ESP8266

I2S 模块说明

版本 1.0

乐鑫科技 IOT 团队

bbs.espressif.com

免责申明和版权公告

本文中的信息，包括供参考的 URL 地址，如有变更，恕不另行通知。

文档“按现状”提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档不负任何责任，包括使用本文档内信息产生的侵犯任何专利权行为的责任。

本文档在此未以禁止反言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。

Wi-Fi 联盟成员标志归 Wi-Fi 联盟所有。

文中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

目录

1. 功能综述 1

2. 模块配置 2

2.1. I2S 模块配置 2
2.1.1. I2S 模块复位配置 2
2.1.2. I2S 模块启动配置 2
2.1.3. 发送 / 接收 FIFO 模式配置 3
2.1.4. 声道模式配置 4
2.1.5. 时钟模式配置 4
2.1.6. 其他配置 5
2.2. 链表配置 5
2.3. SLC 模块配置 6
2.3.1. 基本配置 6
2.3.2. 写入首地址 6
2.3.3. 启动 SLC 传输 6

3. 接口函数说明 7

3.1. 空隙函数 7
3.1.1. void i2s_test 7
3.1.2. void i2s_init 7
3.1.3. void creat_one_link 7
3.1.4. void slc_init 7
3.2. 配置函数 7
3.2.1. CONF_RXLINK_ADDR 7
3.2.2. CONF_TXLINK_ADDR 8
3.3. 启动函数 8
3.3.1. START_RXLINK 8
3.3.2. START_TXLINK 8

1. 功能综述

ESP8266 I2S 模块包含一个独立的发送单元和一个独立的接收单元。发送与接收单元各自有一组三线接口：

- 时钟线
- 数据线
- 左右声道选择线

说明：

当数据线写入*"0"*时，时钟和数据输出将停止。

I2S 模块的传输方向见表 1-1。

表1-1: I2S 模块传输方向

	发送	接收
时钟线	输出／输入	输出／输入
数据线	输出	输入
频道选择线	输出	输入

说明：

I2S 模块发送和接收都配有独立的 *FIFO*，其深度为 *128*，宽度为 *32* 比特。两组 *FIFO* 可以通过软件直接访问操作，也可以通过 *SLC* 模块对 *FIFO* 进行自动的 *DMA* 操作。

2. 模块配置

2.1. I2S 模块配置

2.1.1. I2S 模块复位配置

I2SCONF 寄存器中的第 0 位~第 3 位为 I2S 提供软件复位功能，软件需要写入"1"后，写入"0"来完成对应的复位操作，具体为：

- 第 0 位: I2S_TX_RESET
- 第 1 位: I2S_RX_RESET
- 第 2 位: I2S_TX_FIFO_RESET
- 第 3 位: I2S_RX_FIFO_RESET

2.1.2. I2S 模块启动配置

提供运行时钟

要启动 I2S 模块接受数据或发送数据，首先要调用系统函数为 I2S 模块提供运行时钟：

```
i2c_writeReg_ Mask_def (i2c_bbpll, i2c_bbpll_en_audio_clock_out, 1)
```

启动发送模块

I2SCONF 寄存器中的第 8 位用于启动发送模块。

- 如果配置为主机发送模式时，当该位置"1"时，发送模块会一直输出时钟信号、左右声道信号及数据。第一帧数据为 0，随后会移出 FIFO 中的数据。
  - 如果 FIFO 从未写入数据，则数据线会一直保持为 0。
  - 如果 FIFO 将所有写入数据发送完毕，且没有新数据写入时，数据线会循环输出 FIFO 中的最后一个数据。
- 如果配置为从机被动发送模式时，发送模块会等待接收端时钟信号来启动发送模块。

启动接收模块

I2SCONF 寄存器中的第 9 位用于启动接收模块。

- 如果配置为主机接收模式时：
- 当该位置"1"时，接收模块会一直输出时钟信号，并对数据和声道选择线进行采样。
- 该位写"0"，才能停止时钟发送。

- 如果配置为从机被动接收模式时，则准备接收来自于主机的任何发送。

2.1.3. 发送 / 接收 FIFO 模式配置

FIFO 访问模式

I2S_FIFO_CONF 中的第 12 位定义 FIFO 的访问模式。

- 第 12 位置“1”，则 FIFO 由 SLC 模块 DMA 操作，并且对软件直接访问 FIFO 都会是无效操作
- 第 12 位置“0”，则使用软件直接访问 FIFO。
- 第 12 位默认为“1”。

