GR5xx APP驱动用户手册

版本： **1.7**

发布日期： **2025-04-22**

深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司。保留一切权利。

非经本公司书面许可，任何单位和个人不得对本手册内的任何部分擅自摘抄、复制、修改、翻译、传播，或将其全部或部分用于商业用途。

商标声明

GOODiX 和其他汇顶商标均为深圳市汇顶科技股份有限公司的商标。本文档提及的其他所有商标或注册商标，由各自的所有人持有。

免责声明

本文档中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。

深圳市汇顶科技股份有限公司（以下简称“GOODIX”）对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。GOODIX对因这些信息及使用这些信息而引起的后果不承担任何责任。

未经GOODIX书面批准，不得将GOODIX的产品用作生命维持系统中的关键组件。在GOODIX知识产权保护下，不得暗中或以其他方式转让任何许可证。

深圳市汇顶科技股份有限公司

总部地址：深圳市福田区梅康路1号汇顶科技总部大厦26楼

电话：+86-755-33338828　　邮编：518000

网址：www.goodix.com

前言

编写目的

本文详细介绍了APP_DRV（Application Driver）外设模块驱动的架构、API分类、参数结构及常用配置、APP_DRV驱动的使用方法，简要介绍部分模块的示例工程，旨在帮助开发者掌握各外设驱动的使用方法，利用驱动中的API接口实现上层应用程序。

读者对象

本文适用于以下读者：

- 芯片用户
- 开发人员
- 测试人员
- 技术支持工程师

版本说明

本文档为第8次发布，对应的产品系列为GR5526（SDK V1.0.0及之后版本）、GR551x（SDK V2.0.1及之后版本）、GR5x25（SDK V1.0.0及之后版本）、GR533x（SDK V1.0.0及之后版本）和GR5405（SDK V1.1.7及之后版本）。

修订记录

版本	日期	修订内容
1.0	2023-01-10	首次发布
1.1	2023-04-20	• 更新“app_xxx_pin_cfg_t成员”和“APP_DRV配置”章节。 • 删除“APP PDM驱动”下的两个宏定义。
1.2	2023-08-30	• 增加GR5x25适配描述。 • 优化内容描述。
1.3	2023-10-08	增加GR533x适配描述。
1.4	2024-01-08	优化内容描述。
1.5	2024-03-07	增加GR5405适配描述。
1.6	2024-09-03	• 更新TX_ONCE_MAX_SIZE宏定义。 • 增加IO驱动接口函数描述： app_io_set_speed、app_io_set_strength、app_io_set_intput_type。 • 增加app_gpiote_deinit接口函数描述。 • 更新app_comp_pin_t结构体描述。

版权所有 © 2025 深圳市汇顶科技股份有限公司

版本	日期	修订内容
		• 增加PWM驱动接口函数描述： app_pwm_resume、app_pwm_pause、 app_pwm_set_coding_data_in_one_channel、 app_pwm_set_coding_data_in_three_channels、 app_pwm_start_coding_in_one_channel、 app_pwm_start_coding_in_three_channels。 • 增加PWM DMA驱动接口函数描述： app_pwm_dma_init、app_pwm_dma_deinit、app_pwm_start_coding_with_dma。 • 增加APP_BOD_EVENT_LEVEL_XX宏定义，删除APP_BOD_EVENT_LEVEL_1 ～ APP_BOD_EVENT_LEVEL_15宏定义。 • 更新BOD结构体bod_init_t、app_bod_evt_t和枚举app_bod_evt_type_t的描述。 • 增加BOD驱动接口函数描述： app_bod_static_mode_enable、app_bod_event_auto_power_bypass_enable。
1.7	2025-04-22	• 更新“版本介绍”。 • “app_spi_params_t”章节增加SPI Slave相关注意事项。

版权所有 © 2025 深圳市汇顶科技股份有限公司

目录

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

版权所有 © 2025 深圳市汇顶科技股份有限公司

1 概述

1.1 版本介绍

为了降低代码耦合性，并减少固件大小，APP_DRV实现了外设模块DMA传输模式的分离，将传输接口分为轮询方式传输接口（app_xxx_transmit_sync）、中断方式传输接口（app_xxx_transmit_async）及DMA方式传输接口（app_xxx_dma_transmit_async）。

