AI 是模拟输入，AO 是模拟输出

P – 表示电源，P3 表示 3.3V 的 IO 电源，PA 表示模拟 3.3V 电源，PA1 表示模拟 1.2V 电源

G – 表示地线，GA 表示模拟地

8. 极限参数

参 数	符 号	参 数 范 围	单 位
内核电压	V_{CCINT}	1.08~1.32	V
端口电压	V_{CCIO}	2.97~3.63	V
管脚输入电压	V_{IN}	2.97~3.63	V
工作温度范围	T_{amb}	-40~85	°C
贮存温度范围	T_{STG}	-40~120	°C

9. 直流电气参数（除非特别指定，V_{DD}=3.3V，T_{AMB}=25°C）

参 数	符 号	测 试 条 件	最小值	典型值	最大值	单位
内核工作电压	V_{CCINT}	正常工作	1.08	1.20	1.32	V
端口工作电压	V_{CCIO}	正常工作	2.97	3.30	3.63	V
主振模式 VDD12 工作电流	I_{CCIO}	端口V_{DD33}电源电流	160	200	220	mA

参　　数	符　　号	测 试 条 件	最小值	典型值	最大值	单位
RISC 待机模式 VDD12 工作电流	I_{CCIO}	端口V_{DD33}电源电流	60	80	100	mA
电路待机模式 VDD12 工作电流	I_{CCIO}	端口V_{DD33}电源电流	4	6	8	mA
主振模式 VDD33 工作电流	I_{CCIO}	端口V_{DD33}电源电流	30	40	50	mA
RISC 待机模式 VDD33 工作电流	I_{CCIO}	端口V_{DD33}电源电流	16	18	20	mA
电路待机模式 VDD33 工作电流	I_{CCIO}	端口V_{DD33}电源电流	8	10	12	mA
工作频率　低频时钟	--	--	6.0			MHz
上拉电阻	R_{PU}		60	70	80	KΩ
晶振反馈电阻	R_{OSC}	--	--	1.0	--	MΩ
高电平输入电压	V_{IH}	--	2.4	3.3	3.63	V
低电平输入电压	V_{IL}	$V_{IN}=V_{DD}$	0	0.8	1.2	V
高电平输入电流	I_{IH}	$V_{IN}=V_{DD}$	--	0	--	μA
低电平输入电流	I_{IL}	$V_{IN}=V_{SS}$	--	0	--	μA
高电平输出电流	I_{OH}	V_{OH}=2.4V，I_{OH}=8mA	--	20	--	mA
低电平输出电流	I_{OL}	V_{OL}=0.4V，I_{OH}=8mA	--	10	--	mA
信噪比（DAC）	SNR_{DAC}	10KΩ负载，A-Weighted	--	90	--	dB
信噪比（ADC）	SNR_{ADC}	A-Weighted	--	85	--	dB
总谐波失真（DAC）	THD_{DAC}	FS -1 dB	--	80	--	dB
总谐波失真（ADC）	THD_{ADC}	FS -1 dB	--	80	--	dB

10. 功能描述

10.1. 简单描述

SC6138A 是一颗高集成度，专注于无线音乐、音频编解码、音效处理的 SoC 芯片。它集成了 APU 单元，专门来处理音频相关的算法，同时还将音频 DAC/ADC、音频接口、USB/SD/以太网控制器等集成在单芯片中，从而实现音乐播放，无线推送和音效处理等多方面的应用。

10.2. 测试和调试

10.2.1. 调试功能描述

主要特性如下：

（1） 进入 RISC 调试模式，需要外部 FLASH，软件配置进入；

（2） RISC 调试模式下，启动/初始化程序通过 JTAG 下载到内部 4K RAM 和叠封 SDRAM；

（3） 采用标准的 JTAG 调试接口；

10.2.2. 调试模式使用说明

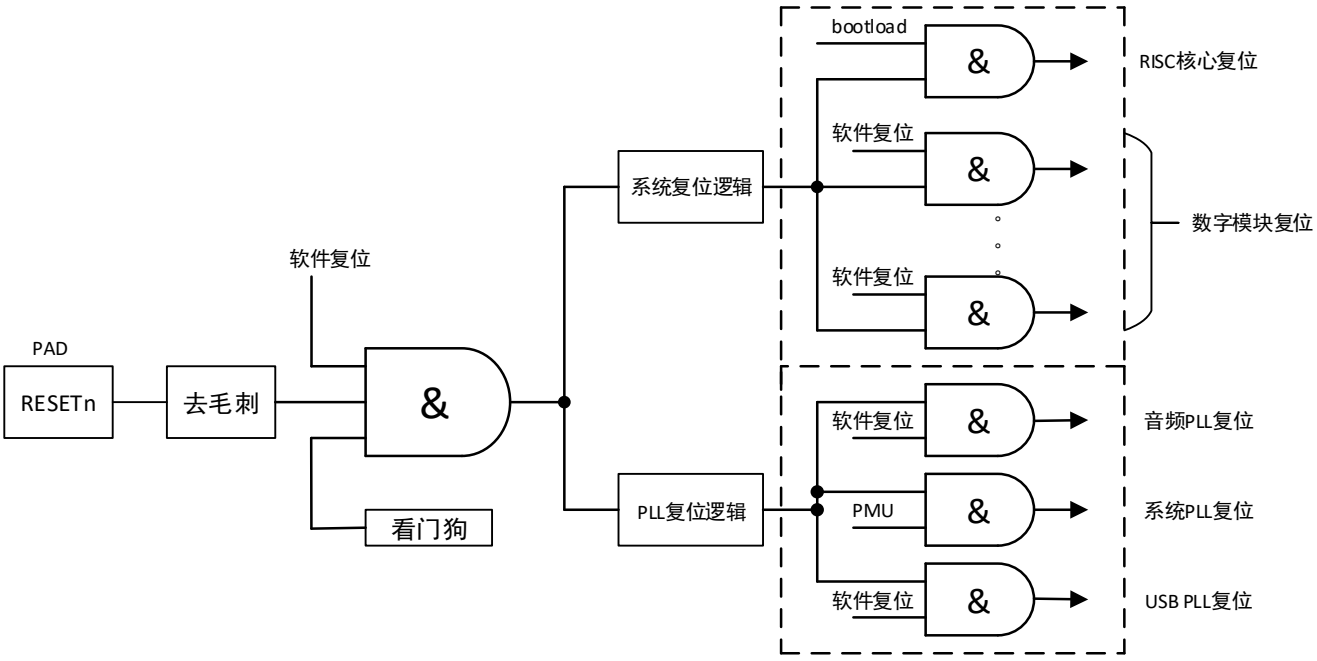
支持的 JTAG 接口

管脚名称	Debug接口信号
GPIO1_11	JTAG 的 TCK
GPIO1_12	JTAG 的 TMS
GPIO1_13	JTAG 的 TDI
GPIO1_14	JTAG 的 TDO
GPIO1_10	JTAG 的 TRSTN

10.3. 系统状态控制

10.3.1. 复位

内置三种类型复位，分别是外部复位，看门狗复位和软件复位。



软件复位，均是通过寄存器来控制，具体参见功能模块的系统控制模块和 VLSP 模块。

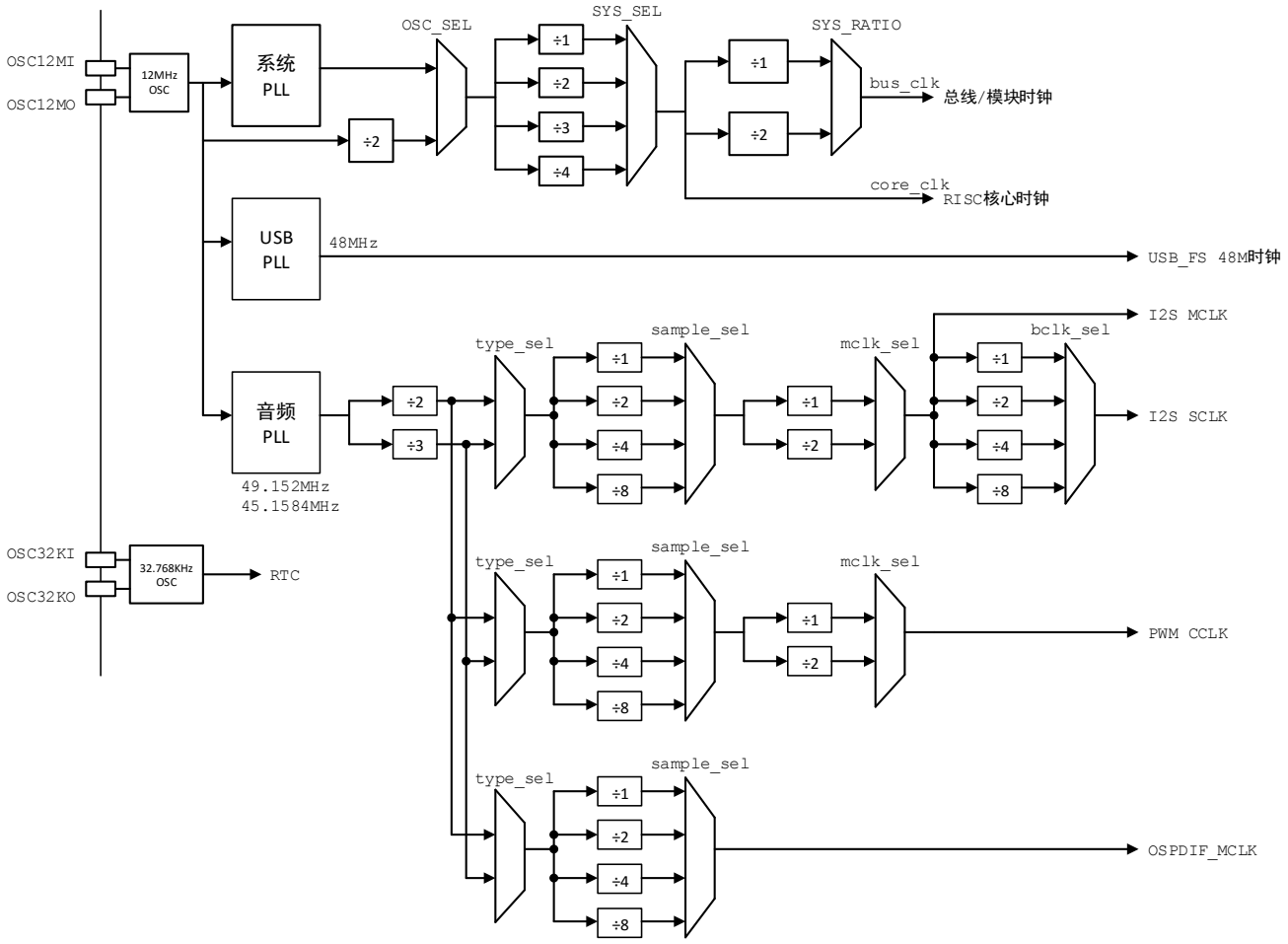
10.3.2. 时钟配置

芯片有两个时钟源，

- 12MHz 振荡器时钟
- 32.768KHz 振荡器时钟

其中 32.768KHz 振荡器时钟只给 RTC，作为计时时钟。

芯片包含三个 PLL，分别是 SYS_PLL，AUDIO_PLL 和 USB_PLL，分别为系统逻辑、音频部分和 USB1.1 控制器提供时钟，这三个 PLL 均以 12MHz 振荡器作为时钟源。



10.3.2.1. 系统时钟

系统主时钟产生的框图，系统时钟主要包括两个，core_clk 和 bus_clk。其中 core_clk 是 RISC 的核心时钟，而 bus_clk 是总线时钟以及各个模块的总线时钟（或者模块主时钟）。core_clk 和 bus_clk 是同步时钟，其比例关系支持 1:1 和 2:1，上电默认是 2:1。

系统 PLL 的频率上电时是由三个外部管脚在复位期间的电平决定的，复位完成后，这三个管脚恢复原先的功能。如下表：

SPLL_SEL[2] SF_CLK	SPLL_SEL[1] SF_MISO	SPLL_SEL[0] SF_MOSI	频率值 （MHz）	备注
0	0	0	90	
0	0	1	150	
0	1	0	180	
0	1	1	216	
1	0	0	240	
1	0	1	264	
1	1	0	300	
1	1	1	非法	上电不能使用

系统时钟除了上电配置外，在运行过程中也可以动态切换，这主要靠 PMU 模块来控制。具体参见 10.3.4.2 功耗管理单元。

10.3.2.2. 音频时钟

芯片集成专用的音频 PLL，满足特殊的频率要求（49.152MHz 和 45.1584MHz），产生的时钟分别提供给 I2S，OSPIDF 和 PWM。

10.3.3. 启动流程

上电复位完成后，除了 RISC 之外，其他部分复位均释放。模块 Bootloader 开始从外部 SPI FLASH 默认地址 0x0 处拷贝 4K 代码到内部 SRAM（0x1FC0_0000），拷贝完成后，RISC 复位释放，RISC 开始从 SRAM 取指运行。

10.3.4. 低功耗管理

10.3.4.1. 电源域

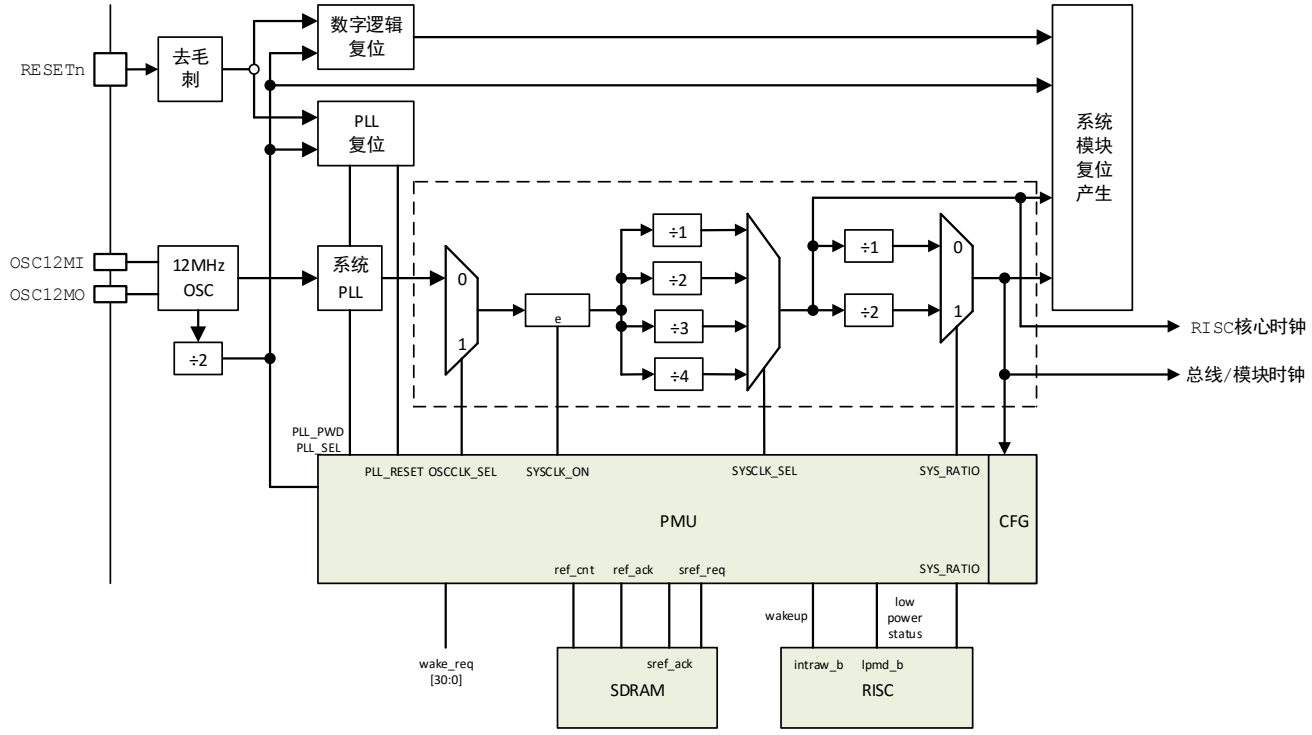
芯片分为两组电源域，

电源域	电源网络	电压(V)	关联模块	关联电源网络	备注
PD0	POWER0	3.3	IO，模拟模块	外部提供	
	POWER1	1.2	数字逻辑、模拟模块	外部提供	
PD1	POWER2	3.3	RTC 模块	外部电池	包括内置 RAM

PD0 和 PD1 可以独立供电，相互独立。PD0 中的 POWER0 和 POWER1 要求同时上下电。

10.3.4.2. 功耗管理单元

PMU 硬件模块根据 CPU 请求、状态信息和外部 GPIO 中断源，自动的、安全的控制系统时钟，主要包括 SYS_PLL 的关断和输出频率调整、SYS_CLK 多个频率选择、CPU 时钟门控开关等。下图为 PMU 控制的结构框图：



PMU 模块工作时钟为外部晶振时钟（12MHz）的二分频，PMU 可以独立工作，不需要考虑 PLLs、CPU 时钟等的开关状态。

当 CPU 向 PMU 请求进入低功耗状态后，PMU 模块会根据具体的请求类型，自动的安全关掉 SYS_PLL 或者 CPU 时钟。

之后 PMU 模块会一直监控唤醒中断源，一旦监测到唤醒中断后，会根据请求类型打开 SYS_PLL 和 CPU 时钟，发出唤醒中断给 CPU 来唤醒 CPU。一共有 31 个唤醒中断源。每个中断源可以独立配置，包括中断使能或禁止、触发极性、触发脉冲宽度等。

低功耗流程

PMU 低功耗控制流程分为 Bypass 和 Non-Bypass 两种类型：

当选择 Bypass 类型时，CPU 只需要配置 PMU_CFG 和 PLL_CFG_L/PLL_CFG_H，寄存器的配置值会直接送给时钟选择、时钟门控和 PLL 模块，时钟和 PLL 会立刻产生相应的变化。因为一般程序都是运行在 SDRAM 上，而切换时钟会导致 SDRAM 时钟改变，极有可能导致 SDRAM 数据出错。严重不推荐使用此种办法。

当选择 Non-Bypass 类型时，PMU 模式有四种低功耗模式，分别是时钟分频切换（Clock Switch），PLL 频率切换（PLL Frequency Update），CPU 时钟关断（CPU Off），SYS_PLL 关闭（PLL Powerdown）。CPU 和 PMU 执行如下流程来进行状态的安全切换：

- CPU 首先配置 PMU_CFG 寄存器来选择相应的低功耗模式。如果 CPU 需要进入 Clock Gate 或者 PLL Powerdown 模式，则还需配置相关的中断寄存器。配置结束后，CPU 可以立刻调用 wait 指令，进入低功耗状态。
- PMU 会不断查询 CPU 是否已经进入低功耗状态。
- 一旦 CPU 进入低功耗状态后，如果 PMU 处于 Clock Switch 或者 PLL Frequency Update 模式，PMU 首先会更新 Clock 或者 PLL，然后通过中断唤醒 CPU。
- 如果 PMU 处于 Clock Gate 或者 PLL Powerdown 模式，PMU 首先会关掉 Clock 或者 PLL。然后根据配置使能相关的 GPIO 唤醒中断并等待中断。一旦检测到期望的 GPIO 唤醒中断，PMU 将会通过中断唤醒 CPU。

CPU 被唤醒后，应清除 PMU 中断和进入工作模式。

寄存器描述

模块寄存器基地址

Module	Base Addr	End Addr	Addr Space
PMU	0x1A04_0000	0x1A04_FFFF	64K

Table - 1 寄存器列表

Offset	Name	Description	Type
0x00	PMU_CFG	PMU 配置寄存器	RW
0x04	PLL_CFG_L	PLL 配置寄存器低位	RW
0x08	PLL_CFG_H	PLL 配置寄存器高位	RW
0x0c	INT_MASK_CFG	PMU 外部中断屏蔽寄存器	RW
0x10	INT_ POLARITY_CFG	PMU 外部中断极性寄存器	RW
0x14	INT_CFG	PMU 外部中断配置寄存器	RW
0x18	INT_STATUS	PMU 中断状态	RO
0x1c	INT_LEVEL_CNT_CFG	PMU 外部中断电平触发计数器值	RW
0x20	PMU_STATUS	PMU 状态寄存器	RO
0x24	PLL_STATUS_L	PLL 状态寄存器低位	RO
0x28	PLL_STATUS_H	PLL 状态寄存器高位	RO
0x2c	WAIT_CNT	PMU 状态转换等待计数器值	RW

Table - 2 PMU_CFG 位描述

Field	Name	Description	Access	Reset
31:15	reserved	reserved	reserved	reserved
14	CLK_RATIO	CPU:BUS 时钟比 0 <->1:1 1 <->2:1	RW	1
13				
12	CPUCLK_ON	CPU 时钟打开 （Bypass 模式下有效，其他模式下 PMU 自动控制）	RW	0x0
11	CLK_SOURCE	系统时钟源选择：		

		0 <-> 系统 PLL 1 <-> 晶振时钟二分频		
10:8	CLK_SWITCH	系统时钟分频选择： 0 <-> /1 1 <-> /2 2 <-> /3 3 <-> /4	RW	0x0
7	REFCNT_EN	SDRAM 刷新寄存器配置使能	RW	0x0
6	SREF_REQ	SDRAM 自刷新软件请求		
5	SREF_MODE	SDRAM 自刷新模式 0：软件配置 1：PMU 自动控制 如果改变了总线频率，务必根据时钟频率设置 SDRAM 刷新寄存器值 REF_CNT 和使能 REFCNT_EN，SREF_MODE 也要使能。		
4:3	MODE	低功耗模式，需要 CPU 首先进入 wait 状态，然后更新时钟和 PLL 状态，最后根据低功耗模式唤醒 CPU 0 <-> 系统时钟切换 1 <-> CPU 时钟关闭 2 <-> 系统 PLL 配置更新 3 <-> 系统 PLL 关闭	RW	0x0
2	BYPASS	低功耗 BYPASS 模式 如果设置为 1， CLK_SOURCE/CLK_SWITCH/PLL_CFG 将直接作用于时钟控制上，会导致无法预知的问题。严重不建议使用。	RW	0x0
1	REQ	低功耗请求，软件置 1，硬件自动清零	RWAC	0x0

0	EXIT	低功耗退出，软件置 1，硬件自动清零	RWAC	0x0

Table - 3 PLL_CFG_L 位描述

Field	Name	Description	Access	Reset
31:28	PLL_SEL	系统 PLL 频率快速配置	RW	0xD
27:16	reserved	reserved	reserved	reserved
15:10	PLL_DM	系统 PLL 输入分频，仅在 PLL_SEL=0xF 时有效	RW	0x0
9:4	PLL_DN	系统 PLL 反馈分频，仅在 PLL_SEL=0xF 时有效	RW	0x0
1	PLL_PWD	系统 PLL 关闭，高电平有效，仅在 BYPASS 模式下使用	RW	0x0
0	PLL_RESET	系统 PLL 复位，仅在 BYPASS 模式下使用	RW	0x0

Table - 4 PLL_CFG_H 位描述

Field	Name	Description	Access	Reset
31:16	reserved	reserved	reserved	reserved
15:0	REF_CNT	SDRAM 控制器刷新计数器值 SDRAM 时钟*7.8，取整数	RW	0x0