发送 FIFO 模式

I2S_FIFO_CONF 中的第 13~15 位为 i2s_tx_fifo_mod 控制发送数据格式。

值	描述
0	每声道 16 位全数据(双声道，FIFO 存法 16 位左数据，16 位右数据，16 位左数据)
1	每声道 16 位半数据 (单声道，FIFO 存法 16 位数据，16 位无效，16 位数据)
2	每声道 24 位全数据中断 (双声道，FIFO 存法 24 位左数据，8 位无效，24 位右数据，8 位空)
3	每声道 24 位半数据中断(单声道，FIFO 存法 24 位数据，8 位无效，24 位数据，8 位空)
4	每声道 24 位全数据继续(左右声道，FIFO 存法 24 位左数据，24 位右数据)
5	每声道 24 位半数据继续(单声道，FIFO 存法 24 位数据，24 位数据)
6~7	无效

接收 FIFO 模式

I2S_FIFO_CONF 中的第 16~18 位为 i2s_rx_fifo_mod 控制接收数据格式。

值	描述
0	每声道 16 位全数据
1	每声道 16 位半数据
2	每声道 24 位全数据中止
3	每声道 24 位半数据中止
4~7	无效

2.1.4. 声道模式配置

发送声道模式

I2SCONF_CHAN 中的第 0~2 位为发送声道模式 (tx_chan_mod)。

值	描述
0	双声道
1	右声道（左声道和右声道都放右声道的数据）
2	左声道（左声道和右声道都放左声道的数据）
3	右声道（左声道放个常数，从 regfile 来）
4	左声道（右声道放个常数，从 regfile 来）

接收声道模式

I2SCONF_CHAN 中的第 3~4 位为接收声道模式 (rx_chan_mod)。

值	描述
0	双声道
1	右声道
2	左声道

2.1.5. 时钟模式配置

在 *I2SCONF* 中：

- 第 16~21 位设置 I2S 模块输入时钟预分频（I2S_CLKM_DIV_NUM）。

- 第 22~27 位为通信时钟信号分频（I2S_BCK_DIV_NUM）。

2.1.6. 其他配置

寄存器 I2SRXEOF_NUM 设定了 RX FIFO 在触发 SLC 传输时，需要接收的数据个数，以 4 字节作为单位。

参见 DEMO 中的 i2s_reg.h 定义，其余说明将会陆续更新。

2.2. 链表配置

ESP8266 的 SDIO 收发数据包直接会被 DMA 传输到对应的内存。8266 软件中会定义链表注册结构体（或数组）以及一个（或多个）缓存空间。

如图 2-1，本例中只使用一个缓存空间链表也只有一个元素。将缓存的首地址写入链表注册结构体，并完成其他信息，在将链表结构体首地址写入 8266 对应硬件寄存器，DMA 就能自动操作 SDIO 与缓存空间。链表注册结构体具体为：