以UART为例，下表显示实现外设模块DMA传输模式的分离，增加了DMA模式设置、传输和控制等相关接口，在进行DMA模式传输时需调用这些接口。

表 1-1 UART接口函数

API		芯片				
		GR551x	**GR5526**	**GR5x25**	**GR533x**	**GR5405**
UART	app_uart_init	Y	Y	Y	Y	Y
	app_uart_deinit	Y	Y	Y	Y	Y
	app_uart_transmit_async	Y	Y	Y	Y	Y
	app_uart_transmit_sync	Y	Y	Y	Y	Y
	app_uart_receive_async	Y	Y	Y	Y	Y
	app_uart_receive_sync	Y	Y	Y	Y	Y
	app_uart_transmit_sem_sync	Y	Y	Y	Y	Y
	app_uart_receive_sem_sync	Y	Y	Y	Y	Y
	app_uart_flush	Y	Y	Y	Y	Y
	app_uart_abort	Y	Y	Y	Y	Y
	app_uart_abort_transmit	Y	Y	Y	Y	Y
	app_uart_abort_receive	Y	Y	Y	Y	Y
	app_uart_get_handle	Y	Y	Y	Y	Y
	app_uart_dma_init	**Y**	**Y**	**Y**	**Y**	**Y**
	app_uart_dma_deinit	**Y**	**Y**	**Y**	**Y**	**Y**
	app_uart_dma_transmit_async	**Y**	**Y**	**Y**	**Y**	**Y**
	app_uart_dma_receive_async	**Y**	**Y**	**Y**	**Y**	**Y**
	app_uart_transmit_dma_sg_llp	N	Y	Y	N	N
	app_uart_receive_dma_sg_llp	N	Y	Y	N	N

说明:

上表中“Y”表示支持；“N”表示不支持。下文表格中同理。

以UART为例，下图展示UART实现DMA传输的流程，在使用DMA方式进行传输，首先需要调用app_uart_dma_init进行DMA初始化，即可使用DMA方式传输接口app_uart_dma_transmit_async进行数据的传输。