Table - 5 INT_ MASK_CFG 位描述

Field	Name	Description	Access	Reset
31	reserved	reserved	reserve	reserve

			d	d
30:0	INT_MASK	中断源位屏蔽 0 <-> enable 1 <-> disable	RW	0x0

31 个唤醒源：

序号	唤醒源	描述
0	RTC 中断	
1	看门狗中断	
2	Timer 中断	
3	GPIO1 中断	
4	GPIO2 中断	
5	GPIO1_0	
6	GPIO1_1	
7	GPIO1_2	
8	GPIO1_3	
9	GPIO1_4	
10	GPIO1_5	
11	GPIO1_6	
12	GPIO1_7	
13	GPIO1_8	
14	GPIO1_9	
15	GPIO1_10	
16	GPIO1_11	
17	GPIO1_12	
18	GPIO1_13	
19	GPIO1_14	
20	GPIO1_15	

21	GPIO1_16	
22	GPIO1_17	
23	GPIO1_18	
24	GPIO1_19	
25	GPIO1_20	
26	GPIO2_9	
27	GPIO2_10	
28	GPIO2_11	
29	GPIO2_12	
30	中断控制器输出	

Table - 6 INT_POLARITY_CFG 位描述

Field	Name	Description	Access	Reset
31	reserved	reserved	reserved	reserved
30:0	INT_POL	唤醒中断极性 1 <-> 高电平有效 0 <-> 低电平有效	RW	0xff

Table - 7INT_CFG Bit Descriptions

Field	Name	Description	Access	Reset
31:1	reserved	reserved	reserved	reserved
0	INT_CLR	唤醒中断清 0	RWAC	0x0

Table - 8 INT_STATUS Bit Descriptions

Field	Name	Description	Access	Reset
31	PMU_INT	PMU 中断状态，时钟切换和 PLL 频率更新模式，改变后，PMU 将会自动立刻 CPU	RO	
30:0	GPIO_INT	唤醒源中断状态	RO	

Table - 9 INT_LEVEL_CNT_CFG Bit Descriptions

Field	Name	Description	Access	Reset
31:8	reserved	reserved	reserved	reserved
7:0	LEVEL_CNT	唤醒源中断电平长度计数器，用于滤除外部 GPIO 的毛刺	RW	RW

Table - 10PMU_STATUS 位描述

Field	Name	Description	Access	Reset
31:29	reserved	reserved	reserved	reserved
28	SREF_ACK	SDRAM 自刷新 ACK	RO	
27:20	CNT	PMU 内部计数器	RO	
19:16	FSM	PMU 状态机状态	RO	
15:9	reserved	reserved	reserved	reserved
8	SREF_REQ			
7	REF_LOAD			
6	CLK_RATIO			
5	SYSCLK_ON			
4	CPUCLK_ON		RO	

3:0	CLK_SEL		RO	

Table - 11 PLL_STATUS_L 位描述

Field	Name	Description	Access	Reset
31:0	PLL_STATUS_L	PLL 配置寄存器状态	RO	

Table - 12 PLL_STATUS_H 位描述

Field	Name	Description	Access	Reset
31:0	PLL_STATUS_H	PLL 配置寄存器状态	RO	

Table - 13 WAIT_CNT 描述

Field	Name	Description	Access	Reset
31:24	reserved	reserved	reserved	reserved
23:16	RST_CNT	pll reset wait counter PLL 复位等待计数器值	RW	0x8c
14:10	PD_CNT	pll power down wait counter PLL 关闭等待计数器值	RW	0x19
9:5	GATE_CNT	clock gate wait counter 系统时钟关闭等待计数器值	RW	0x3
4:0	SWITCH_CNT	系统时钟切换等待计数器值	RW	0x7

10.4. 模拟模块

10.4.1. 按键 ADC

10.4.1.1. 整体描述

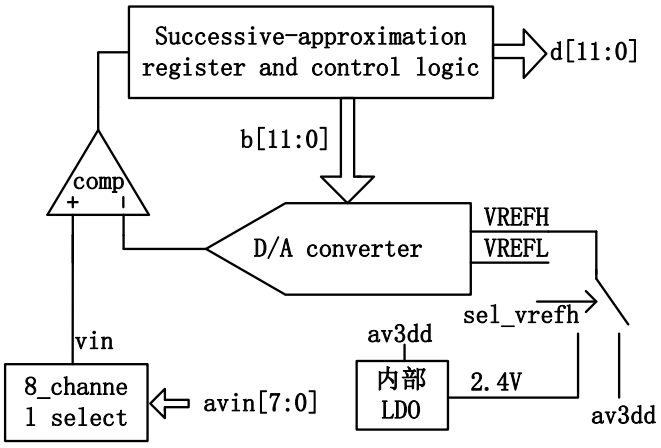
按键 ADC 是 12 位 8 通道 SAR ADC，其主要作用是将模拟信号转换为 12 位的数字信号，以便识别输入的不同按键值。高 7 位用开关电容阵列实现，低 5 位用电阻阵列实现；该版本将输入数字信号的高电平均修改为 1.2V，同时加入 LDO，输出 2.4V 电压，作为可选择基准，使得采样数值不跟随电源电压变化；

10.4.1.2. 特征

- 8 通道输入选择
- 12bit 精度
- 最大 12Ksps 采样率
- 内部集成输入上拉电阻
- 内部备用 2.4V 基准可选择，使得采样结果不跟随电源电压变化

10.4.1.3. 模块描述

➢ 功能框图



功能框图

➢ 详细说明

详细说明

端口名	I/O	端口描述	PAD
pd	I	ADC 使能信号。为 1 时关闭所有时钟和模拟信号，为 0 时 ADC 模块能正常工作	/
c[2:0]	I	输入模拟信号选通控制信号 c2 c1 c0 VIN 0 0 0 ain0 0 0 1 ain1 0 1 0 ain2 0 1 1 ain3 1 0 0 ain4	/

		1 0 1 ain5 1 1 0 ain6 1 1 1 ain7	
conv_st	I	ADC 的使能信号。0 有效时，下降沿触发，ADC 开始正常工作。为 1 时，ADC 停止转换。高电平 1 个 clk，低电平至少 14 个 clk。	/
rsel[1:0]	I	上拉电阻大小及按键脉冲检测控制信号。 CONT1 CONT0 状态 0 0 上拉电阻为 12K（支持 6 位按键） 0 1 上拉电阻为 33K（支持 8 位按键） 1 * 屏蔽上拉电阻及脉冲计检测功能，输入电压和上拉电阻全部由外部提供	/
finish	O	AD 转换结束信号，0 表示正在进行转换，1 为结束转换。只有一个时钟宽度，用来告知转换结束，并可用它来锁存数据。	/
d[11:0]	O	12 位 ADC 输出数据信号	/
detect	O	当 ADC 关断时，能检测输入信号的跳变，从而输出指示信号。当没有按键信号时，8 个输入端全部上拉到电源，输出为 0。当有按键信号时，输出为 1。高电平为 3.3V	/
ext_cnvc	I	ADC 外部控制使能, ext_cnvc 为高时,ADC 由 conv_st 控制转换；否则, ADC 每 16 个时钟自动转换一次。	/
sel_vrefh	I	使用内部 2.4V 基准选择位；高电平有效，即 sel_vrefh=1 时使用内部 2.4V 作为 SARADC 的基准，默认值为 sel_vrefh=0	/

10.4.2. AUDIO CODEC

10.5. 数字系统

10.5.1. 系统架构

以 RISC32 处理器内核为中心，APU 负责音频解码、后处理等，通过 AMBA 总线直接控制的一系列模块和存储器的体系。系统框图如下：

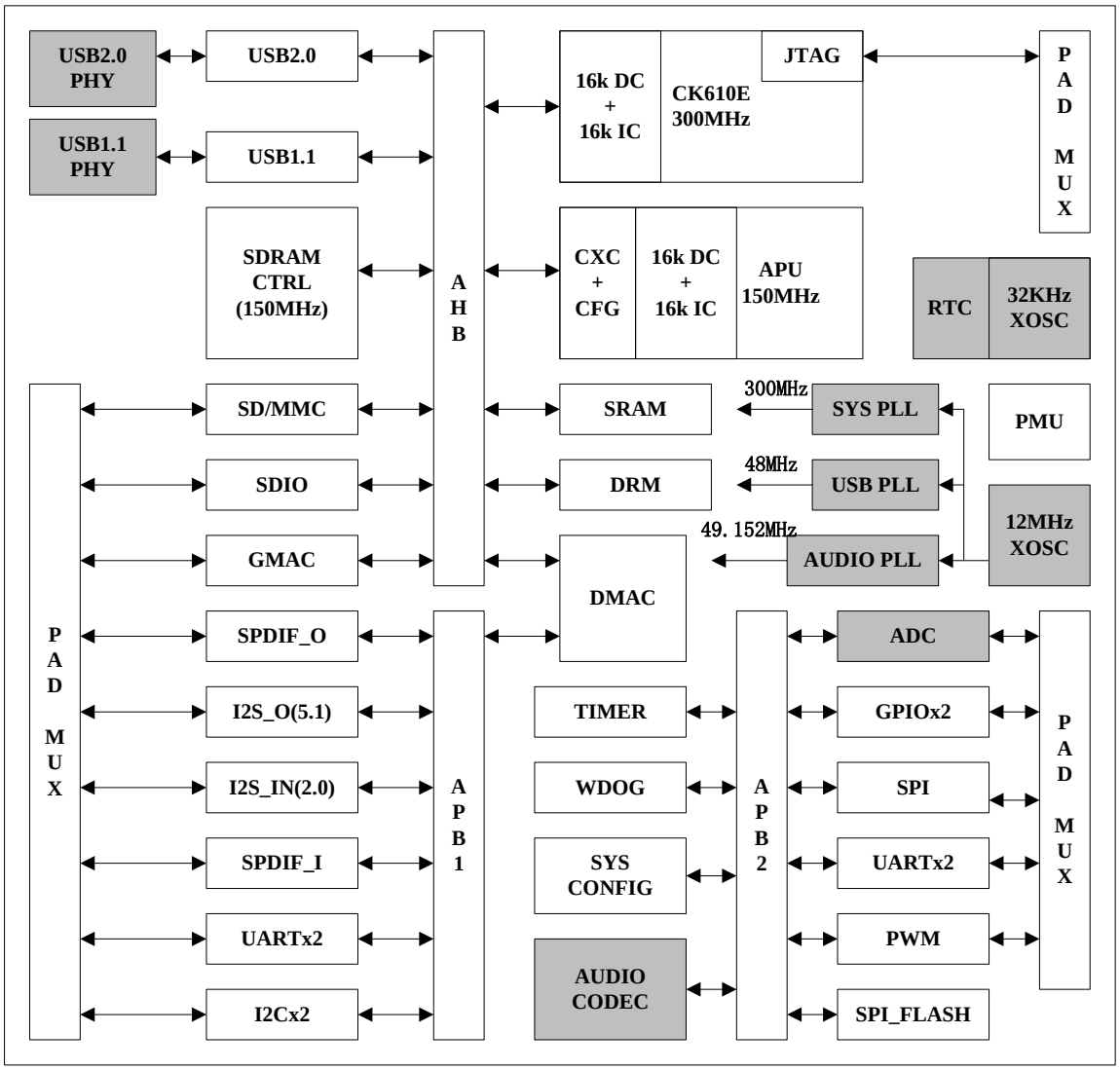


图 3.4 系统结构框图

10.5.2. 主要功能

RISC32 处理器内核来源于 CK610，集成 AMBA 总线控制器，运行频率在 300MHz 以上。程序放在外部 SDRAM，通过 AMBA 总线载入。内部有 16KB 指令 Cache 和 16KB 数据 Cache。

APU 主要负责音频数据处理，包括音频编解码、音频后处理等。其内置 DSP 拥有 16KB 数据 Cache 和 16KB 指令 Cache，专用音频处理加速引擎。

BOOT_LOADER 模块用于在 RISC32 启动前，将初始化代码从芯片外部 SPI FLASH 载入到片内 4KB 大小的 SRAM 中。RISC32 启动直接在 SRAM 上运行 BOOT 程序。

DMAC 模块为集总式的 DMA 控制器，内部硬件 5 个通道，支持任务链表。

USB OTG 模块内置高速 USB2.0 PHY，支持 USB2.0 高速传输，支持 Host/Device Dual Role，内置 DMA 功能。

USB 1.1 模块内置全速 PHY，支持 USB1.1 全速传输，支持 Host/Device Dual Role，内置 DMA 功能。

SD/MMC/SDIO 模块，是一个兼容 SD，MMC，eMMC，SDIO 等协议的控制器，可以控制 SD、MMC 卡的读写，内置 DMA 功能。

SDRAM 控制器具有多路数据通道，x16 外部数据位宽，支持 SDRAM 低功耗模式。

I2S 模块两个，分别面向 5.1 DAC，5.1 ADC，支持 DMAC 进行数据搬运。

SPDIF 模块为主机模式，用于 S/PDIF 协议音频数据的发送，支持 DMAC 进行数据搬运。

Timer 模块是包含 3 个独立的计时器模块，可以预置计数，和设置最大计数值，每个计数器 32 位。

UART 模块内置 64byte 缓存，波特率可以达到 1MHz，支持 DMAC 进行数据搬运。

SYS CTRL 模块是整个芯片的系统级功能寄存器控制模块，涉及到时钟，复位，模拟模块，IO 复用等配置。

SPI CTRL 模块是通用的支持 SPI 协议的控制器，通过软件编程，硬件辅助，实现功能。

SPI FLASH CTRL 模块控制外部的 SPI FLASH。支持软件控制读写 SPI FLASH 和硬件直接读取 SPI FLASH 两种模式。RISC 可以该模块直接在 SPI flash 上运行指令以及读取数据。

10.5.3. RISC 描述

10.5.3.1. 主要功能

RISC 处理器的主要特点如下：

- 高性能的 32 位数据长度， 16 位指令长度；
- 精简指令集计算机结构（RISC）；
- 8 级流水线，完全对软件透明；
- 双发射指令，乱序执行；
- 设计功耗 0.4mW/MHz；
- 支持 AMBA2.0/AMBA3.0 内部总线与接口；
- 两级跳转预测；
- 支持返回地址栈；
- 高性能片上高速缓存；
- 支持 big endian 和 little endian；
- 支持直写和写回操作的数据 cache；
- 内部硬件调试模块支持片上硬件调试；
- 支持快速中断，支持向量中断和自动向量中断；
- 支持指令重构；

10.5.3.2. 微体系结构

内存管理单元可以让超级用户自由定义 4 个地址空间的访问权限，权限划分为：不可读写/只读/可读写。4 个地址空间可重叠，从而可形成各种访问控制模式。

RISC 包含片内 MMU，其具有 4 表项全相联的数据 μTLB、2 表项全相联的指令 μTLB 和 32 表项 2 路组相联 jTLB。提供 2 级 TLB 匹配：μTLB 提升转换速度，对用户透明；jTLB 提高匹配率，用户可配置。如果在 μTLB 中没有 TLB 失配，1 个时钟周期产生物理地址；如果在 μTLB 中存在 TLB 失配，但在 jTLB 中没有 TLB 失配，3 个时钟周期产生物理地址；如果在 jTLB 中存在 TLB 失配，软件支持再填充 jTLB。

指令提取单元，可以配备高速缓存，采用关键指令先取和发射以及添加指令暂存器等手段来增加指令的发射效率，利用先进二级指令跳转预测、全局和局部结合的记录来减少因为转移指令造成的流水线空闲，预测准确率高达 94%。

存储单元支持不间断流水线，开辟了八个写缓冲区，具有内部 Bypass 机制，利用快速退休加快指令的执行速度。存取指令可以操作字节、半字和字，并可以在字节和半字操作时自动扩展 0。这些存取指令可以流水执行，使得数据吞吐量达到一个周期存取一个数据。

总线接口单元支持突发传送，支持关键字优先的地址访问，可以在不同的系统时钟与 CPU 时钟比例（1:1，1:2）下工作。

标准 JTAG 调试接口支持各种调试方式，包括软件设置断点方式、硬件设置断点方式、单步和多步指令跟踪、跳转指令跟踪等 7 种方式，可以在线调试 CPU、通用寄存器（GPR）、可选择寄存器（AGPR）、协处理器 0（CP0）和内存。

指令执行单元利用指令依赖表格和操作数前馈来有效地处理数据竞争，实现高性能的指令发射。高性能的指令退休通过 8 个退休缓存器组的非序回收，并行回收，按序退休，快速退休来实现。支持原子操作，自动分解 LDM/STM 指令。

算术运算单元包括两个 32 位的算术逻辑单元（ALU），两个桶型移位器和一个流水线乘法器。算术逻辑运算绝大部分可以在一个周期内完成。

16 个通用寄存器用于提供源操作数和存放指令执行结果。寄存器 R0 通常用作当前堆栈指针，寄存器 R15 通常用作链接寄存器存放子程序的返回地址。

ALU 执行标准的 32 位整数操作，支持快速找 1 算法（FF1）。

乘法器支持 16×16 和 32×32 整数乘法，除法器的设计采用了快速算法，当除数与被除数大小比较相近时能够大大减少运算所需的时钟数(3～37 个)。

一个条件/进位位（C）用于条件测试和多于 32 位的算术逻辑运算。一般的，C 位只在显式的测试/比较运算中被改变，而不是正常指令运算的副作用。但是在某些特定的把条件设置和运算结合的指令中可能例外。

中断响应快，16 个可选择寄存器用于减少在中断异常处理时花费的时间。支持矢量和自动矢量中断。

10.5.3.3. 中断控制

RISC32 内核本身具有普通中断和快速中。芯片设计中指采用了普通中断。

为响应较多的外部硬件中断，因此将中断收集模块 PIC 的总中断连接到 RISC 普通中断口上，实现利用 ICTL 模块将外部硬件中断扩展为 32 个，对应列表如下。

中断位	中断源	说明
PIC_IRQ0		
PIC_IRQ1		
PIC_IRQ2	UART3_INTR	UART3 控制器中断触发
PIC_IRQ3	UART4_INTR	UART4 控制器中断触发
PIC_IRQ4	I2C2_INTR	I2C2 控制器中断触发
PIC_IRQ5	ADC_INTR	按键 ADC 控制器中断触发
PIC_IRQ6	RTC_INTR	RTC 控制器中断触发
PIC_IRQ7	I2C1_INTR	I2C1 控制器中断触发
PIC_IRQ8	TO0_INTR	Timer 0 号定时器中断触发
PIC_IRQ9	TO1_INTR	Timer 1 号定时器中断触发
PIC_IRQ10	TO2_INTR	Timer 2 号定时器中断触发
PIC_IRQ11	TO3_INTR	Timer 3 号定时器中断触发
PIC_IRQ12	CODEC_INTR	AUDIO CODEC 中断触发
PIC_IRQ13	MS_INTR	Timer 毫秒计数器中断触发
PIC_IRQ14	S_INTR	Timer 秒计数器中断触发
PIC_IRQ15	OSPDIF_INTR	SPDIF_O 控制器中断触发
PIC_IRQ16	DRM_INTR	DRM 控制器中断触发
PIC_IRQ17	DAC_I2S_INTR	DAC 应用的 I2S 控制器中断触发
PIC_IRQ18	ADC_I2S_INTR	ADC 应用的 I2S 控制器中断触发
PIC_IRQ19	SDMMC_INTR	SDMMC 控制器中断触发
PIC_IRQ20	SDIO_INTR	SDIO 控制器中断触发
PIC_IRQ21	GPIO1_INTR	GPIO1 控制器 1 中断触发
PIC_IRQ22	GPIO2_INTR	GPIO1 控制器 1 中断触发
PIC_IRQ23	ISPDIF_INTR	SPDIF_I 控制器中断触发
PIC_IRQ24	WDOG_INTR	看门狗控制器中断触发
PIC_IRQ25	USBFS_INTR	USB1.1 控制器中断触发

PIC_IRQ26	GMAC_INTR	GMAC 控制器中断触发
PIC_IRQ27	USBHS_INTR	USB2.0 控制器中断触发
PIC_IRQ28	APU_INTR	APU 中断触发
PIC_IRQ29	HS_UART_INTR	高速应用 UART1 控制器中断触发
PIC_IRQ30	LS_UART_INTR	低速应用 UART2 控制器中断触发
PIC_IRQ31	DMAC_INTR	DMA 控制器中断触发

10.5.4. 地址空间

SDRAM 作为运行主存还是将 SPI FLASH 作为运行主存，可以配置出两种总线地址分配空间。

AHB总线SLAVE		大小	物理地址	重映射地址
S8	SPI_FLASH	16MB	0x1900_0000~0x19FF_FFFF	0x0000_0000~0x00FF_FFFF
S2	SDRAM	256MB	0x0000_0000~0x0FFF_FFFF	0x0100_0000~0x0FFF_FFFF
S3	MX_BRIDGE		0x1A00_0000~0x1AFF_FFFF	
S7	DMAC		0x1C00_0000~0x1CFF_FFFF	
S5	USB2.0		0x1D00_0000~0x1DFF_FFFF	
S9	AX_BRIDGE		0x1E00_0000~0x1EFF_FFFF	
S10	USB1.1		0x1F00_0000~0x1F0F_FFFF	
S11	DRM		0x1F10_0000~0x1F1F_FFFF	
S1	BOOT_SRAM		0x1FC0_0000~0x1FC0_0FFF	
S4	SDRAM_CFG		0x1FD0_0000~0x1FD0_00FF	

MX_BRIDGE		大小	物理地址	备注
S0	SYS_CTRL	64KB	0x1A00_0000~0x1A00_FFFF	系统寄存器控制
S1	GPIO1	64KB	0x1A01_0000~0x1A01_FFFF	
S2	GPIO2	64KB	0x1A02_0000~0x1A02_FFFF	
S3	GMAC	64KB	0x1A03_0000~0x1A03_FFFF	以太网控制器
S4	PMU	64KB	0x1A04_0000~0x1A04_FFFF	低功耗管理
S5	SD/MMC	64KB	0x1A05_0000~0x1A05_FFFF	
S6	SDIO	64KB	0x1A06_0000~0x1A06_FFFF	
S7	SPI_FLASH	64KB	0x1A07_0000~0x1A07_FFFF	配置接口
S8	SPI_CTRL	64KB	0x1A08_0000~0x1A08_FFFF	普通SPI控制器
S9	PIC	64KB	0x1A09_0000~0x1A09_FFFF	中断控制器
S10	TIMER	64KB	0x1A0A_0000~0x1A0A_FFFF	
S11	APU_CXC	64KB	0x1A0B_0000~0x1A0B_FFFF	APU/RISC通信
S12	APU_CFG	64KB	0x1A0C_0000~0x1A0C_FFFF	APU运行配置
S13	RTC	4KB	0x1A0D_0000~0x1A0D_0FFF	
S13	WDOG	4KB	0x1A0D_1000~0x1A0D_1FFF	
S13	VLSP	4KB	0x1A0D_2000~0x1A0D_2FFF	
S13	ADC	4KB	0x1A0D_3000~0x1A0D_3FFF	按键SAR_ADC控制器
S14	CODEC	64KB	0x1A0E_0000~0x1A0E_FFFF	音频ADC/DAC
S15	PWM	64KB	0x1A0F_0000~0x1A0F_FFFF	