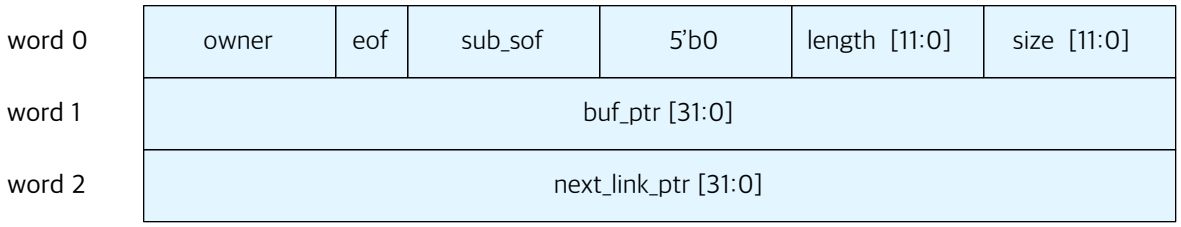

图2-1: 链表注册结构体

名称		描述
owner	1'b0	当前链接对应缓冲的操作者为软件。MAC 不使用该位。
	1'b1	当前链接对应缓冲的操作者为硬件。
eof		帧结束标志（对于 AMPDU 的子帧结束，该标识是不会置上的）。 ▸ 在 MAC 发送帧时用于指示帧结束 link。对于 eof 位置上的链接，它的 buffer_length[11:0] 必须等于该帧剩下的长度，否者 MAC 会上报错误。 ▸ 在 MAC 接收帧时用于指示帧已接收完成。此时该值由硬件置上。
sub_sof		子帧起始链接标识，区分 AMPDU 帧内的不同子帧，仅用于 MAC 发送时。
length[11:0]		缓冲实际占用的大小。
size[11:0]		缓冲的总大小。
buf_ptr[31:0]		缓冲的起始地址。
next_link_ptr[31:0]		下一个描述符的起始地址。在 MAC 接收帧时该值为0，表示已无空缓冲用于接收。

2.3. SLC 模块配置

2.3.1. 基本配置

SLC 模块为 ESP8266 提供多个模块的 DMA 服务。

进行以下设置，表示 SLC 模块用于 I2S 的 FIFO 传输。

- 配置 SLC_CONF0 中的第 12~13 位（SLC_MODE）为“01”。
- 将 SLC_RX_DSCR_CONF 的第 17 位（SLC_INFOR_NO_REPLACE）和第 17 位 SLC_TOKEN_NO_REPLACE）都设置为“1”。

2.3.2. 写入首地址

SLC_RX_LINK 与 SLC_TX_LINK 寄存器的第 0~19 位为收发链表注册结构体的首地址的前 20 位。在启动 SLC 硬件传输前需要将设定好的注册链表首地址写入寄存器。

2.3.3. 启动 SLC 传输

SLC_RX_LINK 与 SLC_TX_LINK 寄存器的第 29 位为启动 SLC 传输控制位，在缓存空间，注册链表并把链表地址前 20 位写入硬件后，将该位置 1 启动 SLC 传输。

3. 接口函数说明

以下函数可在如下地址中找到：

```
/app/driver/i2s.c and /app/include/driver/i2s.h
```

3.1. 空隙函数

3.1.1. void i2s_test

函数	void i2s_test(void)
功能	I2S 模块读写测试程序，DEMO 中的核心函数，用于测试 I2S 的发送接收通信。
参数	无

3.1.2. void i2s_init

函数	void i2s_init(uint8 slc_en)
功能	配置 I2S 相关寄存器。
参数	slc_en: SLC 模块访问使能，0 为软件操作 FIFO，其他数值为 SLC 模块直接访问 FIFO。原理详见 2.1.3 节。

3.1.3. void creat_one_link

函数	void creat_one_link (uint8 own, uint8 eof,uint8 sub_sof, uint16 size, uint16 length, uint32* buf_ptr, uint32* nxt_ptr, struct sdio_queue* i2s_queue)
功能	设置一个链表注册结构体。
参数	struct sdio_queue* i2s_queue: 待配置结构体空间的首地址。 其余参数详见 2.2 节。

3.1.4. void slc_init

函数	void slc_init (uint8 trans_dev)
功能	SLC 模块基础配置。原理详见 2.3 节。
参数	uint8 trans_dev: SLC 模块访问设备，1 为 I2S，0 为 SDIO，其余输入无效。

3.2. 配置函数

3.2.1. CONF_RXLINK_ADDR

函数	CONF_RXLINK_ADDR(addr)
功能	配置接收单元链表结构体地址到寄存器，原理详见 2.3 节。
参数	addr: 链表结构体地址。

3.2.2. CONF_TXLINK_ADDR

函数	CONF_TXLINK_ADDR(addr)
Feature	配置发送单元链表结构体地址到寄存器，原理详见 2.3 节。
Parameter	addr: 链表结构体地址。

3.3. 启动函数

3.3.1. START_RXLINK

函数	START_RXLINK()
功能	启动 SLC 模块接收单元传输,原理详见 2.3 节。
参数	无

3.3.2. START_TXLINK

函数	START_TXLINK()
功能	启动 SLC 模块发送单元传输,原理详见 2.3 节。
参数	无