版权所有 © 2025 深圳市汇顶科技股份有限公司

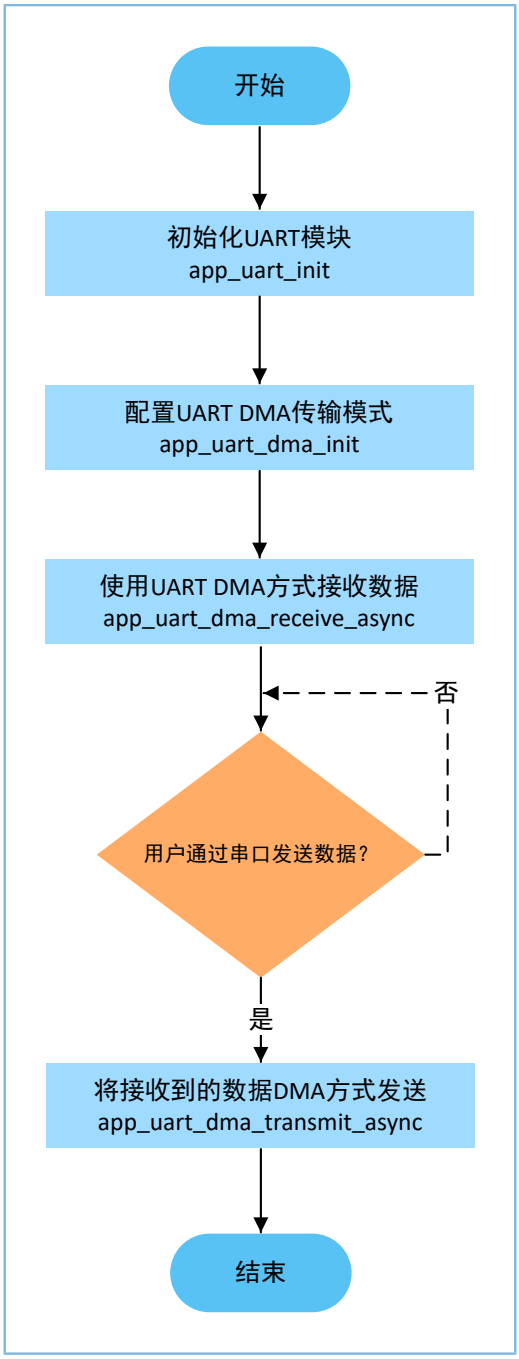

图 1-1 UART使用DMA方式传输

1.2 驱动架构

外设驱动软件架构分为应用层（Application Layer，APP）驱动、硬件抽象层（Hardware Abstraction Layer，HAL）驱动和底层（Low Layer，LL）驱动。

下文简称为APP驱动、HAL驱动和LL驱动，架构图如下所示。

版权所有 © 2025 深圳市汇顶科技股份有限公司

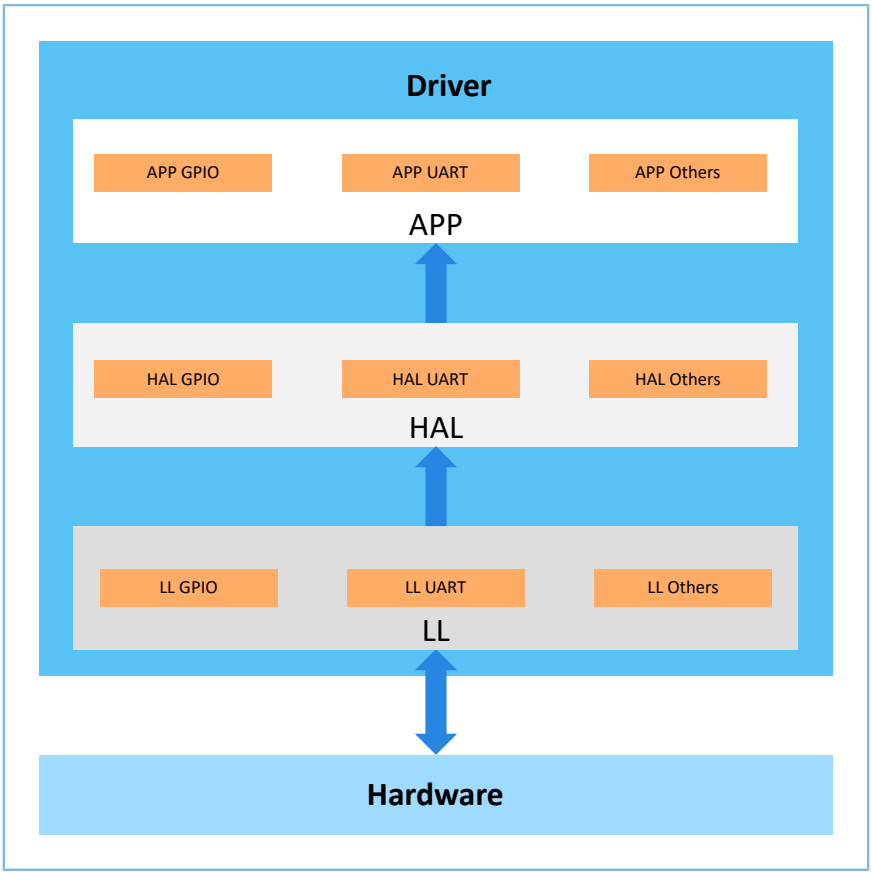

图 1-2 驱动架构

APP、HAL与LL层在软件架构上是相互关联的。APP驱动基于HAL驱动进行封装，接口主要实现外设常用功能，以便应用层调用。HAL层驱动通过调用LL层的内联函数实现对底层硬件的访问。

通常情况下，推荐用户使用APP层接口。

1.2.1 APP驱动

APP驱动基于HAL驱动把各外设常用的功能封装成一系列API接口，除具有HAL驱动的所有特点外，更稳定、安全、易用。开发者可直接使用这些接口编写应用程序而无需关心寄存器配置。