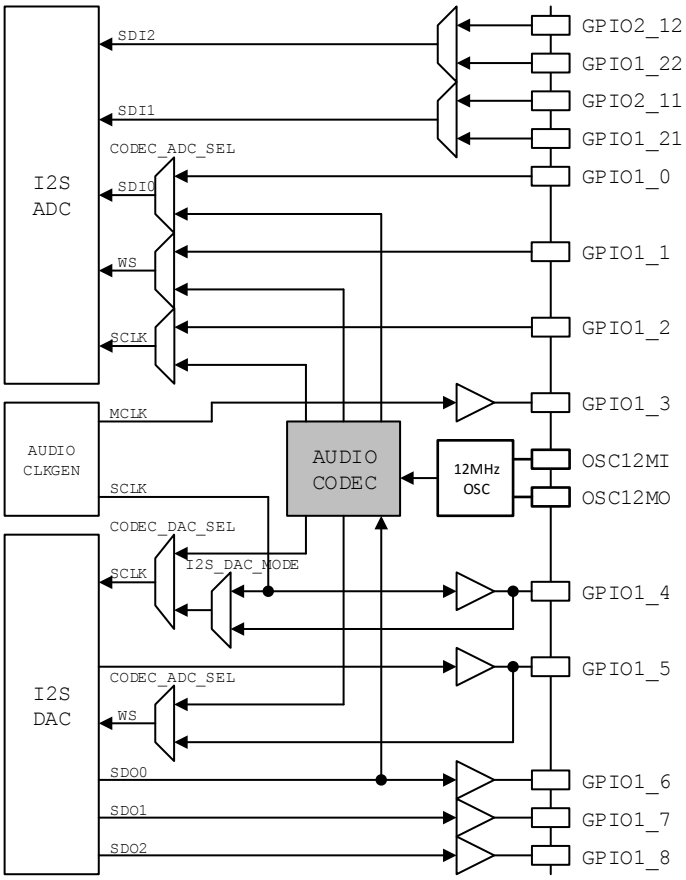
	AX_BRIDGE	大小	物理地址	
S0	UART_HS	64KB	0x1E00_0000~0x1E00_FFFF	高速UART
S1	UART_LS	64KB	0x1E01_0000~0x1E01_FFFF	普通UART
S2	SPDIF_IN	64KB	0x1E02_0000~0x1E02_FFFF	
S3	I2S_DAC	64KB	0x1E03_0000~0x1E03_FFFF	
S4	I2S_ADC	64KB	0x1E04_0000~0x1E04_FFFF	
S5	SPDIF_0	64KB	0x1E05_0000~0x1E05_FFFF	
S6	I2C1	64KB	0x1E06_0000~0x1E06_FFFF	
S7	SPIFLAH_MEM0	16MB	0x0000_0000~0x00FF_FFFF	
S8	SPIFLAH_MEM1	16MB	0x1900_0000~0x19FF_FFFF	
S9	UART3	64KB	0x1E07_0000~0x1E07_FFFF	普通UART
S10	UART4	64KB	0x1E08_0000~0x1E08_FFFF	普通UART
S11		64KB	0x1E09_0000~0x1E09_FFFF	
S12	I2C2	64KB	0x1E0A_0000~0x1E0A_FFFF	

10.5.5. 音频系统

下面章节介绍下音频相关的接口和互联。

10.5.5.1. AUDIO CODEC 和 I2S

内置 CODEC 在 10.4.2 章节中已经描述，下面着重讲述其音频接口（I2S）与芯片的互联关系。



AUDIO CODEC 模块直接采用 12MHz 晶振时钟，内部含有音频 PLL，用于产生音频相关的时钟。其 I2S 接口均是 Master，也就是自己提供 WS 和 SCLK。内部的数字模块 I2S DAC 和 I2S ADC，可以联通到 AUDIO CODEC 或者外部管脚，只能选其一。

详细的 I2S 描述参见 10.6 功能模块相关章节。

10.5.5.2. SPDIF 输入

芯片内置一个 SPDIF 接收模块，通过内部的选择网络，可以选择 8 个管脚中的任一个。

SPDIF_IN_SEL[1:0]	SPDIN_CH1_EN[3:0]	SPDIN_CH2_EN[3:0]	管脚
00	0001	0000	GPIO1_9
00	0000	0001	GPIO2_13
01	0010	0000	GPIO1_10
01	0000	0010	GPIO2_14
10	0100	0000	GPIO1_11

10	0000	0100	GPIO2_15
11	1000	0000	GPIO1_12
11	0000	1000	GPIO2_16

10.6. 功能模块

这里分模块介绍功能模块与接口设备。

10.6.1. PWM

10.6.1.1. 功能特性

实现 8 路独立的 PWM 输出。其中，每两路共用一个总的脉冲周期计数，每个通路的高电平宽度可独立配置。

内置 16 位的预分频，可以对输入时钟进行预分频处理，8 通路有效。每个通路根据预分频时钟进行独立的脉冲宽度调制，调制范围也是 16 位。

10.6.1.2. 寄存器描述

物理基地址：0x1A0F_0000

偏移地址	寄存器名	宽度	描述	读写属性	复位值
0x00	PWMPSC	16	预分频比值	RW	0x0000
0x04	PWM0D	16	通道 0 脉冲输出高电平周期数或者当前通道计数值	RW	0x0000
0x08	PWM1D	16	通道 1 脉冲输出高电平周期数	RW	0x0000
0x0c	PWM2D	16	通道 2 脉冲输出高电平周期数或者当前通道计数值	RW	0x0000
0x10	PWM3D	16	通道 3 脉冲输出高电平周期数	RW	0x0000
0x14	PWM4D	16	通道 4 脉冲输出高电平周期数或者当前通道计数值	RW	0x0000
0x18	PWM5D	16	通道 5 脉冲输出高电平周期数	RW	0x0000
0x1c	PWM6D	16	通道 6 脉冲输出高电平周期数或者当前通道计数值	RW	0x0000
0x20	PWM7D	16	通道 7 脉冲输出高电平周期数	RW	0x0000
0x24	PWMP01	16	通道 0 和通道 1 脉冲输出总周期数	RW	0x0fff
0x28	PWMP23	16	通道 2 和通道 3 脉冲输出总周期数	RW	0x0fff
0x2c	PWMP45	16	通道 4 和通道 5 脉冲输出总周期数	RW	0x0fff
0x30	PWMP67	16	通道 6 和通道 7 脉冲输出总周期数	RW	0x0fff
0x34	PWMCON	16	脉冲使能控制	RW	0x0000

PWM*n*D ： *n* = 0,2,4,6

地址偏移：0x4 + n*4

Field	Name	Description	Access	Reset
15:0	PWM*n*D	通道 0 脉冲输出高电平周期数或者当前通道计数值， PWMCON[n/2]： 0，通道 n 脉冲输出高电平周期数 1， 通道 n 与通道 n+1 的脉冲计数值， 只读	R/W	0x0000

PWMCON

地址偏移：0x34

Field	Name	Description	Access	Reset
11	PWM7EN	通道 7 使能开关，	R/W	0x0
10	PWM6EN	通道 6 使能开关，	R/W	0x0
9	PWM5EN	通道 5 使能开关，	R/W	0x0
8	PWM4EN	通道 4 使能开关，	R/W	0x0
7	PWM3EN	通道 3 使能开关，	R/W	0x0
6	PWM2EN	通道 2 使能开关，	R/W	0x0
5	PWM1EN	通道 1 使能开关，	R/W	0x0
4	PWM0EN	通道 0 使能开关，	R/W	0x0
3	READSEL3	PWM0D 读控制，该值为 1 时，PWM0D 中读取的数值为通道 0 和通道 1 当前的计数值；该值为 0 时，PWM0D 中读取的数值为通道 0 的脉冲输出高电平周期数	R/W	0x0
2	READSEL2	PWM2D 读控制，该值为 1 时，PWM2D 中读取的数值为通道 2 和通道 3 当前的计数值；该值为 0 时，PWM2D 中读取的数值为通道 2 的脉冲输出高电平周期数	R/W	0x0
1	READSEL1	PWM4D 读控制，该值为 1 时，PWM4D 中读取的数值为通道 4 和通道 5 当前的计数值；该值为 0 时，PWM4D 中读取的数值为通道 4 的脉冲输出高电平周期数	R/W	0x0
0	READSEL0	PWM6D 读控制，该值为 1 时，PWM6D 中读取的数值为通道 6 和通道 7 当前的计数值；该值为 0 时，PWM6D 中读取的数值为通道 6 的脉冲输出高电平周期数	R/W	0x0

10.6.2. SDRAM 控制器

SDRAM 控制器 x16 外部数据位宽，支持 SDRAM 低功耗模式。

10.6.2.1. 功能特性

- 支持 SDRAM 控制器与 SDRAM 外部颗粒之间 4 字 BURST 传输
- 支持 Self Reflesh，Power Down 和 Deep Power Down 三种低功耗模式
- 支持 256Mbit~16Mbit 容量，x16 或 x8 的标准 SDRAM 颗粒，也支持 512Kx16 特殊 SDRAM 颗粒

10.6.2.2. 寄存器说明

SDRAM 控制器的配置物理基地址为 0x1FD0_0000。该地址空间上有四个寄存器，用于 SDRAM 控制器的配置以及初始化。寄存器整体映射表如下。

偏移地址	寄存器名	描述	读写属性	复位值
0x00	SDR_REG0	配置寄存器 0	RW	0x00F00000
0x04	SDR_REG1	配置寄存器 1	RW	0x00
0x08	SDR_REG2	刷新计数器寄存器	RW	0x0080

SDR_REG0

区域	名称	类型	功能
31:25,16, 12,8,4,0	-	-	保留，默认值为 0
24	A	RW	AHB 端口访问 SDRAM 自动进行 pre_charge 控制。 0：无效； 1：有效（默认）；
23，22	R[1:0]	RW	SDRAM 的 RAS 到 CAS 延迟。 00：保留； 01：保留； 10：读状态 RAS 到 CAS 延迟=2； 11：读状态 RAS 到 CAS 延迟=3（默认）；
21，20	C[1:0]	RW	CAS 延迟（latency）。 00：保留； 01：CAS latency=1； 10：CAS latency=2； 11：CAS latency=3（默认）；
19	X	RW	外部 SDRAM 总线数据位宽。 0：32（默认）； 1：16；
18	C	RW	SDRAM CKE 使能。 0：CKE 使能控制，节省功耗（默认）； 1：CKE 一直有效为高；
17	E	RW	SDRAM 时钟控制。 0：时钟 CLKOUT 一直输出（默认）； 1：当 SDRAM 空闲状态，时钟 CLKOUT 停止。
15，11，7，3	B[3:0]	RW	SDRAM Bank 选择。 0：2 bank 颗粒； 1：4 bank 颗粒； 实际只有设置 B[0]起作用。
14，10，6，2	T[3:0]	RW	SDRAM 颗粒数据位宽选择。 0：x16 或 x32 颗粒（默认）； 1：x8 颗粒； 实际只有设置 T[0]起作用。
13，9，5，1	F[3:0]	RW	SDRAM 颗粒容量选择。 0：不是 256Mbit 的颗粒（默认）； 1：256Mbit 颗粒； 实际只有设置 F[0]起作用。

SDR_REG1

区域	名称	类型	功能
31:6,4	-	-	保留，默认值为 0
5	B	R	SDRAM 控制器状态。 0：空闲状态；

			1：忙碌状态；
3	W	RW	SDRAM 控制器写缓存使能。 0：无效（默认）； 1：有效； （实际硬件没有写缓存，无效果）
2	R	RW	SDRAM 控制器读缓存使能。 0：无效（默认）； 1：有效；
1	M	RW	SDRAM 颗粒初始化控制位。 IM=00：正常功能（默认）； IM=01：使能 SDRAM Mode Reg Set； IM=10：自动发出 PALL 到 SDRAM； IM=11：自动发出 NOP 到 SDRAM；
0	I	RW	

SDR_REG2

区域	名称	类型	功能
15：0	-	RW	SDRAM 刷新周期对应的 HCLK 时钟个数。 例如：refresh time=16us，HCLK 频率为 50MHz， 则需要配置的数值为：16*50=800。

10.6.2.3. 使用介绍

芯片外部的 SDRAM 颗粒在使用前必须进行初始化，CPU 通过设置 SDRAM 控制器的寄存器，实现初始化过程，具体流程如下：

1、设置 I 和 M 比特位，使控制器自动发出 NOP 到 SDRAM 颗粒；

2、等待 200us；

3、设置 I 和 M 比特位，使控制器自动发出 PRECHARGE ALL 到 SDRAM 颗粒；

4、设置 Refresh timer 寄存器为 10，使控制器每个 10 个时钟发出 refresh 命令；

5、等待大约 80 个时钟周期（即 8 个 refresh 命令周期）；

6、设置实际工作的刷新周期值到 Refresh timer 寄存器；

7、设置 I 和 M 比特位，使控制器进入使能 SDRAM Mode Reg Set，同时 CPU 发起一个读操作，是 HADDR[26:11]的编码对应到 SDRAM 地址线 AddrOut[14:0]输出到 SDRAM 颗粒，进行 mode reg 设置。

8、设置配置寄存器 0，关键参数要与对 SDRAM 颗粒的 mode reg 设置一致。

9、设置 I 和 M 比特位，使控制器进入正常工作模式；

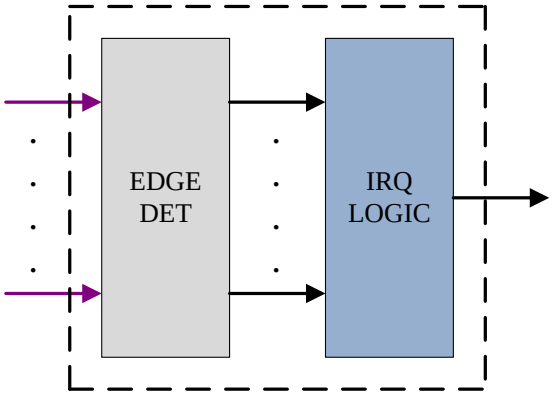
10.6.3. 中断控制器（ICTL）

10.6.3.1. 功能特性

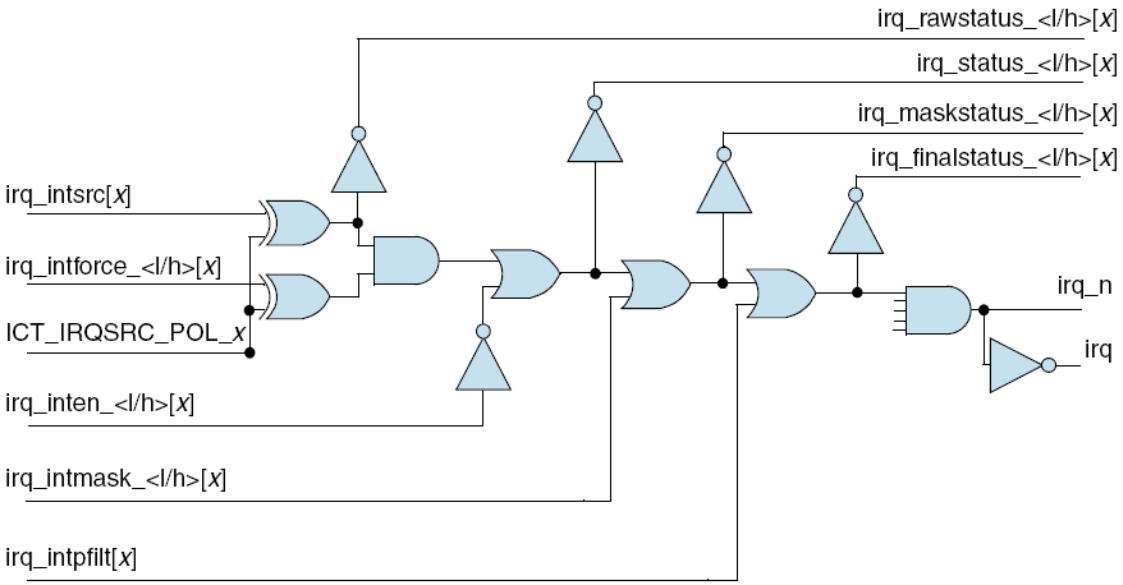
- 支持 32 个中断源
- 软中断支持
- 中断屏蔽支持
- 电平/沿触发可选，上升、下降、双沿可选

10.6.3.2. 模块框图

如下图，中断控制器内部框图。EDGE_DET 主要完成电平/沿中断的产生，IRQ_LOGIC 主要是单中断请求的产生。



如下图是 IRQ_LOGIC 内部的逻辑关系：



ICT_IRQSRC_POL_x 为 1，中断高电平有效。irq_intpfilt[x]无效，为 1。

10.6.3.3. 寄存器说明

基地址：0x1A09_0000

寄存器列表

偏移地址	寄存器名	宽度	描述	读写	复位值
0x000	IRQ_INTEN	32	IRQ 中断源使能	RW	0x00000000
0x004					
0x008	IRQ_INTMASK	32	IRQ 中断源 Mask	RW	0x00000000
0x00c					
0x010	IRQ_INTFOCE		IRQ 软中断	RW	0x00000000
0x014					
0x018	IRQ_RAWSTATUS		IRQ 原始中断状态	R	0x00000000
0x01c					
0x020	IRQ_STATUS		IRQ 使能后中断状态	R	0x00000000
0x024					

0x028	IRQ_MASKSTATUS		IRQ Mask 后中断状态	R	0x00000000
0x02c					
0x030	IRQ_FINALSTATUS		IRQ 最后的中断状态	RW	0x00000000
0x200	IRQ_INTPASS		IRQ 旁路	RW	0xffffffff
0x210	IRQ_EDGEBOTH		IRQ 双沿选择	RW	0x00000000
0x220	IRQ_EDGESEL		IRQ 上升/下降沿选择	RW	0x00000000
0x230	IRQ_INTCLR		IRQ 沿中断清零	W	

IRQ_INTEN

Offset：0x000，IRQ 中断源使能

Field	Name	Description	Access	Reset
31：0	irq_inten	IRQ 中断源使能，每一位控制一个中断源： 0：禁止 1：使能	RW	0x00000000

IRQ_INTMASK

Offset：0x008，IRQ 中断源 Mask

Field	Name	Description	Access	Reset
31：0	IRQ_INTMASK	IRQ 中断源 Mask，每一位 Mask 一个中断： 0：Unmask 中断 1：Mask 中断	RW	0x00000000

IRQ_INTFORCE

Offset：0x010，IRQ 软中断

Field	Name	Description	Access	Reset
31：0	IRQ_INTFORCE	IRQ 软中断，每一位写 1 产生软中断	RW	0x00000000

IRQ_RAWSTATUS

Offset：0x018，IRQ 原始中断状态

Field	Name	Description	Access	Reset
31：0	IRQ_RAWSTATUS	IRQ 原始中断状态，每一位代表一个中断源的状态： 0：无中断 1：有中断	RW	0x00000000

IRQ_STATUS

Offset：0x020，IRQ 原始中断状态

Field	Name	Description	Access	Reset
31：0	IRQ_STATUS	IRQ 使能后中断状态，每一位代表一个中断源的状态： 0：无中断 1：有中断	RW	0x00000000

IRQ_MASKSTATUS

Offset：0x028，IRQ Mask 后中断状态

Field	Name	Description	Access	Reset
31：0	IRQ_MASKSTATUS	IRQ Mask 后中断状态，每一位代表一个中断源的状态： 0：无中断 1：有中断	RW	0x00000000

IRQ_FINALSTATUS

Offset：0x030，IRQ 最终中断状态

Field	Name	Description	Access	Reset
31：0	IRQ_FINALSTATUS	IRQ 最终中断状态，每一位代表一个中断源的状态： 0：无中断 1：有中断	RW	0x00000000

IRQ_INTPASS

Offset：0x200，IRQ 电平/沿中断检测旁路

Field	Name	Description	Access	Reset
31：0	IRQ_INTPASS	IRQ 电平/沿中断检测旁路，每一位控制一个中断源： 0：检测 1：旁路	RW	0xffffffff

IRQ_EDGEBOTH

Offset：0x210，IRQ 双沿检测选择

Field	Name	Description	Access	Reset
31：0	IRQ_EDGEBOTH	IRQ 双沿中断检测选择，每一位控制一个中断源： 0：单沿中断 1：双沿中断	RW	0x00000000

IRQ_EDGESEL

Offset：0x220，IRQ 上升/下降沿选择

Field	Name	Description	Access	Reset
31：0	IRQ_EDGESEL	IRQ 上升/下降沿选择，每一位控制一个中断源： 0：上升沿 1：下降沿	RW	0x00000000

IRQ_EDGECLR

Offset：0x230，IRQ 沿中断清零

Field	Name	Description	Access	Reset
31：0	IRQ_EDGECLR	IRQ 沿中断清零，写 1 清理	W	0x00000000

10.6.4. 系统控制模块

系统控制模块，SYS_CTRL 是系统级配置与状态寄存器的专用控制模块，用于控制系统的各种功能配置以及获取系统状态信息。主要功能包含时钟使能、时钟选择、PLL 设置、软件复位、PHY 配置等。

10.6.4.1. 功能特性

- PLL 模块控制；
- Audio 专用时钟产生配置；
- 功能模块软件复位控制；
- IO 复用配置；
- SDRAM 时钟相位调整，低功耗模式控制；
- USB PHY 功能配置；
- 系统状态信息获取；

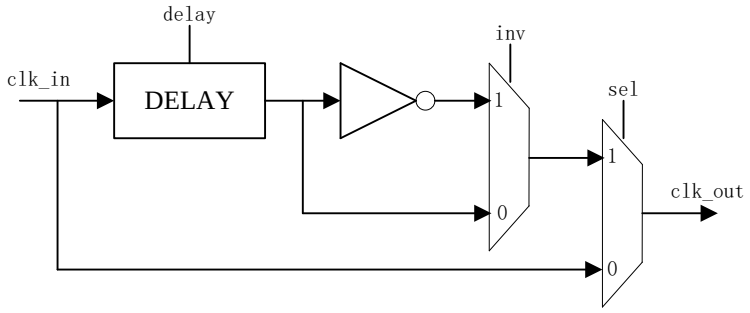
10.6.4.2. 寄存器说明

SYS_CTRL 模块的物理基地址为 0x1A00_0000。整体寄存器地址映射表如下：

偏移	寄存器名	宽度	描述	读写	复位值
0x00	CTR_REG0	32	Reserved	RW	0x00000000
0x04	CTR_REG1	32	SD/SDIO 外部时钟相位配置	RW	0x00000000
0x08	CTR_REG2	32	SD/SDIO/CODEC/ADC 时钟配置	RW	0x00000000
0x0c	CTR_REG3	32	模数混合管脚（GPIO1_9~16）模拟使能	RW	0x00000000
0x10	CTR_REG4	32	音频时钟使能	RW	0x9A000000
0x14	CTR_REG5	32	音频时钟选择	RW	0x00000000
0x18	CTR_REG6	32	功能模块复位 1	RW	0xE0001FFF
0x1c	CTR_REG7	32	管脚复用使能 2	RW	0x00000001
0x20	CTR_REG8	32	系统通用配置 1	RW	0x00000000
0x24	CTR_REG9	32	管脚复用使能 1	RW	0x00000000
0x28					
0x2c	CTR_REG11	32	USB_HS 通用配置	RW	0x00012E08
0x30	CTR_REG12	32	功能模块时钟使能 1	RW	0x00000006
0x34	CTR_REG13	32	功能模块复位 2	RW	0x00000FD2
0x38	CTR_REG14	32	功能模块时钟使能 2	RW	0x00000000
0x3c	CTR_REG15	32	系统通用配置 2	RW	0x00000000
0x40	CTR_REG16	32	USB_FS 电阻控制	RW	0x00000000
0x44	CTR_REG17	32	管脚驱动能力选择 1	RW	0x00000000
0x48	CTR_REG18	32	管脚驱动能力选择 2	RW	0x00000000
0x4c	CTR_REG19	32	管脚驱动能力选择 3	RW	0x00000000
0x50	CTR_REG20	32	Reserved	RW	0x00000000
0x54	CTR_REG21	32	管脚上拉电阻使能 1	RW	0x00000000
0x58	CTR_REG22	32	管脚上拉电阻使能 2	RW	0x00000000
0x80					
0x90					
0x94					
0x98					
0x9c					

CTR_REG1（SD/SDIO 外部时钟相位配置）

offset：0x04，SD/SDIO 外部时钟相位配置



Field	Name	Description	Access	Reset
3:0	SD_DRV_DELAY	SD/MMC 控制器 clk_drv 延时选择 每级大约 0.4ns	R/W	0
4	SD_DRV_INV	SD/MMC 控制器 clk_drv 延时反相选择 0：不反相； 1：反相	R/W	0
5	SD_DRV_SEL	SD/MMC 控制器 clk_drv 选择 0：不经过延时逻辑 1：经过延时逻辑	R/W	0
7：6	RESERVE			
11:8	SD_SAMPLE_DELAY	SD/MMC 控制器 clk_sample 延时选择 每级大约 0.4ns	R/W	0
12	SD_ SAMPLE_INV	SD/MMC 控制器 clk_sample 延时反相选择 0：不反相； 1：反相	R/W	0
13	SD_ SAMPLE_SEL	SD/MMC 控制器 clk_sample 选择 0：不经过延时逻辑 1：经过延时逻辑	R/W	0
15:14	RESERVE			
19:16	SDIO_DRV_DELAY	SDIO 控制器 clk_drv 延时选择 每级大约 0.4ns	R/W	0
20	SDIO_DRV_INV	SDIO 控制器 clk_drv 延时反相选择 0：不反相； 1：反相	R/W	0
21	SDIO_DRV_SEL	SDIO 控制器 clk_drv 选择 0：不经过延时逻辑	R/W	0

Field	Name	Description	Access	Reset
		1：经过延时逻辑		
23：22	RESERVE			
27：24	SDIO_SAMPLE_DELAY	SDIO 控制器 clk_sample 延时选择 每级大约 0.4ns	R/W	0
28	SDIO_ SAMPLE_INV	SDIO 控制器 clk_sample 延时反相选择 0：不反相； 1：反相	R/W	0
29	SDIO_ SAMPLE_SEL	SDIO 控制器 clk_sample 选择 0：不经过延时逻辑 1：经过延时逻辑	R/W	0
31:30	RESERVE			

CTR_REG2（SD/SDIO/CODEC/ADC 时钟配置）

offset：0x08，SD/SDIO/CODEC/ADC 时钟配置

Field	Name	Description	Access	Reset
7:0	RESERVE			
10:8	SD_CCLK_SEL	SD/MMC 控制器 cclk 时钟频率选择： 0：2 分频； 1：2 分频； 2：3 分频； 3：4 分频； 4：5 分频； 5：6 分频； 6：8 分频； 7：16 分频；	R/W	0
11	SD_CCLK_EN	SD/MMC 控制器 cclk 时钟使能： 0：时钟关闭 1：时钟打开	R/W	1
14:12	SDIO_CCLK_SEL	SDIO 控制器 cclk 时钟频率选择： 0：2 分频； 1：2 分频； 2：3 分频； 3：4 分频； 4：5 分频； 5：6 分频；	R/W	0

Field	Name	Description	Access	Reset
		6：8 分频； 7：16 分频；		
15	SDIO_CCLK_EN	SDIO 控制器 cclk 时钟使能： 0：时钟关闭 1：时钟打开	R/W	1
18:16	MC_CCLK_SEL	CODEC 配置接口时钟频率选择： 0：2 分频； 1：2 分频； 2：3 分频； 3：4 分频； 4：5 分频； 5：6 分频； 6：8 分频； 7：16 分频；	R/W	0
19	MC_CCLK_EN	CODEC 配置接口时钟使能： 0：时钟关闭 1：时钟打开	R/W	0
21:20	ADC_SEL	按键 ADC 转换时钟选择： 00：12M/20 01：12M/20/2 10：12M/20/3 11：12M/20/4	RW	0
31:22	RESERVE			

CTR_REG3（模数混合管脚模拟使能）

offset：0x0c，模数混合管脚模拟使能，按键 ADC 的模拟输入使能

Field	Name	Description	Access	Reset
0	ADC_IN0_EN	管脚 GPIO1_9 按键 ADC 输入 IN0 使能	R/W	0
1	ADC_IN1_EN	管脚 GPIO1_10 按键 ADC 输入 IN1 使能	R/W	0
2	ADC_IN2_EN	管脚 GPIO1_11 按键 ADC 输入 IN2 使能	R/W	0
3	ADC_IN3_EN	管脚 GPIO1_12 按键 ADC 输入 IN3 使能	R/W	0
4	ADC_IN4_EN	管脚 GPIO1_13 按键 ADC 输入 IN4 使能	R/W	0
5	ADC_IN5_EN	管脚 GPIO1_14 按键 ADC 输入 IN5 使能	R/W	0
6	ADC_IN6_EN	管脚 GPIO1_15 按键 ADC 输入 IN6 使能	R/W	0

Field	Name	Description	Access	Reset
7	ADC_VREFO_EN	管脚 GPIO1_16 按键 ADC 参考电压使能输出	R/W	0
31：8	reserve			

CTR_REG4（音频时钟使能）

offset：0x10，音频时钟使能

ch1=DAC=I2S Out

ch2=ADC=I2S In

Field	Name	Description	Access	Reset
0	RESERVE		R/W	0
1	CH1_BCLK_EN	通道 1 产生的 bclk 输出使能， 0：输出无效 1：输出有效	R/W	0
2	CH1_MCLK_EN	通道 1 产生的 mclk 输出使能， 0：输出无效 1：输出有效	R/W	0
3	SPDIF_EN	spdif 的 mclk 输出使能， 0：输出无效 1：输出有效	R/W	0
8:4	RESERVE		R/W	0
9	CH2_BCLK_EN	通道 2 产生的 bclk 输出使能， 0：输出无效 1：输出有效	R/W	0
10	CH2_MCLK_EN	通道 2 产生的 mclk 输出使能， 0：输出无效 1：输出有效	R/W	0
31:11	RESERVE		R/W	0

CTR_REG5（音频时钟选择）

offset：0x14，音频时钟选择

ch1=DAC=I2S Out

ch2=ADC=I2S In

ch3=SPDIF

Field	Name	Description	Access	Reset
0	RESERVE		R/W	0
2:1	CH1_BCLK_SEL	通道 1 产生 bit clk 的时钟分频, 00: 1/1 01: 1/2 10: 1/4 11: 1/8	R/W	00
3	CH1_MCLK_SEL	通道 1 产生 mclk 的时钟分频, 0: 1/1 1: 1/2	R/W	0
5:4	CH1_SAMPLE_SEL	通道 1 产生 256fs 的时钟分频, 00: 1/1 01: 1/2 10: 1/4 11: 1/8	R/W	00
6	CH1_TYPE_SEL	通道 1 选择固定的 2 或 3 时钟分频， 0: 1/2，用于 44.1K 或 48K 采样频率系列 1: 1/3，用于 32K 采样频率系列	R/W	0
8:7	RESERVE			
10:9	CH2_BCLK_SEL	通道 2 产生 bit clk 的时钟分频, 00: 1/1 01: 1/2 10: 1/4 11: 1/8	R/W	00
11	CH2_MCLK_SEL	通道 2 产生 mclk 的时钟分频, 0: 1/1 1: 1/2	R/W	0
13:12	CH2_SAMPLE_SEL	通道 2 产生 256fs 的时钟分频, 00: 1/1 01: 1/2 10: 1/4 11: 1/8	R/W	00
14	CH2_TYPE_SEL	通道 2 选择固定的 2 或 3 时钟分频， 0: 1/2，用于 44.1K 或 48K 采样频率系列 1: 1/3，用于 32K 采样频率系列	R/W	0
19:15	RESERVE			
21:20	CH3_SAMPLE_SEL	通道 3 产生 256fs 的时钟分频,	R/W	00

Field	Name	Description	Access	Reset
		00: 1/1 01: 1/2 10: 1/4 11: 1/8		
22	CH3_TYPE_SEL	通道 3 选择固定的 2 或 3 时钟分频， 0: 1/2，用于 44.1K 或 48K 采样频率系列 1: 1/3，用于 32K 采样频率系列	R/W	0
31:23	RESERVE		R/W	0

CTR_REG6（功能模块复位 1）

offset：0x18，功能模块复位 1

Field	Name	Description	Access	Reset
0	OSPDIF_MRSTN	SPDIF_O 模块 mclk 时钟域复位，0 有效，需要软件释放	R/W	1
1	OSPDIF_PRSTN	SPDIF_O 模块 pclk 时钟域复位，0 有效，需要软件释放	R/W	1
2	ISPDIF_MRSTN	SPDIF_I 模块 mclk 时钟域复位，0 有效，需要软件释放	R/W	1
3	ISPDIF_PRSTN	SPDIF_I 模块 pclk 时钟域复位，0 有效，需要软件释放	R/W	1
4	I2S_DAC_MRSTN	用于 DAC 的 I2S 模块 sclk 时钟域复位，0 有效，需要软件释放	R/W	1
5	I2S_ADC_MRSTN	用于 ADC 的 I2S 模块 sclk 时钟域复位，0 有效，需要软件释放	R/W	1
6	MC_RSTN	audio codec 配置模块复位，0 有效，需要软件释放	R/W	1
7	CODEC_RSTN	audio codec 复位，0 有效，需要软件释放	R/W	1
8	I2S_DAC_PRSTN	用于 DAC 的 I2S 模块 pclk 时钟域复位，0 有效，需要软件释放	R/W	1
9	I2S_ADC_PRSTN	用于 ADC 的 I2S 模块 pclk 时钟域复位，0 有效，需要软件释放	R/W	1
10	DRM_RSTN	用于 DRM 模块复位，0 有效，需要软件释放	R/W	1
11	HS_UART_SRSTN	用于 high speed 的 UART 模块复位，0 有效，需要软件释放	R/W	1

Field	Name	Description	Access	Reset
12	LS_UART_SRSTN	用于 low speed 的 UART 模块复位，0 有效，需要软件释放	R/W	1
13	GPIO1_RSTN	用于 GPIO1 模块复位，0 有效，需要软件释放	R/W	1
14	GPIO2_RSTN	用于 GPIO2 模块复位，0 有效，需要软件释放	R/W	1
15	USBHS_PRSTN	USB_HS 模块 UTMI 时钟域的复位，0 有效，需要软件释放。	R/W	1
16	USBGS_HRSTN	USB_HS 模块 HCLK 时钟域的复位，0 有效，需要软件释放。	R/W	1
17	USBFS_PRSTN	USB_FS 模块 UTMI 时钟域的复位，0 有效，需要软件释放。	R/W	1
18	USBFS_HRSTN	USB_FS 模块 HCLK 时钟域的复位，0 有效，需要软件释放。	R/W	1
19	SDMMC_HRSTN	SDMMC 模块中 HCLK 时钟域的复位，0 有效，需要软件释放	R/W	1
20	SDIO_HRSTN	SDIO 模块中 HCLK 时钟域的复位，0 有效，需要软件释放	R/W	1
21	DMAC_HRSTN	DMAC 模块中 HCLK 时钟域的复位，0 有效，需要软件释放	R/W	1
22	SDRAM_HRSTN	SDRAM 模块中 HCLK 时钟域的复位，0 有效，需要软件释放	R/W	1
23	GMAC_HRSTN	GMAC AHB 总线接口的复位，0 有效，需要软件释放	R/W	1
24	I2C1_PRSTN	I2C1 APB 总线接口的复位，0 有效，需要软件释放	R/W	1
25	TIMER_RSTN	Timer 模块复位，0 有效，需要软件释放	R/W	1
26	SPI_RSTN	SPI Normal 复位，0 有效，需要软件释放	R/W	1
27	DSP_CRSTN	DSP 内核时钟域复位，0 有效，需要软件释放	R/W	1
28	DSP_PRSTN	DSP 接口时钟域复位，0 有效，需要软件释放	R/W	1
29	SRAM_RSTN	SRAM 模块复位，0 有效，需要软件释放	R/W	1
30	BOOT_RSTN	BOOT 模块复位，0 有效，需要软件释放	R/W	1
31	PMU_RSTN	PMU 模块复位，0 有效，需要软件释放	R/W	1

CTR_REG7（管脚复用使能 2）

offset：0x1c，管脚复用使能 2

默认通道 1 优先级最大。

Field	Name	Description	Access	Reset
7:0	PWM_CH1_EN	PWM 模块 8 路输出，通道 1 复用使能，1 有效	R/W	0
11:8	SPDIN_CH2_EN	SPDIF_I 模块 4 路输入，通道 2 复用使能，1 有效	R/W	0
13：12	ADC_CH2_EN	ADC I2S SDI1/SDI2 输入，通道 2 复用使能，1 有效	R/W	0
14	UARTHS_CH2_EN	高速 UART 模块，通道 2 复用使能，1 有效	R/W	0
15	SDIO_DET _CH2_EN	SDIO 控制器 detect 信号通道 2 复用使能，1 有效		
16	I2S_ADC_MODE	I2S ADC 模块工作模式， 0：Slave 模式，SCLK/WS 输入； 1：Master 模式，SCLK/WS 输出；		
17	I2S_DAC_MODE	I2S DAC 模块工作模式， 0：Slave 模式，SCLK/WS 输入； 1：Master 模式，SCLK/WS 输出；		
18	CODEC_ADC_SEL	I2S ADC SCLK/WS 选择， 0：不选择 codec； 1：选择 codec sclk 输出；		
19	CODEC_DAC_SEL	I2S DAC SCLK/WS 选择， 0：不选择 codec； 1：选择 codec sclk 输出；		
21：20	SPDIF_IN_SEL	SPDIF_I 模块输入选择， 00：SPDIF_I0 01：SPDIF_I1 10：SPDIF_I2 11：SPDIF_I3		
22	I2S_ADC_SCLK_INV	I2S ADC SCLK 反向选择		
23	I2S_DAC_SCLK_INV	I2S DAC SCLK 反向选择		
26:24	PWM_CH2_EN	PWM 模块 8 路输出，通道 2 复用使能，1 有效 24：pwm[2]； 25：pwm[3]； 26：pwm[1]；	R/W	0
27	SDIO_CH2_EN	SDIO 模块 CLK/CMD/D0，通道 2 复用使能，1 有效	R/W	0
28	SDIO_DAT_CH2_EN	SDIO 模块 D1~3，通道 2 复用使能，1 有效	R/W	0
29~30				
31	JTAG_EN	RISC JTAG 接口使能	R/W	0

CTR_REG8（系统通用配置 1）

offset：0x20，系统通用配置 1

Field	Name	Description	Access	Reset
0	GMAC_MAC_SPEED	GMAC 网络速度选择： 0：10Mbps； 1：100Mbps；	R/W	0
1	GMAC_PHY_INTF	GMAC 外部 PHY 接口选择： 只能为 1，RMII 接口。	R/W	1
2	BOOTSRAM_CSN	BOOT 模块 SRAM 片选	R/W	0
8：3	RESERVE			
9	RUN_FLASH	运行在 FLASH 使能	R/W	0
16：10	RESERVE			
17	PDN_REQ	SDRAM 控制器低功耗请求控制，1 有效	R/W	0
18	DPDN_REQ	SDRAM 控制器超低功耗请求，1 有效	R/W	0
31:19	RESERVE			

CTR_REG9（管脚复用使能 1）

offset：0x24，管脚复用使能 1

Field	Name	Description	Access	Reset
0	UART_HS_EN	高速 UART 模块的复用使能，1 有效	R/W	0
1	UART_LS_EN	低速 UART 模块的复用使能，1 有效	R/W	0
2	DAC_MCLK_EN	用于 DAC 的 mclk 输出复用使能，1 有效	R/W	0
3	UART4_CH1_EN	UART4 通道 1 复用使能，1 有效	R/W	0
4	UART4_CH2_EN	UART4 通道 2 复用使能，1 有效	R/W	0
5	SPI_MIN_CH1_EN	通用 SPI 控制器 CS/CLK/MOSI 通道 1 复用使能，1 有效	R/W	0
6	SPI_IN_CH1_EN	通用 SPI 控制器 MISO 通道 1 复用使能，1 有效	R/W	0
7	SDMMC_EN	SD/MMC 控制器基本复用使能，1 有效	R/W	0
8	SDIO_CH1_EN	SDIO 控制器基本通道 1 复用使能，1 有效	R/W	0
9	SDMMC_DET_EN	SD/MMC 控制器 detect 信号复用使能，1 有效	R/W	0
10	SDIO_DET _CH1_EN	SDIO 控制器 detect 信号通道 1 复用使能，1 有效	R/W	0

Field	Name	Description	Access	Reset
11	SF_DAT_EN	SPI FLASH 控制器 IO2/3 复用使能	R/W	0
12	ADC_EN	用于 ADC 的 I2S 模块中 sclk，ws 复用使能，1 有效	R/W	0
13	UART3_CH1_EN	UART3 通道 1 复用使能，1 有效	R/W	0
14	UART3_CH2_EN	UART3 通道 2 复用使能，1 有效	R/W	0
15	DAC_EN	用于 DAC 的 I2S 模块中 sclk，ws 复用使能，1 有效	R/W	0
18：16	DAC_FD_EN	用于 DAC 的 I2S 模块中 sd0/1/2 复用使能，1 有效	R/W	000
21：19	ADC_FD_EN	用于 ADC 的 I2S 模块中 sd0/1/2 复用使能，1 有效	R/W	000
22	SPDIF_EN	SPDIF 模块的复用使能，1 有效	R/W	0
23	SDMMC_DAT_EN	SD/MMC 数据 D1/2/3 复用使能，1 有效	R/W	0
24	SDIO_DAT_CH1_EN	sdio 数据 D1/2/3 通道 1 复用使能，1 有效	R/W	0
25	I2C2_EN	I2C2 控制器复用使能，1 有效	R/W	0
26	GMAC_EN	GMAC 控制器复用使能，1 有效	R/W	0
27	I2C1_EN	I2C1 控制器复用使能，1 有效	R/W	0
31：28	SPDIN_CH1 _EN	SPDIF_I 模块通道 1 复用使能，1 有效	R/W	0

CTR_REG11（USB_HS 通用配置）

offset：0x2c，USB_HS 通用配置

Field	Name	Description	Access	Reset
0	RESERVE		R/W	0
1	USBHS_UTMISRP_BVALID	USBHS 控制器 utmisrp_bvalid 信号反相	R/W	0
2	USBHS_UTMIOTG_AVALID	USBHS 控制器 utmiotg_avalid 信号		
3	TX_ENABLE_N	fs_data_ext 和 fs_se0_ext 输出使能。 低电平有效。	R/W	1
4	USBHS_UTMI_HOSTDISCONNECT	USBHS 控制器 utmi_ hostdisconnect 信号	R/W	0
5	USBHS_HOSTDISCONNECT_SEL	选择提供给 USBHS 控制器的 utmi_ hostdisconnect。 1：来自系统寄存器控制 usbhs_utmi_hostdisconnect； 0：来自 USB2.0 PHY 的输出；	R/W	0
15：6	RESERVE		R/W	0
16	USBHS_VBUSVALID_SEL	选择提供给 USBHS 控制器的 utmiotg_vbusvalid。 1：来自系统寄存器控制 usbhs_utmiotg_vbusvalid。 0：来自 USB2.0 PHY 的输出。		

Field	Name	Description	Access	Reset
17	USBHS_UTMIOTG_VBUSVALID	USBHS 控制器 utmiotg_vbusvalid 信号	R/W	0
18	USBHS_DISCHRGVBUS_SEL	选择提供给 USBHS 控制器的 otg_dischrgvbus。 1：来自系统寄存器控制 usbhs_otg_dischrgvbus。 0：来自 USB2.0 PHY 的输出。	R/W	0
19	USBHS_OTG_DISCHRGVBUS	USBHS 控制器 otg_dischrgvbus 信号	R/W	0
20	USBHS_CHRGVBUS_SEL	选择提供给 USBHS 控制器的 otg_chrgvbus。 1：来自系统寄存器控制 usbhs_otg_chrgvbus。 0：来自 USB2.0 PHY 的输出。	R/W	0
21	USBHS_OTG_CHRGVBUS	USBHS 控制器 otg_chrgvbus 信号	R/W	0
22	USBHS_IDDIG_SEL	选择提供给 USBHS 控制器的 utmiotg_iddig。 1：来自系统寄存器控制 usbhs_utmi_iddig。 0：来自 USB2.0 PHY 的输出。	R/W	0
23	USBHS_UTMI_IDDIG	USBHS 控制器 iddig 信号	R/W	0
24	USBFS_UTMI_IDDIG	USBFS 控制器 iddig 信号		
31：25	RESERVE		R/W	0

CTR_REG12（功能模块时钟使能 1）

offset：0x30，功能模块时钟使能 1

Field	Name	Description	Access	Reset
0	GPIO1_CLKON	GPIO1 模块时钟使能	R/W	1
1	GPIO2_CLKON	GPIO2 模块时钟使能	R/W	1
2	I2S0_CLKON	I2S_O（DAC）模块时钟使能	R/W	1
3	I2S1_CLKON	I2S_I（ADC）模块时钟使能	R/W	1
4	GMAC_CLKON	GMAC 以太网模块时钟使能	R/W	1
5	UARTLS_CLKON	UART_LS 模块时钟使能	R/W	1
6	UARTHS_CLKON	UART_HS 模块时钟使能	R/W	1
7	I2C_CLKON	I2C 模块时钟使能	R/W	1
8	SDIO_CLKON	SDIO 模块时钟使能	R/W	1
9	SDMMC_CLKON	SD/MMC 模块时钟使能	R/W	1
10	SPDIFO_CLKON	SPDIF_O 模块时钟使能	R/W	1
11	SPDIFI_CLKON	SPDIF_I 模块时钟使能	R/W	1

Field	Name	Description	Access	Reset
12	SPI_CLKON	SPI_Normal 模块时钟使能	R/W	1
13	TIMER_CLKON	TIMER 模块时钟使能	R/W	1
14	DSPC_CLKON	DSP 模块核心时钟使能	R/W	1
15	DSPP_CLKON	DSP 模块总线时钟使能	R/W	1
16	DMAC_CLKON	DMAC 模块时钟使能	R/W	1
17	DRM_CLKON	DRM 模块时钟使能	R/W	1
18	USBHS_CLKON	USB_HS 模块时钟使能	R/W	1
19	USBFS_CLKON	USB_FS 模块时钟使能	R/W	1
20	SDRAM_CLKON	SDRAM 模块时钟使能	R/W	1
21	SRAM_CLKON	SRAM 模块时钟使能	R/W	1
22	BOOT_CLKON	Bootloader 模块时钟使能	R/W	1
23	CPUS_CLKON	CPU 总线时钟使能	R/W	1
24			R/W	1
25	PMU_CLKON	PMU 模块时钟使能	R/W	1
26	VLSPP_CLKON	VLSP 模块总线时钟使能	R/W	1
27	RTC_CLKON	RTC 模块配置时钟使能	R/W	1
28	PWM_CLKON	PWM 模块时钟使能	R/W	1
29	SPDIFIM_CLKON	SPDIF 模块音频时钟使能	R/W	1
30	ADC_CLKON	按键 ADC 模块时钟使能	R/W	1
31	UART3_CLKON	UART3 模块时钟使能	R/W	1