APP驱动主要具有以下特点：

- 各外设基础参数统一定义，开发者只需配置基础参数，无需关心实现细节，可实现整个外设的初始化操作。
- 提供轮询、中断和DMA方式传输接口，使用更灵活。
- 支持自动睡眠管理，在MCU唤醒后无需再次初始化外设。
- 自身维护状态管理，通过各接口返回值可快速确定外设当前状态。
- 初始化可选择传入用户的中断回调函数，当中断事件触发，会调用用户中断回调函数。

APP驱动文件位于SDK包的`SDK_Folder\drivers`目录下。其中`\inc`目录下的头文件如*app_xxx.h*包含了该驱动的结构体申明和函数接口申明；`\src`目录下的源文件如*app_xxx.c*是对驱动函数的实现。

版权所有 © 2025 深圳市汇顶科技股份有限公司

说明:

SDK_Folder为SDK的根目录。

1.2.2 HAL驱动

HAL驱动把各个外设所有功能封装成一系列简单的API接口，开发者利用这些接口可轻松实现底层外设硬件与上层应用程序的交互。

HAL驱动主要具有以下特点：

- 驱动API可兼容同系列的芯片，部分芯片的特殊功能采用扩展API实现。
- 多实例支持，API可在同一外设的多个实例（如I2C1、I2C2）中调用。
- 可在外设初始化/反初始化API中调用用户回调函数，实现时钟、GPIO、中断、DMA的初始化/反初始化。
- 可在外设中断事件或错误中调用回调函数，通知用户相应事件被触发。
- 驱动中的阻塞处理采用超时机制实现，可避免程序死循环。

GR5405芯片的HAL驱动文件位于`SDK_Folder\hal_drv`目录，包括两个子目录：

- include：HAL驱动头文件（*hal_xxx.h*），包含驱动的结构体申明和函数接口申明。
- source：HAL驱动源文件（*hal_xxx.c*），包含驱动函数的实现。

说明:

其他芯片系列的HAL驱动代码未开源，仅提供头文件，位于`SDK_Folder\drivers\inc\hal`目录。

1.2.3 LL驱动

LL驱动通过内联函数的方式封装了各个外设寄存器的原子操作。因此，LL驱动比HAL驱动更接近硬件，可提供全面覆盖外设功能的API函数。开发者可采用LL驱动对HAL驱动无法覆盖到的外设功能进行配置。对于性能要求苛刻的应用场景，开发者可直接使用LL驱动。

LL驱动主要具有以下特点：

- 采用内联函数实现，无函数调用开销。
- 屏蔽了寄存器操作，可移植性高，易于复用。
- 功能覆盖全面。

1.3 API分类

1.3.1 初始化类接口

用于初始化/反初始化外设以及该外设使用的公共资源，包括NVIC中断使能/失能、GPIO引脚上下拉及复用初始化/反初始化、DMA通道初始化/反初始化、睡眠管理函数注册/注销等，如app_adc_init和app_adc_deinit。

版权所有 © 2025 深圳市汇顶科技股份有限公司

1.3.2 功能类接口

用于开启或关闭某项功能，如app_tim_start和app_tim_stop。

1.3.3 传输类接口

传输类接口可分为轮询方式传输接口 、中断方式传输接口和DMA方式传输接口，如app_uart_transmit_sync、app_uart_transmit_async和app_uart_dma_transmit_async。在这种基础传输方式上，个别外设也定制了用于特殊操作的接口。

1.3.4 控制类接口

用于在驱动使用过程中改变参数配置，如app_pwm_config_channel。

1.3.5 获取句柄类接口

用于获取当前外设句柄，如app_i2c_get_handle。外设句柄主要区分不同的外设实例，在特殊情况下可以通过获取外设句柄，调用Hal层接口或者获取Hal层参数。