CTR_REG13（功能模块复位 2）

offset：0x34，功能模块复位 2

Field	Name	Description	Access	Reset
0	USBPHY_RSTN	USB_HS PHY 复位，0 有效，需要软件释放	R/W	1
1	VLSP_CRSTN	VLSP 模块 cclk 时钟域复位，0 有效，需要软件释放	R/W	1
2	VLSP_PRSTN	VLSP 模块 pclk 时钟域复位，0 有效，需要软件释放	R/W	1
3	PWM_PRSTN	PWM 模块 pclk 时钟域复位，0 有效，需要软件释放	R/W	1
4	ADC_RSTN	按键 ADC 控制器复位，0 有效，需要软件释放	R/W	1

Field	Name	Description	Access	Reset
5	UART3_RSTN	UART3 模块复位，0 有效，需要软件释放	R/W	1
6	UART4_RSTN	UART4 模块复位，0 有效，需要软件释放	R/W	1
7	PWM_CRSTN	PWM 模块 cclk 时钟域复位，0 有效，需要软件释放	R/W	1
31:8			R/W	1

CTR_REG14（功能模块时钟使能 2）

offset：0x38，功能模块时钟使能 2

Field	Name	Description	Access	Reset
0	UART4_CLKON	UART4 模块时钟使能	R/W	0
31:1			R/W	0

CTR_REG15（系统通用配置 2）

offset：0x3c，系统通用配置 2

Audio PLL 配置详见 PLL 模块。

Field	Name	Description	Access	Reset
15~0				
16	UPLL_EN	USB 48M PLL 使能	R/W	1’b1
17	APLL_EN	Audio PLL 使能	R/W	1’b1
18	UPLL_RSTN	USB 48M PLL 复位，0 有效，软件释放	R/W	1’b1
19	APLL_RSTN	Audio PLL 复位，0 有效，软件释放	R/W	1’b1
28:20	APLL_DN	Audio PLL 微调控制位	R/W	
29	APLL_CS	Audio PLL 微调使能	R/W	1’b0
30	WDOG_CLKEN	WDOG 模块时钟使能	R/W	1’b0
31	APLL_MODE	Audio PLL 频率模式 0:45MHz 1:49MHz	R/W	1’b0

CTR_REG16（USB_FS 上下拉电阻控制）

offset：0x40，USB_FS 电阻控制

Field	Name	Description	Access	Reset
4:0	USBFS_DM_RES_TRIM	USBFS PHY D-电阻阻值调节	R/W	4'h0
5	USBFS_DM_RESUP_EN	USBFS PHY D-上拉电阻使能	R/W	1'b0
6	USBFS_DM_RESDN_EN	USBFS PHY D-下拉电阻使能	R/W	1'b0
7	RESERVE			
12:8	USBFS_DP_RES_TRIM	USBFS PHY D+电阻阻值调节	R/W	4'h0
13	USBFS_DP_RESUP_EN	USBFS PHY D+上拉电阻使能	R/W	1'b0
14	USBFS_DP_RESDN_EN	USBFS PHY D+下拉电阻使能	R/W	1'b0
31：15	RESERVE			

CTR_REG17（管脚驱动能力选择 1）

offset：0x44，管脚驱动能力选择 1

00：2mA

01：4mA

10：8mA

11：24mA

Field	Name	Description	Access	Reset
1:0	GPIO1_0_DS	GPIO1_0 驱动能力选择	R/W	2'b00
3:2	GPIO1_1_DS	GPIO1_1 驱动能力选择	R/W	2'b00
5:4	GPIO1_2_DS	GPIO1_2 驱动能力选择	R/W	2'b00
7:6	GPIO1_3_DS	GPIO1_3 驱动能力选择	R/W	2'b00
9:8	GPIO1_4_DS	GPIO1_4 驱动能力选择	R/W	2'b00
11:10	GPIO1_5_DS	GPIO1_5 驱动能力选择	R/W	2'b00
13:12	GPIO1_6_DS	GPIO1_6 驱动能力选择	R/W	2'b00
15:14	GPIO1_7_DS	GPIO1_7 驱动能力选择	R/W	2'b00
17:16	GPIO1_8_DS	GPIO1_8 驱动能力选择	R/W	2'b00
19:18	GPIO1_9_DS	GPIO1_9 驱动能力选择	R/W	2'b00
21:20	GPIO1_10_DS	GPIO1_10 驱动能力选择	R/W	2'b00
23:22	GPIO1_11_DS	GPIO1_11 驱动能力选择	R/W	2'b00
25:24	GPIO1_12_DS	GPIO1_12 驱动能力选择	R/W	2'b00
27:26	GPIO1_13_DS	GPIO1_13 驱动能力选择	R/W	2'b00

Field	Name	Description	Access	Reset
29:28	GPIO1_14_DS	GPIO1_14 驱动能力选择	R/W	2'b00
31:30	GPIO1_15_DS	GPIO1_15 驱动能力选择	R/W	2'b00

CTR_REG18（管脚驱动能力选择 2）

offset：0x48，管脚驱动能力选择 2

00：2mA

01：4mA

10：8mA

11：24mA

Field	Name	Description	Access	Reset
1:0	GPIO1_16_DS	GPIO1_16 驱动能力选择	R/W	2'b00
3:2	GPIO1_17_DS	GPIO1_17 驱动能力选择	R/W	2'b00
5:4	GPIO1_18_DS	GPIO1_18 驱动能力选择	R/W	2'b00
7:6	GPIO1_19_DS	GPIO1_19 驱动能力选择	R/W	2'b00
9:8	GPIO1_20_DS	GPIO1_20 驱动能力选择	R/W	2'b00
11:10	GPIO1_21_DS	GPIO1_21 驱动能力选择	R/W	2'b00
13:12	GPIO1_22_DS	GPIO1_22 驱动能力选择	R/W	2'b00
15:14	GPIO1_23_DS	GPIO1_23 驱动能力选择	R/W	2'b00
17:16	GPIO1_24_DS	GPIO1_24 驱动能力选择	R/W	2'b00
19:18	GPIO1_25_DS	GPIO1_25 驱动能力选择	R/W	2'b00
21:20	GPIO1_26_DS	GPIO1_26 驱动能力选择	R/W	2'b00
23:22	GPIO1_27_DS	GPIO1_27 驱动能力选择	R/W	2'b00
25:24	GPIO1_28_DS	GPIO1_28 驱动能力选择	R/W	2'b00
27:26	GPIO1_29_DS	GPIO1_29 驱动能力选择	R/W	2'b00
31:28				

CTR_REG19（管脚驱动能力选择 3）

offset：0x4C，管脚驱动能力选择 3

00：2mA

01：4mA

10：8mA

11：24mA

Field	Name	Description	Access	Reset
1:0	GPIO2_0_DS	GPIO2_0 驱动能力选择	R/W	2'b00
3:2	GPIO2_1_DS	GPIO2_1 驱动能力选择	R/W	2'b00
5:4	GPIO2_2_DS	GPIO2_2 驱动能力选择	R/W	2'b00
7:6	GPIO2_3_DS	GPIO2_3 驱动能力选择	R/W	2'b00
9:8	GPIO2_4_DS	GPIO2_4 驱动能力选择	R/W	2'b00
11:10	GPIO2_5_DS	GPIO2_5 驱动能力选择	R/W	2'b00
13:12	GPIO2_6_DS	GPIO2_6 驱动能力选择	R/W	2'b00
15:14	GPIO2_7_DS	GPIO2_7 驱动能力选择	R/W	2'b00
17:16	GPIO2_8_DS	GPIO2_8 驱动能力选择	R/W	2'b00
19:18	GPIO2_9_DS	GPIO2_9 驱动能力选择	R/W	2'b00
21:20	GPIO2_10_DS	GPIO2_10 驱动能力选择	R/W	2'b00
23:22	GPIO2_11_DS	GPIO2_11/ GPIO2_16 驱动能力选择	R/W	2'b00
25:24	GPIO2_12_DS	GPIO2_12/ GPIO2_17 驱动能力选择	R/W	2'b00
27:26	GPIO2_13_DS	GPIO2_13/ GPIO2_18 驱动能力选择	R/W	2'b00
29:28	GPIO2_14_DS	GPIO2_14/ GPIO2_19 驱动能力选择	R/W	2'b00
31:30	GPIO2_15_DS	GPIO2_15/ GPIO2_20 驱动能力选择	R/W	2'b00

CTR_REG21（管脚上拉电阻使能 1）

offset：0x54，管脚上拉电阻使能 1

n=29

Field	Name	Description	Access	Reset
n:0	GPIO1_n_REN	GPIO1_n 上拉电阻使能	R/W	1'b0
31:n+1				

CTR_REG22（管脚上拉电阻使能 2）

offset：0x58，管脚上拉电阻使能 2

n=20

Field	Name	Description	Access	Reset
n:0	GPIO2_n_REN	GPIO2_n 上拉电阻使能	R/W	n+1'd0

Field	Name	Description	Access	Reset
31:n+1				

STS_REG0（系统通用状态）

offset：0x80，系统通用状态

Field	Name	Description	Access	Reset
3:0	CLK_SEL	RISC 系统 PLL 的设置状态， 0001：90MHz 0011：150MHz 0101：180MHz 0111：216MHz 1001：240MHz 1011：264MHz 1101：300MHz 1111：软件配置	RO	4'bxxx1
15：4	RESERVE			
16	SELF_REFLAH_ACK	SDRAM 控制器自刷新应答，1 有效	RO	
17	PDN_ACK	SDRAM 控制器低功耗应答，1 有效	RO	
18	DPDN_ACK	SDRAM 控制器超低功耗应答，1 有效	RO	
31:19	RESERVE			

10.6.4.3. 数字 IO 口复用配置说明

1、不同颜色代表同一功能在不同 IO 口的复用，其中标注 ch1 的优先级高；

2、越靠近左侧，优先级越高；

名称	复用关系					
	模拟	数字				STRAP PIN
GPIO1_0		ADC_SDI0			GPIO1_0	
GPIO1_1		ADC_WS	TXD4		GPIO1_1	
GPIO1_2		ADC_SCLK	RXD4		GPIO1_2	
GPIO1_3		DAC_MCLK			GPIO1_3	
GPIO1_4		DAC_SCLK			GPIO1_4	
GPIO1_5		DAC_WS			GPIO1_5	
GPIO1_6		DAC_SDO0			GPIO1_6	
GPIO1_7		DAC_SDO1		PWM2	GPIO1_7	
GPIO1_8		DAC_SDO2		PWM3	GPIO1_8	
GPIO1_9	ADIN0	SPDIF_I0			GPIO1_9	
GPIO1_10	ADIN1	SPDIF_I1	TRSTn		GPIO1_10	
GPIO1_11	ADIN2	SPDIF_I2	TCK		GPIO1_11	
GPIO1_12	ADIN3	SPDIF_I3	TMS		GPIO1_12	
GPIO1_13	ADIN4	TXD3	TDI		GPIO1_13	
GPIO1_14	ADIN5	RXD3	TDO		GPIO1_14	
GPIO1_15	ADIN6	TXD1	SPI_MOSI		GPIO1_15	
GPIO1_16		RXD1	SPI_MISO		GPIO1_16	
GPIO1_17		SCL1	SPI_CLK		GPIO1_17	
GPIO1_18		SDA1	SPI_CS		GPIO1_18	
GPIO1_19		TXD2	SF_IO2		GPIO1_19	
GPIO1_20		RXD2	SF_IO3		GPIO1_20	
GPIO1_21		ADC_SDI1	TXD1	PWM2	GPIO1_21	
GPIO1_22		ADC_SDI2	RXD1	PWM3	GPIO1_22	
GPIO1_23		SD_DET	PWM1		GPIO1_23	
GPIO1_24		SD_D1			GPIO1_24	
GPIO1_25		SD_D0			GPIO1_25	
GPIO1_26		SD_CLK			GPIO1_26	
GPIO1_27		SD_CMD			GPIO1_27	
GPIO1_28		SD_D3			GPIO1_28	
GPIO1_29		SD_D2			GPIO1_29	
GPIO2_0		ENET_TXD1	SDIO_DET		GPIO2_0	
GPIO2_1		ENET_TXD0	SDIO_D1		GPIO2_1	
GPIO2_2		ENET_TX_EN	SDIO_D0		GPIO2_2	
GPIO2_3		ENET_MDC	SDIO_CLK		GPIO2_3	
GPIO2_4		ENET_MDIO	SDIO_CMD		GPIO2_4	
GPIO2_5		ENET_RX_DV	SDIO_D3		GPIO2_5	
GPIO2_6		ENET_RX_ER	SDIO_D2		GPIO2_6	
GPIO2_7		ENET_RXD0		PWM6	GPIO2_7	
GPIO2_8		ENET_RXD1	TXD4	PWM4	GPIO2_8	
GPIO2_9		ENET_CLK	RXD4	PWM5	GPIO2_9	
GPIO2_10		SPDIF_O		PWM7	GPIO2_10	
GPIO2_11		ADC_SDI1	TXD3	SCL2	GPIO2_11	
GPIO2_12		ADC_SDI2	RXD3	SDA2	GPIO2_12	
GPIO2_13			SPDIF_I0	PWM1	GPIO2_13	
GPIO2_14		SDIO_DET	SPDIF_I1		GPIO2_14	
GPIO2_15		SDIO_D1	SPDIF_I2		GPIO2_15	
GPIO2_16		SDIO_D0	SPDIF_I3		GPIO2_16	
GPIO2_17		SDIO_CLK			GPIO2_17	
GPIO2_18		SDIO_CMD			GPIO2_18	
GPIO2_19		SDIO_D3			GPIO2_19	
GPIO2_20		SDIO_D2			GPIO2_20	
SF_CLK						SPLL_SEL2
SF_CSn						
SF_MISO						SPLL_SEL1
SF_MOSI						SPLL_SEL0

10.6.5. UART

SC6138A 内置 4 个 UART，其中 UART_HS 内置深度为 64 的接受和发送缓存，支持 DMAC 进行数据搬运；其他 UART 的 FIFO 深度为 8。

10.6.5.1. 功能特性

独立的接收缓存和发送缓存。
可编程波特率产生器，支持小数分频，UART 模块时钟大于 3.6864MHz 任意频率时钟均可。
支持全双工操作，具有标准异步通讯比特位（start，stop，parity）。支持 False start 比特检测。

10.6.5.2. 寄存器说明

高速 UART 控制器的配置物理基地址为 0x1E00_0000。具体寄存器说明如下表所示。

偏移地址	寄存器	读写	默认值	功能描述
0x000	UARTDR	RW	0x---	UART Data Register. 发送数据和接收数据均寄存于此。写入时位宽为 8 位，写入一次可触发一次传输。读取时位宽为 12 为，其中[7:0]为接收数据，[11:8]为 Overrun Error, Break Error, Parity Error, Framing Error
0x004	UARTRSR	RW	0x0	Receive Status Register.存储错误标志位及接收数据，写入一次可清空所有错误标志位。 7:4 保留 3 ： Overrun Error 2 ： Break Error 1 ： Parity Error 0 ： Framing Error
0x018	UARTFR	R	0b-10010---	Flag Register 15:9 保留 8 ： Ring indicator (RI) 输入端 nUARTRI 的反相 7 ： Transmit FIFO Empty (TXFE). 6 ： Receive FIFO Full (RXFF) . 5 ： Transmit FIFO Full (TXFF). 4 ： Receive FIFO Empty (RXFE) 以上[7:4]均取决于 UARTLCR_H 的 FIFO 使能位 FEN： 若 FEN=1，则在发送/接收 FIFO 为满/空时置 1； 若 FEN=0，则在 transmit / receive holding 寄存器为满/空时置 1 3 ：BUSY 发送 FIFO 不为空时置 1，直至传输完成后置 0，包括传输停止位。 2 ： Data Carrier Detect (DCD). 输入端 nUARTDCD 的反相 1 ： Data Set Ready (DSR). 输入端 nUARTDSR 的反相

				0 ：Clear to Send (CTS). 输入端 nUARTCTS 的反相
0x024	UARTIBRD	RW	0x000	Integer Baud Rate Register. 分频的整数部分
0x028	UARTFBRD	RW	0x00	Fractional Baud Rate Register. 分频的小数部分
0x02C	UARTLCR_H	RW	0x00	Line Control 15:8 保留 7 SPS Stick Parity Select 0 ：不检查奇偶校验位 1 ：若 EPS=0，则奇偶校验位固定为 1； 若 EPS=1，则奇偶校验位固定为 0； 6:5 WLEN Word Length b11 = 8bits b10 = 7bits b01 = 6bits b00 = 5 bits FEN FIFO Enable. 为 1 时发送和接收 FIFO 使能。为 0 时深度为 1 个字节 STP2 2 Stop Bits. 每帧结束后发送 2 个停止位 2 EPS Even Parity Select. 奇偶校验选择。 0：奇校验，检查数据及校验位中是否有奇数个 1 1：偶校验，检查数据及校验位中是否有偶数个 1 PE Parity Enable 0：关闭，数据帧结尾不加校验位 1：使能 0 Send Break. 置 1 时，UARTTXD 在传输完当前字后保持低电平。执行 break 指令时，软件应将此位置 1 并且持续至少 2 个完整帧。
0x030	UARTCR	R/W	0x0300	Control Register CTSEn Clear to Send hardware flow control enable. 置 1 时，仅在 nUARTCTS 为低时传输 14 RTSEn Ready to Send hardware flow control enable.置 1 时，仅在接收 FIFO 有空间时请求发送数据。 13 Out2 输出端口 nUARTOut2 的反相驱动源。 12 Out1 输出端口 nUARTOut1 的反相驱动源。 11 RTS Request to Send. 输出端口 nUARTRTS 的反相驱动源。 10 DTS Data Transmit Ready. 输出端口 nUARTRDTR 的反相驱动源。 9 RXE Receive Enable. 若传输中使能关闭时，将完成正在传输的字后再停止。需要 SIR 时，也需将此位置 1。 8 TXE Transmit Enable. 若传输中使能关闭时，将完成正在传输的字后再停止。需要 SIR 时，也需将此位置 1。

				7 LBE Loopback Enable. 6:3 保留 SIRLP SIR low-power IrDA mode SIREN SIR ENDEC Enable 0 UARTEN UART Enable. 需要 SIR 时，也需将此位置 1。
0x034	UARTIFLS	RW	0x12	Interrupt FIFO Level Select Register. 15:6 保留 5:3 RXIFLSEL Receive Interrupt FIFO Level Select. 2:0 TXIFLSEL Transmit Interrupt FIFO Level Select 触发发送/接收中断的 FIFO 内的数据量： b000 = 1/8 FIFO b001 = 1/4 FIFO b010 = 1/2 FIFO b011 = 3/4 FIFO b100 = 7/8 FIFO b101~b111 保留

10.6.5.3. 使用介绍

初始化 UART 模块的寄存器。配置如下：

UARTIBRD = 0x0020； //分频比的整数部分。非标准波特率，仅供仿真

UARTFBRD = 0x00； //分频比的小数部分。

UARTLCR_H = 0x0070；设置字长 8 位，打开 FIFO 使能，打开 PEN，其他均关闭。

UARTCR = 0x0301； //打开 RXE, TXE, UARTEN, 其他均关闭。

（注意设置分频比后，必须写一次 UARTLCR_H 才能生效）

分频比 = $F_{UARTCLK}$ / (16*Baud Rate)。例如：所需波特率 230400，UARTCLK = 4MHz，则：分频比 = 4000000 / (16*230400) = 1.085。整数部分为 1，小数部分为 0.085，integer(0.085*64+0.5) = 5，因此：

UARTIBRD = 0x0001；

UARTFBRD = 0x05；

（实际产生的分频比 = 1+5/64 = 1.078，有少量误差。）

10.6.6. SPI FLASH 控制器

SPI FLASH CTRL 为专用功能模块，主要应用于芯片外部 SPI FLASH 的访问，内建寄存器访问模式和内存访问模式。提供 6 根信号(clk, cs, io[3:0]) 跟外部 SPI FLASH 相连。支持 SPI 标准操作模式，也支持 SPI dual，quad 的快速操作指令。为实现对 SPI FLASH 的操作，软件在执行指令时，必须指明指令的类型。当前控制器内置有 8 种指令类型，基本可以实现所有的操作。

10.6.6.1. 功能特性

- 支持 SPI 4-wire 标准协议，支持 SPI dual（2bit），quad（4bit）快速模式。
- 提供直接 memory 读取方式，将 SPI flash 存储单元直接映射到芯片内部物理地址空间（最大 16MB），满足 CPU 程序直接在 SPI FLASH 上运行。
- 提供 8 类 cmd_type 供软件选择，执行各条指令。特殊情况下，软件可以使用控制器提供的手动操作模式。
- 支持 word 快速读操作，支持小端模式。

10.6.6.2. 寄存器说明

SPI FLASH 控制器配置物理基地址为 0x1A07_0000。其寄存器地址映射列表和寄存器详细说明如下。

偏移地址	名称	属性	功能描述	默认值
0x4000	SPICTL	RW	SPI 控制寄存器	0x000003C0
0x4100	SPICMD	RW	SPI 指令寄存器	0x00000000
0x4200	SPIADDR	RW	SPI 地址寄存器	0x00000000
0x4300	SPIDATA	RW	SPI 数据寄存器	0x00000000
0x4400	SPISTATUS	R	SPI 状态寄存器	
0x4500	SPIADDROFFSET	RW	SPI 地址偏移寄存器	0x00000000

SPICTL

Bits	Name	Type	Function
31:13	-	-	Reserved.
17:16	CLK_DIV[3:2]	R/W	配合[3:2]确定 SPI 时钟的频率
15:13			Reserved
12	QUAD	R/W	0：四线模式关闭（默认） 1：四线模式开启
11	DUAL	R/W	0：两线模式关闭（默认） 1：两线模式关闭
10:6	CS_MAX	R/W	CS 信号无效（为 1）的宽度，以系统时钟为周期
5			Reserved
4	RISC_ACCESS	R/W	SPI 访问模式： 0：内存直接映射读模式； 1：寄存器操作读写模式；
3:2	CLK_DIV[1:0]	R/W	SPI 时钟频率配置：CLK_DIV[3:0] 分频值=17-clk_div
1	CLK_MODE	R/W	SPI 时钟模式： 固定为 0
0	RESET	R/W	SPI FLASH 控制器复位

SPICMD

Bits	Name	Type	Function
31:11	-	-	Reserved.
10:8	CMD_TYTE	R/W	命令类型
7:0	COMMAND	R/W	命令字

注：任一条指令的执行都是以 command 为起始。当执行下一条指令时，需保证当前的指令已经执行完成。这个可以通过查询 SPISTATUS 寄存器的第 0 位（spi_idle）来判断。当 idle 位至高时，才可以写新的指令到 command 寄存器中。

CMD_TYPE 列表如下：

- 000：单命令
- 001：命令，从设备接收一个数据；
- 010：命令，向设备发送一个数据；
- 011：编程设备；

- 100：Full 快速读取；
- 101：快速读取；
- 110：手工模式读；
- 111：手工模式写；

SPIADDR

Bits	Name	Type	Function
31:24	-	-	Reserved.
23:0	ADDRESS	R/W	SPI 访问地址

SPIDATA

Bits	Name	Type	Function
31:24	-	-	Reserved.
23:0	DATA	R/W	写，发送数据； 读，接收数据；

SPISTATUS

Bits	Name	Type	Function
31:3	-	-	Reserved.
2	READY_FOR_NEXT	R	0：SPI 设备正在被编程； 1：SPI 设备空闲
1	DATA_READY	R	0：数据尚未准备好； 1：数据已准备好，可以读取 SPIDATA；
0	SPI_IDLE	R	0：SPI 接口忙； 1：SPI 接口空闲；

10.6.6.3. 使用介绍

随机读取 SPI FLASH

随机读取 SPI FLASH，SPI 控制器对于 RISC 来说是透明的。你可以使用内存指针直接访问可见的地址，就像访问 SRAM/SDRAM 一样。用 C 语言，就像下面：

```
int *ptr;
ptr = (int *)(0xb900_0000+SPI_FLASH_ADDR);
temp = *ptr;
```

注意：只能是读取操作。

寄存器命令接口读写 SPI 设备

在复杂的应用中，需要访问设备的一些高级特性。这些可以由寄存器命令接口来完成。

当前支持 7 种不同的命令：

000：单命令

001：命令，从设备接收一个数据；

010：命令，向设备发送一个数据；

011：编程设备；

100：Full 快速读取；

101：快速读取；

110：手工模式读；

111：手工模式写；

10.6.7. SPI 通用控制器

SPI CTRL 为通用 SPI 控制器模块，主要应用于芯片与外部 SPI 接口模块的通讯与控制。硬件配置有 CLK，CS，MOSI，MISO 信号线作为外部接口。

10.6.7.1. 功能特性

- 支持 SPI 4-wire 标准协议，支持 DMA 功能。
- 软件使用控制器完成协议传输，灵活性高，兼容通用 SPI 协议。
- 内建独立的 8x32 发送缓存和 8x32 接收缓存。

10.6.7.2. 寄存器说明

SPI 通用控制器配置物理基地址为 0x1A08_0000。其寄存器详细说明如下表。

偏移地址	名称	位宽	功能
0x0080	SPISR	32	SPI 状态寄存器
0x0084	SPICR	32	SPI 控制寄存器
0x0088	SPISSR	32	SPI 设备选择寄存器
0x008c	reserved	N/A	N/A
0x0090	SPITR	32	SPI 发送 FIFO 数据入口
0x0094	SPIRR	32	SPI 接收 FIFO 数据出口
0x0098	SPITR1	32	SPI 发送数据寄存器
0x009c	SPIRR1	32	SPI 接收数据寄存器

位置	名称	属性	定义
31-27	Reserved	N/A	N/A
26	TXBUF_FULL	R	1：发送 fifo 满； 0：发送 fifo 未满
25	TXFIFO_EMPTY	R	1：发送 fifo 空； 0：发送 fifo 不空
24	TXFE	R	1：发送 fifo 半空； 0：发送 fifo 半满
23-19	Reserved	N/A	N/A
18	RXBUF_FULL	R	1：接收 fifo 满； 0：接收 fifo 未满
17	RXFIFO_EMPTY	R	1：接收 fifo 空； 0：接收 fifo 不空
16	RXDA	R	1：接收 fifo 半满； 0：接收 fifo 半空
15-8	Reserved	N/A	N/A
7	DONE	read	Done bit 1: 表示一个 Byte 传输完成； 0：传输正在进行
6	Reserved		
5	BB	read	SPI 忙

			1：SPI 接口忙； 0：SPI 接口空闲；
4	INT_N	Read clear	中断状态，发送寄存器空、接收寄存器满或者 SPI 错误，低电平有效
3	XMIT_EMPTY	read	发送寄存器空
2	RCV_FULL	read	接收寄存器满
1:0	Reserved	N/A	N/A

位置	名称	属性	定义
31	DMA_TX_EN	RW	1：dma 发送使能； 0：dma 发送无效
30	DMA_RX_EN	RW	1：dma 接收使能； 0：dma 接收无效
29	PURE_TX_EN	RW	1：不通过 fifo 的形式发送数据； 0：通过 fifo 的形式发送数据
28	PURE_RX_EN	RW	1：不通过 fifo 的形式接收数据； 0：通过 fifo 的形式接收数据
27	FIFO_FLUSH	W	1：fifo 复位； 0：无效
26-24	SPI_SEL	RW	1：对应的 spi 片选除能； 0：开启对应的 spi 片选
23-21	RESERVED	N/A	N/A
20-15	BITNUM	R/W	每个 word 传输 bitnum 个比特（在 1 和 32 之间）
14	WR_ONLY	R/W	1:进入只写模式，不填充 RCV reg 来清除接收完成标志
13	RD_ONLY	R/W	1:进入只读模式，不填充 xmit reg 来清除发送完成标志
12	SS_N_CLKEN	R/W	Spi clock 输出使能
11	SCLK_EN	R/W	Spi clock 使能
10-9	N/A	N/A	
8	CLKDIV2	R/W	这位与 CLKDIV 一起决定了 spi clock 的时钟周期
7	SPIEN	R/W	1：使能 SPI 控制器 0：禁止 SPI 控制器
6	INTEN	R/W	1: 使能中断 0: 关闭中断
5	START	R/W	SPI 发送开始 当最后一个数据写入到 SPITR，xmit_empty 有效，START 无效表示是最后一个数据
4-3	CLKDIV	R/W	见 CLKDIV2 说明
2	CPHA	R/W	时钟相位： 0：数据在第一个时钟有效； 1：数据在第二个时钟有效；

1	CPOL	R/W	Clock polarity bit 0: spi clock is low when idle 1: spi clock is high when idle
0	RCV_CPOL	R/W	Receive clock polarity 0: receive data is sampled on the faling edge of spi clock 1: receive data is sampled on the rising edge of spi clock

位置	名称	属性	定义
31-0	TXFIFO_FR	W	发送 fifo 数据入口

位置	名称	属性	定义
31-0	RXFIFO_FR	R	接收 fifo 数据接口

位置	名称	属性	定义
31-0	D31-D0	R/W	SPI transmit data

位置	名称	属性	定义
31-0	D31-D0	R/W	SPI recive data

注意：

A、SPI 模块可以通过 pure_tx_en 和 pure_rx_en 选择通过寄存器方式发送和接收数据，或者通过 fifo 的形式发送和接收数据，支持 dma 模式；

B、spi_sel 的分别对应 cs 的三个输出，在应用中需保证 spi_sel 只有一位为低，即选择一个外设；

C、这个 spi 控制器在开始 start 之前，需要向发送 fifo 填充数据，如果是通过 dma 方式发送数据，则需要软件检测到发送 fifo 非空的情况下，才能使能 start 信号。

10.6.7.3. 使用介绍

配置控制寄存器，选择时钟分频，时钟输出使能，发送和接收相位，以及 bitnum，spi_sel，spi 使能，wronly，rxonly 等；

选择是通过寄存器方式发送和接收数据还是 fifo 的形式：如果是寄存器的形式，跳到 C 步骤；如果是 fifo 的形式，跳到 H 步骤；

使能 pure_tx_en 和 pure_rx_en 信号；

填充 spitr1 寄存器；

使能 start 信号；

等待状态寄存器的 done 信号置位，判断 rcv_full 状态，如果有效，读取 spirr1 寄存器，判断整个传输是否完成，如果完成，跳到 L 步骤；否则跳到 G 步骤；

判断 xmit_empty 信号，如果有效，填充 spitr1 寄存器；跳到 f 步骤；

除能 pure_tx_en 和 pure_rx_en 信号；

使能 dma_tx_en 和 dma_rx_en 信号；

检测发送 fifo 是否非空，如果是，使能 start 信号，开始 spi 传输；

等待 dma 对应通道的完成中断，如果完成，跳到 L 步骤；

除能 start 信号；一个 spi 传输完成

10.6.8. TIMER

Timer 为通用定时器模块，主要用于定时控制。

10.6.8.1. 功能特性

- 独立的 4 组定时器，每组都可以触发中断。
- 定时器计数位宽为 32bit。计数时钟与系统时钟一致

10.6.8.2. 寄存器说明

Timer 模块的物理基地址为 0x1A0A_0000。其寄存器地址映射列表和寄存器详细说明如下。

偏移地址	名称	属性	功能描述	默认值
0x0000	COUNTER0	RW	Timer0 计数寄存器	0x00000000
0x0004	COMPARE0	RW	Timer0 比较值寄存器	0xFFFFFFFF
0x0008	CONTROL0	RW	Timer0 控制寄存器	0x00000000
0x000C	CLKSEL0	RW	Timer0 计数时钟选择	0x00
0x0010	COUNTER1	RW	Timer1 计数寄存器	0x00000000
0x0014	COMPARE1	RW	Timer1 比较值寄存器	0xFFFFFFFF
0x0018	CONTROL1	RW	Timer1 控制寄存器	0x00000000
0x001C	CLKSEL1	RW	Timer1 计数时钟选择	0x00
0x0020	COUNTER2	RW	Timer2 计数寄存器	0x00000000
0x0024	COMPARE2	RW	Timer2 比较值寄存器	0xFFFFFFFF
0x0028	CONTROL2	RW	Timer2 控制寄存器	0x00000000
0x002C	CLKSEL2	RW	Timer2 计数时钟选择	0x00
0x0030	COUNTER3	RW	Timer3 计数寄存器	0x00000000
0x0034	COMPARE3	RW	Timer3 比较值寄存器	0xFFFFFFFF
0x0038	CONTROL3	RW	Timer3 控制寄存器	0x00000000
0x003C	CLKSEL3	RW	Timer3 计数时钟选择	0x00

Counter 寄存器：存放 timer 当前值

31	0
当前计数值	

Compare 寄存器：存放 timer 终止值

31	0
计数器终值	

Control 寄存器：控制 timer 开关和 timer 完成

31	1	0
	TO	EN

EN: default 0, 1 开始计数

TO: default 0, 1 表示发生溢出，保持直到该位被写 0

ClkSel 寄存器：计数时钟选择

31	5	0

0：无分频

1：2 分频

2：4 分频

。。。

10.6.8.3. 使用介绍

Timer 定时器使用比较简单，在使用前设置 Control 寄存器的 EN 为 0；配置 Compare 寄存器值为目标计数值；设置 Control 寄存器的 EN 为 1，启动定时器功能；轮询 Control 寄存器的 TO，判断是否定时溢出，也可以利用 PIC 模块上连接的 timer 中断，判断是否定时溢出。

10.6.9. GPIO

10.6.9.1. 功能特性

GPIO 作为数字 IO 口复用的一种功能，能够控制数字 IO 的方向和电平，获取 IO 的状态以及相应外部中断。

每个 GPIO 方向、电平独立可配；

每个 GPIO 的中断支持电平（高、低），沿（上升、下降或者双沿）；

10.6.9.2. 寄存器说明

基地址：

GPIO1：0x1A01_0000

GPIO2：0x1A02_0000

寄存器列表

偏移	名字	描述	类型
0x000	GPIO_DATA	数据寄存器	RW
0x400	GPIO_DIR	方向寄存器	RW
0x404	GPIO_IS	中断类型选择寄存器	RW
0x408	GPIO_IBE	中断双沿选择寄存器	RW
0x40C	GPIO_IEV	中断事件选择寄存器	RW
0x410	GPIO_IE	中断 Mask 寄存器	RW
0x414	GPIO_RIS	Raw 中断状态寄存器	RW
0x418	GPIO_MIS	Mask 中断状态寄存器	RW
0x41C	GPIO_IC	中断清零寄存器	RW

寄存器名称：GPIO_DATA

寄存器偏移：0x000

位域	名字	描述	属性	复位
n-1:0	GPIO_DATA	当 IO 配置为输出时，可读写，每位控制 IO 的输出电平； 当 IO 配置为输入是，只读，每位反应 IO 的输入电平； n 代表 IO 的数目	RW	0

寄存器名称：GPIO_DIR

寄存器偏移：0x400

位域	名字	描述	属性	复位
n-1:0	GPIO_DIR	IO 方向， 0：输入 1：输出 n 代表 IO 的数目	RW	0

寄存器名称：GPIO_IS

寄存器偏移：0x404

位域	名字	描述	属性	复位
n-1:0	GPIO_IS	中断类型选择， 0：沿触发 1：电平触发 n 代表 IO 的数目	RW	0

寄存器名称：GPIO_IBE

寄存器偏移：0x408

位域	名字	描述	属性	复位
n-1:0	GPIO_IBE	中断双沿选择， 0：上升沿或者下降沿触发，由 GPIO_IEV 决定； 1：双沿触发； n 代表 IO 的数目	RW	0

寄存器名称：GPIO_IEV

寄存器偏移：0x40C

位域	名字	描述	属性	复位
n-1:0	GPIO_IEV	中断事件沿选择， 0：下降沿或者低电平触发；	RW	0

		1：上升沿或者高电平触发； n 代表 IO 的数目		

寄存器名称：GPIO_IE

寄存器偏移：0x410

位域	名字	描述	属性	复位
n-1:0	GPIO_IE	中断屏蔽， 0：屏蔽中断； 1：打开中断； n 代表 IO 的数目	RW	0

寄存器名称：GPIO_RIS

寄存器偏移：0x414

位域	名字	描述	属性	复位
n-1:0	GPIO_RIS	Raw 中断状态， 0：无中断； 1：有中断； n 代表 IO 的数目	R	0

寄存器名称：GPIO_MIS

寄存器偏移：0x418

位域	名字	描述	属性	复位
n-1:0	GPIO_MIS	Mask 中断状态， 0：无中断； 1：有中断； n 代表 IO 的数目	R	0

寄存器名称：GPIO_IC

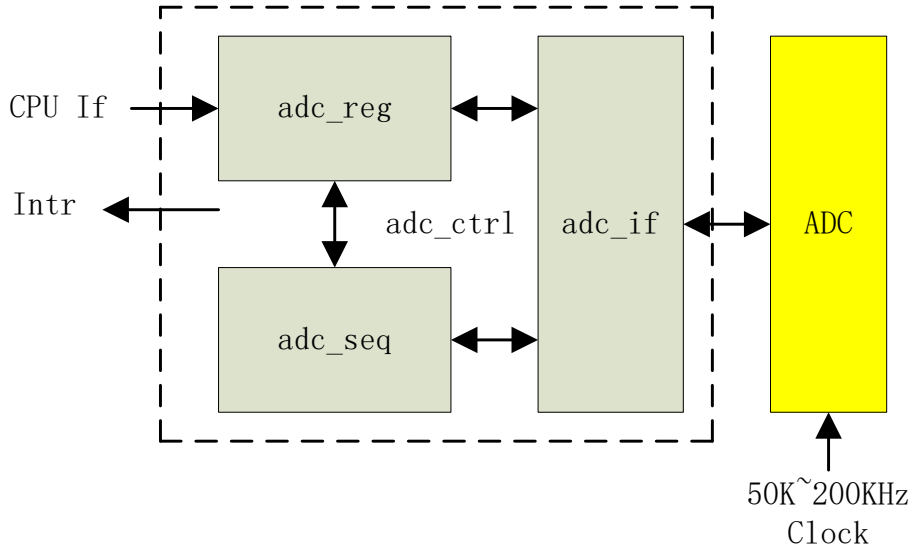
寄存器偏移：0x41C

位域	名字	描述	属性	复位
n-1:0	GPIO_IC	中断清零，写 1 清零，写 0 无效。 n 代表 IO 的数目	R	0

10.6.10. ADC 控制器

10.6.10.1. 模块框图

ADC 控制器是为 ADC 模拟模块 Saradc12b_phan 设计的数字控制部分，主要完成 ADC 的配置，以及单次转换、定时转换等控制。如下图，设计方框图：



ADC 模拟模块采用 3.3V 供电，参考电压为 3.3V、2.4V 可选，输入时钟 50~200KHz，8 通道模拟信号输入，通过 3bit 信号控制选择；同时内部带上拉电阻以及输入模拟信号跳变检测等。

ADC 控制器分为三个模块，其中 adc_reg 完成寄存器配置和单次转换请求的产生；adc_seq 主要完成定时转换请求的产生以及 ADC 采样数据跳变检测；adc_if 主要完成 ADC 转换的控制时序产生以及处理单次、定时的转换请求。

10.6.10.2. 寄存器说明

基地址：0x1A0D_3000

寄存器列表

Offset	Name	Description	Type
0x000	ADC_CFG	ADC 控制器配置	RW
0x004	ADC_STS	ADC 模拟模块状态	RO
0x008	ADC_DATA	ADC 单次转换数据	RO
0x00c	ADC_INT_RSTS	ADC 控制器 RAW 中断	RO
0x010	ADC_INT_MSTS	ADC 控制器 MASK 中断	RO
0x014	ADC_INT_MSK	ADC 控制器中断 MASK	RW
0x018	ADC_INT_CLR	ADC 控制器中断清零	RW
0x100	ADC_SEQ_CFG	ADC 控制器定时转换配置	RW
0x104			
0x108	ADC_SEQ_PERIOD	ADC 控制器定时转换周期配置	RW
0x10c	ADC_SEQ_THR	ADC 控制器定时转换采样值变化阈值	RW
0x110	ADC_SEQ_DATA	ADC 控制器定时转换数据	RO

ADC_CFG Bit Descriptions

Field	Name	Description	Access	Reset
31:7	reserved			
9	ADC_START	ADC 模拟模块启动转换信号		
8	START	ADC 控制器单次转换启动信号，写 1 启动，硬件自动清零		
7	BYPASS	ADC 模拟模块手动控制旁路： 0：ADC 控制器单次转换自动控制，由 START 启动； 1：ADC 控制器直接控制，由 ADC_START 控制		
6:4	CSEL	ADC 模拟模块通道选择（单次转换）		
3:2	RSEL	ADC 模拟模块上下拉控制		
1	AUTO	ADC 模拟模块转换模式： 0：ADC 模拟模块自动转换； 1：ADC 模拟模块转换由控制器控制；		
0	ENABLE	ADC 模拟模块使能		

ADC_STS Bit Descriptions

Field	Name	Description	Access	Reset
31:28	reserved			
27:16	DATA	ADC 模拟模块 din 端口数据		
15:2	reserved			
1	DETECT	ADC 模拟模块 detect 端口状态		
0	FINISH	ADC 模拟模块 finish 端口状态		

ADC_DATA Bit Descriptions

Field	Name	Description	Access	Reset
31:12	reserved			
11:0	DATA	ADC 控制器单次转换数据		

ADC_INT_RSTS Bit Descriptions

Field	Name	Description	Access	Reset
31:4	reserved			

3	SEQ_CHG	ADC 控制器定时转换采样值变化中断		
2	SEQ_FINISH	ADC 控制器定时转换中断		
1	SIN_FINISH	ADC 控制器单次转换中断		
0	DETECT	ADC 模拟模块检测中断		

ADC_INT_MSTS Bit Descriptions

Field	Name	Description	Access	Reset
31:4	reserved			
3	SEQ_CHG	ADC 控制器定时转换采样值变化中断		
2	SEQ_FINISH	ADC 控制器定时转换中断		
1	SIN_FINISH	ADC 控制器单次转换中断		
0	KEYPRESSD	ADC 模拟模块检测中断		

ADC_INT_MSK Bit Descriptions

Field	Name	Description	Access	Reset
31:4	reserved			
3	SEQ_CHG	ADC 控制器定时转换采样值变化中断 MASK		
2	SEQ_FINISH	ADC 控制器定时转换中断 MASK		
1	SIN_FINISH	ADC 控制器单次转换中断 MASK		
0	KEYPRESSD	ADC 模拟模块检测中断 MASK		

ADC_INT_CLR Bit Descriptions

Field	Name	Description	Access	Reset
31:4	reserved			
3	SEQ_CHG	ADC 控制器定时转换采样值变化中断清零		
2	SEQ_FINISH	ADC 控制器定时转换中断清零		
1	SIN_FINISH	ADC 控制器单次转换中断清零		
0	KEYPRESSD	ADC 模拟模块检测中断清零		

ADC_SEQ_CFG Bit Descriptions

Field	Name	Description	Access	Reset
31:5	reserved			
4	CHG	ADC 控制器定时转换采样值变化检测使能		

3:1	CSEL	ADC 控制器定时转换通道选择		
0	ENABLE	ADC 控制器定时转换使能		

ADC_SEQ_PERIOD Bit Descriptions

Field	Name	Description	Access	Reset
31:0	PERIOD	ADC 控制器定时转换周期		

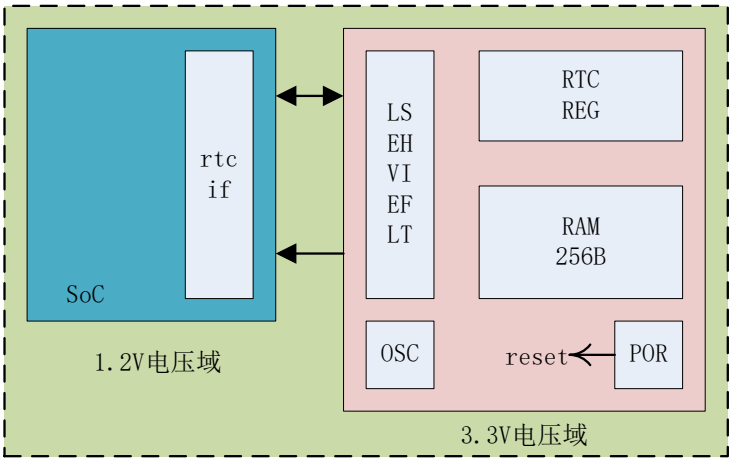
ADC_SEQ_THR Bit Descriptions

Field	Name	Description	Access	Reset
11:0	THR	ADC 控制器定时转换采样变化检测阈值		

10.6.11. RTC 控制器

10.6.11.1. 模块说明

如下图，独立供电 RTC 结构框图：



独立供电 RTC 分为两部分，蓝色部分的 rtc_if 和粉红色部分的 RTC 主模块。模块 rtc_if 只是负责时序转换，以便 RISC 能够访问 RTC 主模块内部的寄存器和内部 RAM；RTC 主模块，工作在 3.3V 电压域，其主要包括：

- LevelShift：负责 1.2V 和 3.3V 电平转换；
- OSC：32.768KHz 晶振振荡器；
- POR：上电复位产生；
- RTC REG：主要的 RTC 相关寄存器；
- RAM：256 字节的 RAM；

RAM 的偏移基准地址：0x400，按 WORD 访问。

10.6.11.2. 功能描述

日历功能：提供时钟/日历功能，编码为 BCD 格式，闰年自动识别，提供闰年标志位 Leap；程序可以直接读写寄存器，能够很方便的完成时钟/日历的设置和读取。

定时器功能：提供一个 8 位定时器，定时器加 1 到预设值时产生溢出脉冲，置位 TF 标志位，如果中断使能位 TIE 为 1 则产生中断信号；TF 只能软件清零。

闹钟功能：提供星期、日、小时、分钟的闹钟功能，当设置某个闹钟使能位为 0 时，则相应的闹钟条件有效。当日历/时

钟数据符合闹钟条件时，则会置位闹钟标志位 AF，如果中断使能位 AIE 为 1 则产生中断信号。闹钟标志位 AF 软件清零。

硬件秒时钟调整功能：集成了数字式时间调整电路，通过内部的高精度时间调整电路，可以有效调整由于晶振频率随环境温度变化而导致的时钟走时偏差；也可以调整由于晶振固有偏差而导致的时钟走时偏差。

可以根据不同的时钟走时精度要求，选择适当的时钟调整周期。时钟调整电路原理如下：时钟调整是在每个时钟调整周期内，根据预先设置的数据改变被调整的秒周期的计数脉冲个数，从而弥补在这个时钟调整周期内时钟走时的偏差，未被调整的秒周期的计数脉冲为 32768/32000 个。例如使用 32.768KHz 晶振，由于晶振温漂而使振荡频率为 32.76775KHz：

如果时钟调整周期选择为 20 秒，RTC 在 20 秒内准确的计数脉冲应该为 655355 个，则被调整的秒周期的计数脉冲个数应设置为：32768+（32767.75-32768）x20=32763（7FFBh）。则在每 20 秒内的某个秒时刻（例如 00、20、40），计数脉冲个数为 32763，这样每 20 秒内的计数脉冲就为：（32768x19）+32763=655355，RTC 的时钟走时就很准确。

如果每 20 秒补偿 1 个时钟脉冲，则时钟走时精度为 1/655355=±1.5ppm。不过这种方法只是对时钟走时进行调整，并不对晶振的振荡频率进行调整。

10.6.11.3. 寄存器说明

基地址：0x1A0D_0000

寄存器均是 8bit 宽度。

表格– RTC 寄存器列表

偏移	名字	描述	类型
0x0100	TIMER	定时器预设值	RW
0x0104	TMCON	定时器控制	RW
0x0108	CLKOUT_CON	时钟输出控制（无效）	RW
0x010C	WEEK_ALARM	星期闹钟	RW
0x0110	DAY_ALARM	日期闹钟，BCD 码格式	RW
0x0114	HOUR_ALARM	小时闹钟，BCD 码格式	RW
0x0118	MIN_ALARM	分钟闹钟，BCD 码格式	RW
0x011C	YEARL	年低字节，BCD 码格式	RW
0x0120	MON	月，BCD 码格式	RW
0x0124	WEEK	星期	RW
0x0128	DAY	日期，BCD 码格式	RW
0x012C	HOUR	小时，BCD 码格式	RW
0x0130	MIN	分钟，BCD 码格式	RW
0x0134	SEC	秒，BCD 码格式	RW
0x0138	YEARH	年高字节，BCD 码格式	RW
0x013C	RTC_CS	RTC 控制和状态	RW

0x0140	RTC_CTRL	RTC 控制	RW
0x0144	SCNT_LOADH	秒计时周期调整寄存器	RW
0x0148	SCNT_LOADL	秒计时周期调整寄存器	RW

寄存器名称：TIMER

寄存器偏移：0x100

位域	名字	描述	属性	复位
7:0	TIMER	定时器预设值	RW	0x0

寄存器名称：TMCON

寄存器偏移：0x104

位域	名字	描述	属性	复位
7	Reserved	Reserved	--	--
6	OSC_CTRL[1]			
5	OSC_CTRL[0]	32KHz 晶振增益控制，1 表示大增益模式	RW	1
4:3	Reserved	Reserved	--	--
2	TE	定时器使能， 0：定时器关闭 1：定时器工作	RW	0
1:0	TD	定时器定时时钟选择信号， 00：4096 01：64 10：1 11：1/60	RW	2'b00

寄存器名称：WEEK_ALARM

寄存器偏移：0x10C

位域	名字	描述	属性	复位
7:4	Reserved	Reserved	--	--
3	WAE	星期闹钟屏蔽， 0：不屏蔽 1：屏蔽	RW	1
2:0	WEEK	星期闹钟寄存器，0~6	RW	3'bxxx

寄存器名称：DAY_ALARM

寄存器偏移：0x110

位域	名字	描述	属性	复位
7	Reserved	Reserved	--	--
6	DAE	日期闹钟屏蔽， 0：不屏蔽 1：屏蔽	RW	1
5:0	DAY	日期闹钟寄存器，01~31BCD 码格式	RW	6'hxx

寄存器名称：HOUR_ALARM

寄存器偏移：0x114

位域	名字	描述	属性	复位
7	Reserved	Reserved	--	--
6	HAE	小时闹钟屏蔽， 0：不屏蔽 1：屏蔽	RW	1
5:0	HOUR	小时闹钟寄存器，00~23BCD 码格式	RW	6'hxx

寄存器名称：MIN_ALARM

寄存器偏移：0x118

位域	名字	描述	属性	复位
7	MAE	分钟闹钟屏蔽， 0：不屏蔽 1：屏蔽	RW	1
6:0	MIN	分钟闹钟寄存器，00~59BCD 码格式	RW	7'hxx

寄存器名称：YEARL

寄存器偏移：0x11C

位域	名字	描述	属性	复位
7:0	YEARL	年寄存器低 8 位，00~99BCD 码格式	RW	8'hxx

寄存器名称：MON

寄存器偏移：0x120

位域	名字	描述	属性	复位
7	LEAP	闰年标志	R	1'bx
6:5	Reserved	--	--	--
4:0	MON	月寄存器，01~12BCD 码格式	RW	5'hxx

寄存器名称：WEEK

寄存器偏移：0x124

位域	名字	描述	属性	复位
7:3	Reserved	--	--	--
2:0	MON	星期寄存器，0~6	RW	3'hx

寄存器名称：DAY

寄存器偏移：0x128

位域	名字	描述	属性	复位
7:6	Reserved	--	--	--
5:0	DAY	日期寄存器，01~31BCD 码格式	RW	6'hxx

寄存器名称：HOUR

寄存器偏移：0x12C

位域	名字	描述	属性	复位
7:6	Reserved	--	--	--
5:0	HOUR	小时寄存器，00~23BCD 码格式	RW	6'hxx

寄存器名称：MIN

寄存器偏移：0x130

位域	名字	描述	属性	复位
7	Reserved	--	--	--
6:0	MIN	分钟寄存器，00~59BCD 码格式	RW	7'hxx

寄存器名称：SEC

寄存器偏移：0x134

位域	名字	描述	属性	复位
7	Reserved	--	--	--
6:0	SEC	秒寄存器，00~59BCD 码格式	RW	7'hxx

寄存器名称：YEARH

寄存器偏移：0x138

位域	名字	描述	属性	复位
7:0	YEARH	年寄存器高 8 位，00~99BCD 码格式	RW	8'hxx

寄存器名称：RTC_CS

寄存器偏移：0x13C

位域	名字	描述	属性	复位
7	PD		RW	1'b0
6	TEST		RW	1'b0
5	OSC_DIS	为 1 时，关闭 32.768KHz 晶振	RW	1'b0
4	TI_TP	定时器中断触发信号选择位。	RW	1'b0

		0：定时器溢出脉冲触发中断，定时器每次溢出都会触发中断 1：TF 信号触发中断，则在软件清零 TF 后才能触发下一次中断		
3	AF	闹钟标志位	RW	1'b0
2	TF	定时器定时溢出标志位	RW	1'b0
1	AIE	闹钟中断使能位	RW	1'b0
0	TIE	定时器中断使能位	RW	1'b0

寄存器名称：RTC_CTRL

寄存器偏移：0x140

位域	名字	描述	属性	复位
7:2	Reserved	--	--	--
1:0	SCTRL	时钟调整周期选择位： 00：10（秒）±3.0（ppm） 01：20（秒）±1.5（ppm） 10：40（秒）±0.75（ppm） 11：60（秒）±0.5（ppm）	RW	2'h0

寄存器名称：SCNT_LOADH

寄存器偏移：0x144

位域	名字	描述	属性	复位
7:0	SCNT_LOADH	秒计时周期调整寄存器	RW	8'h00

寄存器名称：SCNT_LOADL

寄存器偏移：0x148

位域	名字	描述	属性	复位
7:0	SCNT_LOADL	秒计时周期调整寄存器	RW	8'h00

10.6.12. VLSP

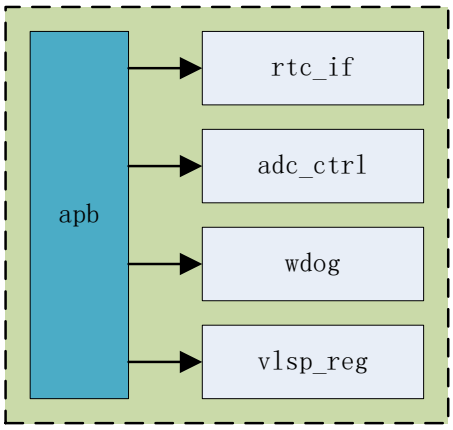
10.6.12.1. 功能特性

VLSP 控制寄存器：提供一个系统软复位的寄存器位。

10.6.12.2. 模块框图

VLSP，Very Low Speed Peripheral，顾名思义，低速外设的意思。

如下图，VLSP 内部框图：



VLSP 模块分为五个部分，蓝色部分的 apb 部分是总线协议转换模块，将 APB 协议转为模块内部的总线；其余四个模块是并行的，通过地址来区分，包括：

- rtc_if：独立供电 RTC 访问接口转换，内部地址 0x0000；
- adc_ctrl：按键 ADC 控制器，内部地址 0x3000；
- wdog：看门狗控制器，内部地址 0x1000；

vlsp_reg：通用的 VLSP 控制寄存器，内部地址 0x2000；

10.6.12.3. 寄存器说明

基地址：0x1A0D_2000

表格–寄存器列表

偏移	名字	描述	类型
0x00	HOT_RST	芯片软复位	RW

寄存器名称：HOT_RST

寄存器偏移：0x00

位域	名字	描述	属性	复位
31:1	Reserved	--	--	--
0	HOT_RST	置 1，复位整个芯片，并自动清零	RW	1'b0

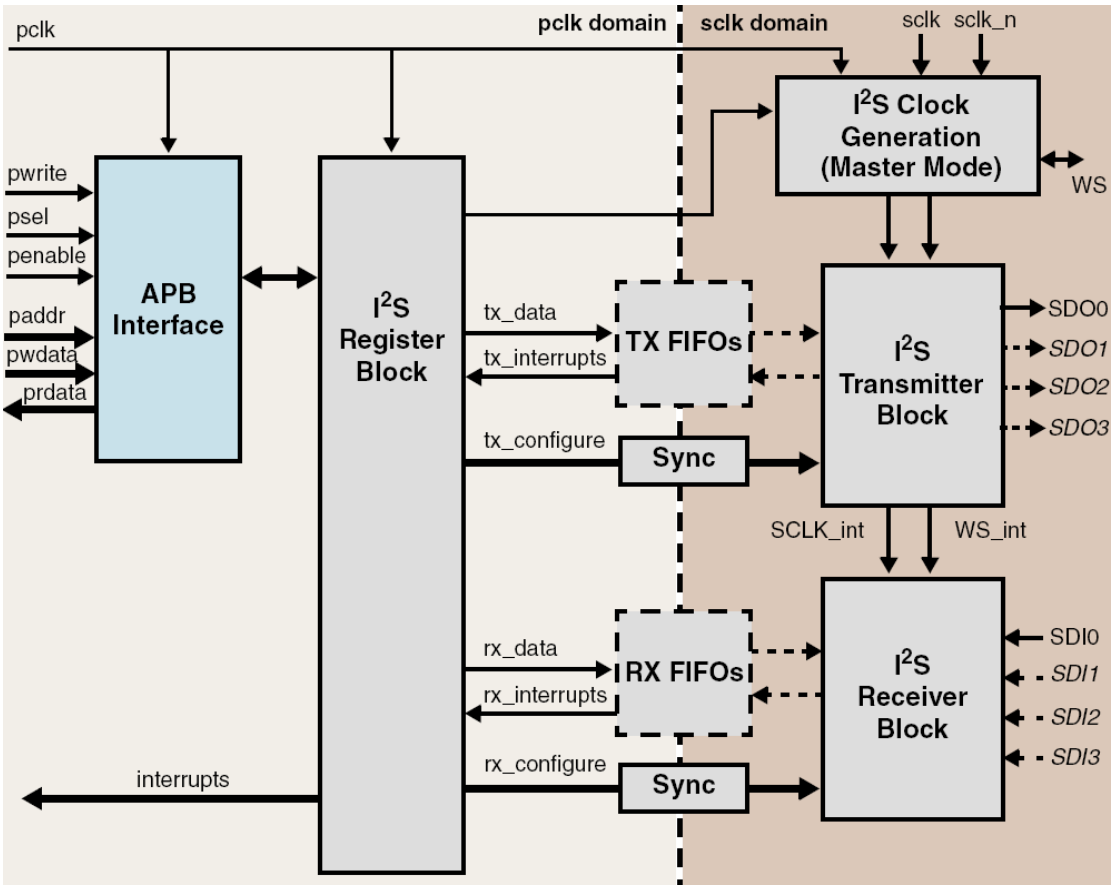
10.6.13. I2S 音频接口

I2S 音频接口，包括 I2S_DAC 和 I2S_ADC，他们使用的是同一设计，只是硬件配置不一样。I2S_DAC 为音频输出，I2S_ADC 为音频输入。

10.6.13.1. 功能特性

- 支持 PDMA 的控制与数据搬移。
- 符合 I2S(Inter-IC Sound)总线标准，支持左对齐模式(Left justified)，右对齐模式(Right justified)以及 DSP/PCM 模式。
- 64x24 发送缓存。

10.6.13.2. 模块框图



其中硬件上只使用了三路音频通道，即 SDO0~SDO2/SDI0~SDI2。

10.6.13.3. 协议时序图

以下三个时序图分别对应为 I²S Data Format、Left-Justified Data Format 和 Right-Justified Data Format。该模块支持 f_{Sample}=32 f_{SCLK} ， f_{Sample}=48 f_{SCLK} 和 f_{Sample}=64 f_{SCLK}。有效数据位可以支持 8、12、16、20 和 24 比特。

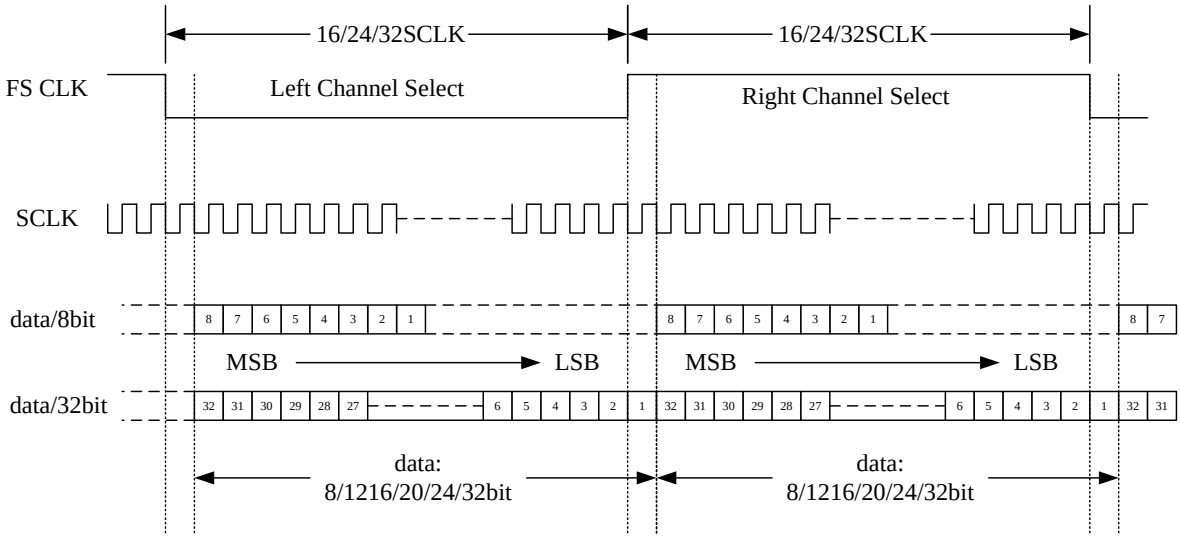


图 I²S Data Format

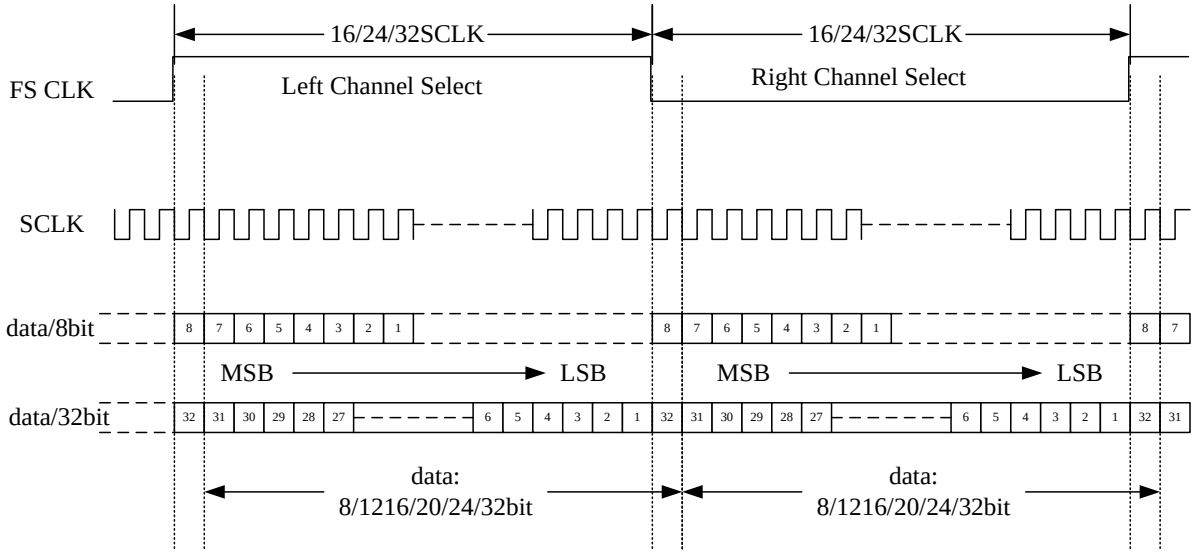


图 Left-Justified Data Format

10.6.13.4. 寄存器说明

I2S_DAC 控制器的配置物理基地址为 0x1E03_0000，I2S_ADC 控制器的配置物理基地址为 0x1E04_0000，其寄存器地址映射列表如下：

地址	寄存器名	宽度	描述	读写属性	复位值
0x000	I2S_CTRL	32	I2S 控制寄存器	RW	0x00000000
0x004	CLK_CTRL	32	I2S 时钟控制寄存器	RW	0x00000000
0x00c	ISRG	32	全局中断寄存器	RW	0x00000000
0x010	RFR0	32	接收通道 FIFO 入口寄存器	RW	0x00000000
0x014	TFR0	32	发送通道 FIFO 入口寄存器	RW	0x00000000
0x018	RCR0	32	接收通道配置寄存器	RW	0x00000000
0x01c	TCR0	32	发送通道配置寄存器	RW	0x00000000
0x020	ISR0	32	通道中断寄存器	R	0x00000000
0x024	IMR0	32	通道中断屏蔽寄存器	RW	0x00000000
0x028	TXFIFO0	32	发送通道 FIFO 状态	R	0x00000000
0x02c	RXFIFO0	32	接收通道 FIFO 状态	R	0x00000000
0x030	RFR1	32	接收通道 FIFO 入口寄存器	RW	0x00000000
0x034	TFR1	32	发送通道 FIFO 入口寄存器	RW	0x00000000
0x038	RCR1	32	接收通道配置寄存器	RW	0x00000000
0x03c	TCR1	32	发送通道配置寄存器	RW	0x00000000
0x040	ISR1	32	通道中断寄存器	R	0x00000000
0x044	IMR1	32	通道中断屏蔽寄存器	RW	0x00000000
0x048	TXFIFO1	32	发送通道 FIFO 状态	R	0x00000000
0x04c	RXFIFO1	32	接收通道 FIFO 状态	R	0x00000000
0x050	RFR2	32	接收通道 FIFO 入口寄存器	RW	0x00000000
0x054	TFR2	32	发送通道 FIFO 入口寄存器	RW	0x00000000

0x058	RCR2	32	接收通道配置寄存器	RW	0x00000000
0x05c	TCR2	32	发送通道配置寄存器	RW	0x00000000
0x060	ISR2	32	通道中断寄存器	R	0x00000000
0x064	IMR2	32	通道中断屏蔽寄存器	RW	0x00000000
0x068	TXFIFO2	32	发送通道 FIFO 状态	R	0x00000000
0x06c	RXFIFO2	32	接收通道 FIFO 状态	R	0x00000000
0x070	RFR3	32	接收通道 FIFO 入口寄存器	RW	0x00000000
0x074	TFR3	32	发送通道 FIFO 入口寄存器	RW	0x00000000
0x078	RCR3	32	接收通道配置寄存器	RW	0x00000000
0x07c	TCR3	32	发送通道配置寄存器	RW	0x00000000
0x080	ISR3	32	通道中断寄存器	R	0x00000000
0x084	IMR3	32	通道中断屏蔽寄存器	RW	0x00000000
0x088	TXFIFO3	32	发送通道 fifo 状态	R	0x00000000
0x08c	RXFIFO3	32	接收通道 fifo 状态	R	0x00000000
0x094	SRESET	32	软件复位寄存器	W	0x00000000

I2S_CTRL 寄存器

地址：0x000

复位值：0x00000000

表格 I2S_CTRL 寄存器描述

区域	描述
31	RX_UNITE_EN：联合传输使能；高电平有效
30:27	RX_UNITE_CHNLSEL：联合输出通道选择；
26	TX_UNITE_EN：联合传输使能；高电平有效
25:22	TX_UNITE_CHNLSEL：联合输出通道选择；
21	保留
20	TXFLUSH：发送 fifo 清空信号，高有效（自复位）；
19:17	保留
16	ITXEN：发送块使能，当该信号被除能时，所有的发送通道被除能；高电平有效
15:13	保留
12	RXFLUSH：接收 fifo 清空信号，高有效（自复位）；
11: 9	保留
8	IRXEN：接收块使能，当该信号被除能时，所有的接收通道被除能；高电平有效
7:1	保留
0	I2S_EN：I2S 使能信号，当该信号除能时，所有传输块和通道被除能，fifo 被清空；高电平有效

CLK_CTRL 寄存器

地址：0x004

复位值：0x00000000

表格 CLK_CTRL 寄存器描述

区域	描述
31:25	保留

24	MODE_EN：主机模式选择；1：主机模式；0：从机模式
23:19	保留
18:16	SCLKGATE：sclk 时钟门控； 0：没有时钟门控 1：在 12 个时钟后门控 2：在 16 个时钟后门控 3：在 20 个时钟后门控 4：在 24 个时钟后门控
15:10	保留
9:8	WSS：ws 周期数选择； 0：16 个周期 1：24 个周期 2：32 个周期 3：保留
7:6	保留
5	保留
4	PCM_MODE：pcm 模式选择 1：选择 pcm 模式 0：选择非 pcm 模式
3:1	保留
0	CLKEN：时钟产生使能信号，高电平有效；

ISRG 寄存器

地址：0x00c

复位值：0x0000_0000

表格 ISRG 寄存器描述

区域	描述
31	保留
30	TXCONFIGERRINTR：配置错误中断，读清除
29	TXFO3：发送 fifo 数据溢出
28	TXFE3：发送 fifo 到达空限值
27	TXCONFIGERRINTR：配置错误中断，读清除
26	CONFIGERRMASK：配置错误中断屏蔽；1：使能中断；0：屏蔽中断；
25	RXFO3：接收 fifo 数据溢出
24	RXDA3：接收 fifo 有效数据超过限值
23:22	保留
21	TXFO2：发送 fifo 数据溢出
20	TXFE2：发送 fifo 到达空限值
19:18	保留
17	RXFO2：接收 fifo 数据溢出
16	RXDA2：接收 fifo 有效数据超过限值
15:14	保留

13	TXFO1：发送 fifo 数据溢出
12	TXFE1：发送 fifo 到达空限值
11:10	保留
9	RXFO1：接收 fifo 数据溢出
8	RXDA1：接收 fifo 有效数据超过限值
7:6	保留
5	TXFO0：发送 fifo 数据溢出
4	TXFE0：发送 fifo 到达空限值
3:2	保留
1	RXFO0：接收 fifo 数据溢出
0	RXDA0：接收 fifo 有效数据超过限值

RFRx 寄存器

地址：0x010, 0x030, 0x050, 0x070

复位值：0x0000_0000

表格 RFRx 寄存器描述

区域	描述
31：WORDSIZE	保留
WORDSIZE-1:0	RFRx：接收通道 x 的 fifo 入口

注：x 为 0,1,2,3

TFRx 寄存器

地址：0x014, 0x034, 0x054, 0x074

复位值：0x0000_0000

表格 TFRx 寄存器描述

区域	描述
31：WORDSIZE	保留
WORDSIZE-1:0	TFRx：发送通道 x 的 fifo 入口

注：x 为 0,1,2,3

RCRx 寄存器

地址：0x018, 0x038, 0x058, 0x074

复位值：0x0000_0000

表格 RCRx 寄存器描述

区域	描述
31	RFF：接收 fifo 清空；高电平有效，自清除
30:29	保留
28:24	RXFTHD：发送 fifo 触发边界，触发值为 RXFTHD+1；
23:22	保留
21	RXMOTOSEL：摩托格式的 pcm 输入；高电平有效
20	RXJUST：数据对齐模式，1：低位先收，0：高位先收；
19:18	保留
17:16	RXMODE：接收模式；

	00：左对齐模式(Left justified) 01：右对齐模式(Right justified) 10：标准 I2S 模式 11：DSP/PCM 模式
15:11	保留
10:8	RWLEN：接收通道字大小； 000：忽略字大小 001：8bit 010：12bit 011：16bit 100：20bit 101：24bit 110：32bit
7:5	保留
4	DMAEN：通道代码使能；高电平有效
3:1	保留
0	RXEN：接收通道启动；高电平有效

TCRx 寄存器

地址：0x01c, 0x03c, 0x05c, 0x07c

复位值：0x0000_0000

表格 TCRx 寄存器描述

区域	描述
31	TFF：传输通道 fifo 清空。高电平有效，自清除
30:29	保留
28:24	TXFTHD：发送 fifo 触发边界，触发值为 TXFTHD+1。
23:22	保留
21	TXMOTOSEL：摩托格式的 pcm 输出；高电平有效
20	TXJUST：数据对齐模式，1：低位先发，0：高位先发。
19:18	保留
17:16	TXMODE：发送模式。 00：左对齐模式(Left justified) 01：右对齐模式(Right justified) 10：标准 I2S 模式 11：DSP/PCM 模式
15:11	保留
10:8	TWLEN：发送通道字大小。 000：忽略字大小 001：8bit 010：12bit 011：16bit 100：20bit

	101：24bit 110：32bit
7:5	保留
4	DMAEN：通道代码使能。高电平有效
3:1	保留
0	TXEN：发送通道启动。高电平有效

ISRx 寄存器

地址：0x020, 0x040, 0x060, 0x080

复位值：0x0000_0000

表格 ISRx 寄存器描述

区域	描述
31:6	保留
5	TXFO：发送 fifo 数据溢出，读清楚
4	TXFE：发送 fifo 到达空限值
3:2	保留
1	RXFO：接收 fifo 数据溢出，读清楚
0	RXDA：接收 fifo 有效数据超过限值

IMRx 寄存器

地址：0x024, 0x044, 0x064, 0x084

复位值：0x0000_0000

表格 IMRx 寄存器描述

区域	描述
31:6	保留
5	TXFOM：发送 fifo 数据溢出，0 为屏蔽
4	TXFEM：发送 fifo 到达空限值，0 为屏蔽
3:2	保留
1	RXFOM：接收 fifo 数据溢出，0 为屏蔽
0	RXDAM：接收 fifo 有效数据超过限值，0 为屏蔽

TXFIFOx 寄存器

地址：0x028, 0x048, 0x068, 0x088

复位值：0x0000_0000

表格 TXFIFOx 寄存器描述

区域	描述
31:16+AW	保留
15+AW:16	RADDR：fifo 读地址，经过同步后
15:AW	保留
AW-1:0	WADDR：fifo 写地址

RXFIFOx 寄存器

地址：0x02c, 0x04c, 0x06c, 0x08c

复位值：0000_0000

表格 RXFIFOx 寄存器描述

区域	描述
31:16+AW	保留
15+AW:16	RADDR：fifo 读地址
15:AW	保留
AW-1:0	WADDR：fifo 写地址，经过同步后

SRESET 寄存器

地址：0x090

复位值：0x0000_0000

表格 sreset 寄存器描述

区域	描述
31:0	SRST：软件复位寄存器，往寄存器写入 0x7255_8841 复位全局寄存器

10.6.13.5. 使用介绍

发送音频数据模式下，CPU 可以通过 LHTR 和 RHTR 寄存器对 FIFO 进行写操作，如 DMA 则通过 TXDMA 寄存器堆 FIFO 进行写操作，当 DMA 操作时，在联合通道情况下，通道 0 和 2 使能时，写入数据的顺序：

Ch0 – Left Data

Ch0 – Right Data

Ch2 – Left Data

Ch2 – Right Data

Ch0 – Left Data

Ch0 – Right Data

……

基本使用流程如下：

a) 使能 I2S 模式，I2S_CTRL[0] = 1；

b) 判断是主机模式还是从机模式，如果是主机模式，配置 CLK_CTRL 的寄存器，使能主机模式和时钟产生模块。

c) 判断是否为联合通道传输模式，如果是联合通道传输模式，配置 I2S_CTRL 寄存器的联合通道选择位，以及配置成发送模式，并使能联合通道功能。

d) 使能发送块，I2S_CTRL[16] = 1；

e) 配置发送模式和发送字长，配置对应的 TCR 寄存器，当为联合通道传输时，只需要配置最小序号对应的 TCR 寄存器即可；例如使能 0，2, 3 的联合通道传输，则只需配置 TCR0 寄存器即可；如果不是联合通过传输，则需要分别配置 TCR0，TCR2，TCR3 寄存器。

f) 配置发送 FIFO 半满触发条件，配置相应的 TCR 寄存器。

g) 使能相应的发送通道，配置相应的 TCR 寄存器使能 DMA 功能和通道。

h) 判断是否为 DMA 传输。

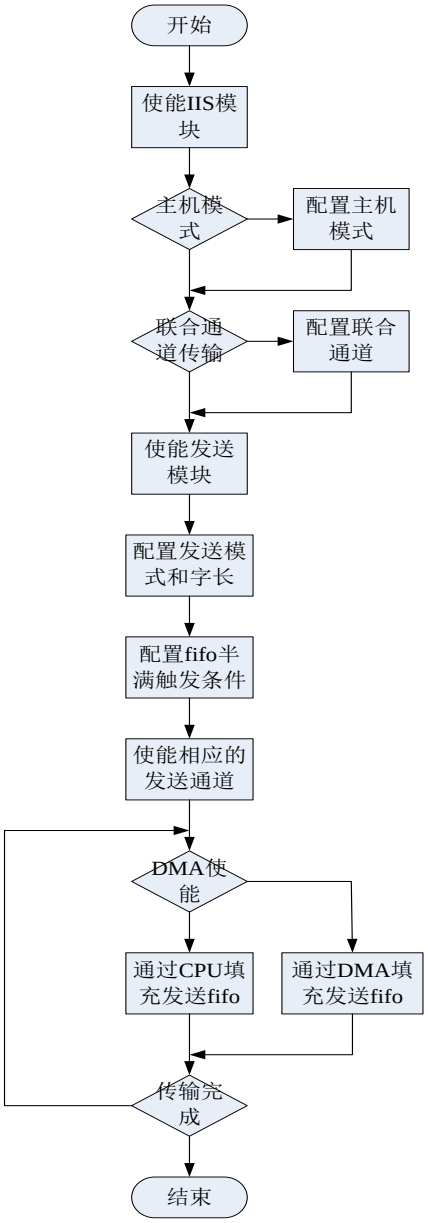
i) 如果是 DMA 传输模式，通过 DMA 填充发送 FIFO。通过相应的 TFR 寄存器填充 FIFO；当为联合通道传输时，只需要填充最小序号对应的 TFR 寄存器即可；例如使能 0，2，3 的联合通道传输，则只需填充 TFR0 寄存器即可；如果不是联合通道传输，则需要分别填充 TFR0，TFR2，TFR3 寄存器，需要配置 3 个 DMA 通道。

j) 如果不是 DMA 传输模式，通过 CPU 填充发送 fifo。通过相应的 TFR 寄存器填充 FIFO；当为联合通道传输时，只需要填充最小序号对应的 TFR 寄存器即可；例如使能 0，2，3 的联合通道传输，则只需填充 TFR0 寄存器即可；如果不是联合通道传输，则需要分别填充 TFR0，TFR2，TFR3 寄存器。

k) 判断传输是否完成，如果没有完成，回到 h，否则，跳到 l。

l) I2S 传输结束，除能 I2S 发送通道和 I2S 发送模块。

流程图如下：



图表 发送功能软件配置流程图

接收音频数据模式下，CPU 可以通过 LRBR 和 RRBR 寄存器读取 FIFO 内的信息，DMA 通道可以通过 RXDMA 寄存器读取，当通过 DMA 读取联合通道情况下，操作流程和发送情况一致。

基本使用流程如下：

a) 使能 I2S 模式，I2S_CTRL[0] = 1；

b) 判断是主机模式还是从机模式，如果是主机模式，配置 CLK_CTRL 的寄存器，使能主机模式和时钟产生模块。

c) 判断是否为联合通道传输模式

d) 如果是联合通道传输模式，配置 I2S_CTRL 寄存器的联合通道选择位，以及配置成接收模式，并使能联合通道功能。

e) 使能接收块，I2S_CTRL[8] = 1；

f) 配置接收模式和接收字长，配置对应的 RCR 寄存器，当为联合通道传输时，只需要配置最小序号对应的 RCR 寄存器即可；例如使能 0，2, 3 的联合通道传输，则只需配置 RCR0 寄存器即可；如果不是联合通过传输，则需要分别配置 RCR0，TCR2，TCR3 寄存器。

g) 配置接收 fifo 半满触发条件，配置相应的 RCR 寄存器。当为联合通道时，同 e 所述。

h) 使能相应的接收通道，配置相应的 RCR 寄存器使能 dma 功能和通道。当为联合通道时，同 e 所述。

i) 判断是否为 dma 传输：

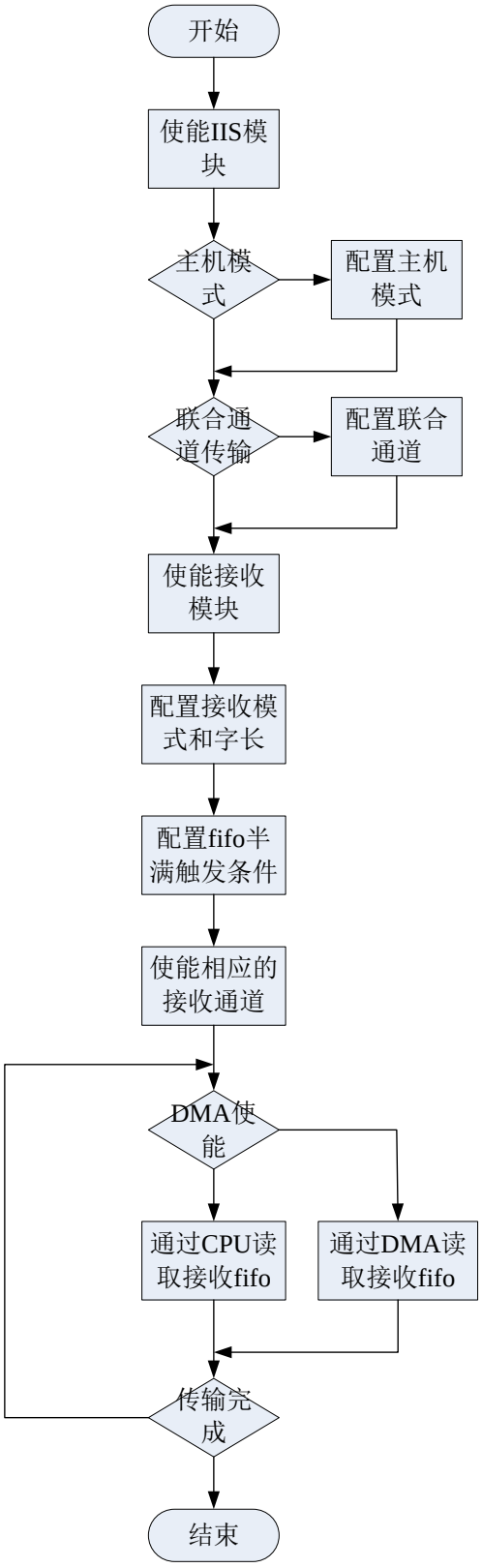
j) 如果是 dma 传输模式，通过 dma 读取接收 fifo。通过相应的 RFR 寄存器读取 fifo 内容；当为联合通道传输时，只需要读取最小序号对应的 RFR 寄存器即可；例如使能 0，2, 3 的联合通道传输，则只需读取 RFR0 寄存器即可；如果不是联合通道传输，则需要分别读取 TFR0，TFR2，TFR3 寄存器，需要配置 3 个 DMA 通道。

k) 如果不是 dma 传输模式，通过 cpu 读取接收 fifo。通过相应的 RFR 寄存器读取 fifo 内容；当为联合通道传输时，只需要填充最小序号对应的 RFR 寄存器即可；例如使能 0，2, 3 的联合通道传输，则只需读取 RFR0 寄存器即可；如果不是联合通道传输，则需要分别读取 RFR0，RFR2，RFR3 寄存器。

l) 判断传输是否完成，如果没有完成，回到 h 步骤，否则，跳到 j 步骤。

m) I2S 传输结束，除能 I2S 接收通道和 I2S 接收模块。

流程图如下：



图表 接收功能软件配置流程图

10.6.14. DMAC

DMAC 为标准 IP 模块。集总式的 DMA 控制器，支持 16 个外部请求（内部硬件通道 5 个）以及任务链表。集成 AHB 总线上，支持存储器与存储器，存储器与外围接口，外围接口与外围接口之间的相互数据传输。主要应用于 SDRAM 数据的搬运以及其他接口设备（I2S，UART，SPDIF）实时数据传输。

10.6.14.1. 功能特性

5 个内部硬件 DMA 通道，通道号分别是 0,1,2,3,6；每个通道支持单方向传输，配备深度为 8WORD 的 FIFO。每个通道有相应的硬件优先级，通道 0 最高。

支持 16 个 DMA 请求，支持 8，16 和 32bit 位宽数据传输。

支持 Single DMA 和 burst DMA 请求信号，DMA burst 的大小可以软件配置。

支持存储器与存储器，存储器与外围接口，外围接口与外围接口之间的相互数据传输。

10.6.14.2. 模块框图

其中：

AHB master interface1 与 SDRAM 直通，直接访问 SDRAM 数据；

AHB master interface2 与外设总线连接，用于对 UART，I2S 等外设的数据读写。

10.6.14.3. 寄存器说明

DMAC 模块物理基地址为 0x1C00_0000。寄存器主要分两类：全局的寄存器和每个通道单独配置的寄存器。

全局的寄存器

名称	位宽	地址	类型	说明
IntStatus	8	0x00	只读	中断状态寄存器，每一位对应相应的一个通道，（以下同）
IntTCStatus	8	0x04	只读	终端计数完成中断请求寄存器，mask 之后
IntTCClear	8	0x08	只写	某位写 1 清除 IntTCStatus 的相应位
IntErrorStatus	8	0x0c	只读	状态错误标志寄存器，mask 之后
IntErrClr	8	0x10	只写	某位写 1 清除 IntErrorStatus 的相应位
RawIntTCStatus	8	0x14	只读	终端计数完成中断请求寄存器，，忽略 mask
RawIntErrorStatus	8	0x18	只读	状态错误标志寄存器，忽略 mask
EnabledChannels	8	0x1c	读写	通道使能
SoftBReq	16	0x20	读写	软件突发数据请求
SoftSReq	16	0x24	读写	软件单个数据请求
SoftLBReq	16	0x28	读写	软件 last 突发数据请求（带 Last 的请求完成时，会产生计数完成中断）
SoftLSReq	16	0x2c	读写	软件 last 单个数据 突发请求
DMACConfiguration	32	0x30	读写	Bit[2]： 0：master 2 little endian 1：master 2 big endian Bit[1]： 0：master 1 little endian 1：master 1 big endian

				Bit[0]： 0：DMAC 关闭 1：DMAC 使能
DMACSync	16	0x34	读写	对应请求是否需要做同步处理，如果为 0，则不做同步处理，1 为做同步处理。

每个通道单独配置的寄存器

DMACCnSrcAddr：寄存器地址：0x100 +n * 0x20 。说明：DMA 取数据的源地址

DMACCnDestAddr：寄存器地址：0x104 +n * 0x20 。说明：DMA 写数据的终端地址

DMACCnLLIAddr：寄存器地址：0x108 +n * 0x20 。说明：下一次链表 DMA 的起始地址，如果为 0，则表示没有后续的链表。如果为非 0 值，则会从该地址起始连续的读取 4 个 WORD 的数据，分别存入到 DMACCnSrcAddr，DMACCnDestAddr，DMACCnLLIAddr 和 DMACCnContro 中，寄存器 DMACCnConfiguration 不可以在链式 DMA 中改变。

DMACCnControl： 寄存器地址：0x10c +n * 0x20。说明：

Bit 位	助记符	说明
31	I	在 LLI 的过程中要不要使能计数完成中断
30：28	Prot	写保护
27	DI	终端地址自增使能
26	SI	源端地址自增使能
25	D	终端 AHB MASTER 选择，0 选择 master1，1 选择 master2
24	S	源端 AHB MASTER 选择，0 选择 master1，1 选择 master2
23：21	DWidth	终端传输的数据位宽 000：字节（8-bit） 001：半字（16-bit） 010：字（32-bit） 011~111：保留
20：18	SWidth	源端传输的数据位宽 同 Dwidth
17：15	DBSize	终端数据突发长度 000：1 001：4 010：8 011：16 100：32 101：64 110：128 111：256
14：12	SBSize	源端数据突发长度 同 DBSize
11：0	TransferSize	数据传输的大小，仅在 DMA 做主控时有效，与源端的数据位宽有关。

DMACCnConfiguration ：寄存器地址：0x110 +n * 0x20。说明：

Bit 位	助记符	说明
22	SBE	源端 burst 使能

20	DBE	终端 burst 使能
18	H	DMA 停止从源端接收数据，但是在 FIFO 中的数据仍然会发送给终端，可以配合 Active 位，来检测 FIFO 是否清空
17	A	检测 FIFO 是否为空，0 表示 FIFO 中已经没有数据（只读）
16	L	为 1 时表示打开锁定传输
15	ITC	终端数据计数完成终端 mask
14	IE	错误状态 mask
13：11	FlowCntl	数据流传输控制，
10	保留	
9：6	DestPeri	终端外设选择，即外设 ID。 0000：UART1 接收； 0001：UART1 发送； 0010：I2S_ADC 接收（SD2）； 0011：UART2 接收； 0100：UART2 发送； 0101：无； 0110：无； 0111：I2S_ADC 接收（SD1）； 1000：无； 1001：I2S_DAC 发送； 1010：无； 1011：I2S_ADC 接收（SD0）； 1100：通用 SPI 控制器发送； 1101：通用 SPI 控制器接收； 1110：SPDIF 发送； 1111：SPDIF 接收；
5	保留	
4：1	SrcPeri	源端外设选择，外设 ID 号，参见 DestPeri 的寄存器说明
0	E	通道使能，启动 DMA

从上面的寄存器来看，全局寄存器设置比较简单。只要把对应通道的使能打开就能工作。而通道寄存器的配置除了地址之外只有两个寄存器需要配置。控制寄存器配置相对比较容易看懂，只需要将对应的需求选中就可以。只有 TransferSize 解释下，它只有在 DMA 做主控的时候会起作用，而且它的数字表面 DMA 需要从源端取得数据的个数，这就和源端的数据位宽有关。它在 LLI 中是可以更改的。

配置寄存器中，SBE 和 DBE 是额外添加的功能，指是否打开 BURST 使能，如果打开，则表示数据向总线的请求尽可能是 BURST 的，最小的 BURST 值为 4，也就是说，如果需要传输的数据能够满足 BURST4 的传输，那么一定会等到满足 BURST4 的条件才会去请求总线，不会跟原来的仅仅是能传输就向总线发请求。造成大量的 BURST1 的总线请求。

FlowCntl 是表明当前的传输是有谁来控制，主要是为了明确当前传输的数据量。

Flowcntl 的值	说明
000	M2M，DMA 主控
001	M2P，DMA 主控
010	P2M，DMA 主控

011	P2P，DMA 主控
100~111	保留

如果源端是 Peri，则需要根据 SrcPeri 的值来选择对应的外设，以便 DMA 控制器正确的选择源端的 REQ。如果终端是 Peri，则需要根据 DestPeri 的值来选择对应的外设，以便 DMA 控制器正确的选择终端的 REQ。

10.6.14.4. 使用介绍

通道之间是有优先级的，通道 0 优先级最高，通道 7 优先级最低。在 M2M 的模式下，由于是 DMA 主控，并且它不需要关注外设的 REQ，所以在该模式下会连续不断的向总线请求数据传输，直到完成 transfersize 大小的数据传输，所以如果是在 M2M 的模式，尽量不要给太高的优先级，以防其他低优先级通道饿死。实际上，通道 6 可以作为 M2M 是较好的选择，它会在每次 BURST 完成之后，插入一个周期的等待，这样如果其他通道有请求则必然会先得到处理。

参考示例：

设置从存储器到外设的传输，外设地址为 0x1E030000，取数地址为 0x10000。

REG(EnabledChannels) = 0x1 ➔ 通道 0 打开

REG(DMACConfiguration) = 0x0 ➔ master1 和 master2 都是小端模式

REG(DMACC0SrcAddr) = 0x10000 ➔ 源端起始地址，注意是实际的物理地址

REG(DMACC0DestAddr) = 0x1E030000 ➔ 终端端起始地址，也是实地址

REG(DMACC0LLIAddr) = 0x200400 ➔ LLI 起始地址,该值不为 0，因此下一次的 DMA 的源地址，终地址，LLI 地址和控制寄存器的设置都从该起始地址开始连续读取获得

REG(DMACCnControl) = 0x85492200➔ 终端地址不自增，即终端是个 FIFO 类型，源端地址自增，终端数据位宽是 32，源端是 32，终端的 BURST 是 8，源端也是 8。数据传输大小为 0x200。

REG(DMACCnConfiguration) = 0x30a41 ➔ SBE 和 DBE 都打开，FlowCntl 选择 M2P 模式，终端外设选择第 10 个，源端为 M 因此不必选择外设，可设为任意值，使能 DMA。此时就会启动 DMA。

LLI 的存储格式：

地址	内容
0x200400	DMACCSrcAddr0
0x200404	DMACCDestAddr0
0x200408	nextLLIAddr0(0x200410)
0x20040c	DMACCControl0
0x200410	DMACCSrcAddr1
0x200414	DMACCDestAddr1
0x200418	nextLLIAddr1(0x0)
0x20041c	DMACCControl1

当 LLIAddr 地址为 0 时，链表结束。

产品名称：	SC6138A	文档类型：	用户手册
版　　权：	杭州士兰微电子股份有限公司	公司主页：	http: //www.silan.com.cn