1.4 初始化参数结构

APP驱动的初始化参数主要包括：

- 外设ID定义app_xxx_id_t，如I2C的ID0、ID1、ID2等。
- 外设角色定义app_xxx_role_t，如I2C的APP_I2C_ROLE_MASTER、APP_I2C_ROLE_SLAVE。
- 外设DMA配置定义app_xxx_dma_cfg_t，如I2C需要配置DMA读写使用的DMA实例和通道。
- 初始化外设GPIO口定义app_xxx_pin_cfg_t，如GPIO上下拉、输入输出、MUX选择。
- 外设数据传输类参数定义xxx_init_t，如SPI的一次传输数据大小、时钟极性、时钟相位、采样频率等。

以I2C为例，初始化参数结构体定义如下：

```
/**
  * @brief I2C parameters structure definition
  */
typedef struct
{
    app_i2c_id_t        id;             /**< specified I2C module ID.      */
    app_i2c_role_t      role;           /**< specified the role of I2C.    */
    app_i2c_pin_cfg_t   pin_cfg;        /**< the pin configuration information for
                                             the specified I2C module.     */
    app_i2c_dma_cfg_t   dma_cfg;        /**< I2C operate mode.             */
    i2c_init_t          init;           /**< I2C communication parameters. */
    i2c_env_t           i2c_dev;
} app_i2c_params_t;
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

1.4.1 app_peripheral_id_t成员

ID成员用于外设拥有多个实例的情况，例如GR5526的I2C外设和UART外设均有6个实例，这里的ID设置为0～5的任意数字，表明使用对应数字的外设实例。当某些外设如DSPI、OSPI、GPU、DC等只有一个实例时，参数结构体则无此成员变量。

1.4.2 app_xxx_pin_cfg_t成员

GPIO引脚设置成员用于需要对IO引脚进行操作的外设，例如UART、SPI、I2C等。该成员由以下四个子成员组成：

- 引脚类型，如普通GPIOA/GPIOB/GPIOC/NORMAL，不掉电AON_GPIO和模拟数字混合MSIOA。仅GR551x支持NORMAL类型，可以被GPIOA/GPIOB替换使用，且GR551x也推荐使用GPIOA/GPIOB。
- 引脚映射，通过配置app_io_mux_t，对照Datasheet中的Pin Mux表，可以将IO配置为对应功能的引脚，以GR5526为例，选择GPIOA4、MUX3可以将GPIOA4配置成UART3的发送功能引脚。
- 引脚编号，除NORMAL类型，该编号可选0～15任意一个数字。NORMAL类型的引脚编号为0～31。
- 引脚模式，可以配置引脚的内部上拉、下拉和浮空。

对于不需要配置GPIO的外设，如TIMER、DUAL_TIMER等，则无需定义该成员变量。

以UART为例，app_uart_pin_cfg_t成员定义如下：

```
typedef struct
{
    app_io_type_t      type;           /**< Specifies the type of UART IO. */
    app_io_mux_t       mux;            /**< Specifies the Peripheral to be
                                            connected to the selected pins. */
    uint32_t           pin;            /**< Specifies the IO pins to be configured.
                                            This parameter can be any value of
                                            @ref GR55xx_pins. */
    app_io_pull_t      pull;           /**< Specifies the Pull-up or Pull-Down
                                            activation for the selected pins. */

}app_uart_pin_t;

/**
  * @brief UART pins config Structures
  */
typedef struct
{
    app_uart_pin_t    tx;              /**< Set the configuration of UART TX pin.  */
    app_uart_pin_t    rx;              /**< Set the configuration of UART RX pin.  */
    app_uart_pin_t    cts;             /**< Set the configuration of UART CTS pin. */
    app_uart_pin_t    rts;             /**< Set the configuration of UART RTS pin. */
}app_uart_pin_cfg_t;
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

1.4.3 xxx_init_t成员

外设参数配置成员用于配置每个外设特有的参数，以PWM为例，pwm_init_t成员定义如下：

```
typedef struct
{
    uint32_t mode;                      /**< Specifies the PWM output mode state.
                                             This parameter can be a value of
                                             @ref PWM_Mode */
    uint32_t align;                     /**< Specifies the PWM alignment mode with
                                             three channels
                                             This parameter can be a value
                                             of @ref PWM_Alignment_Mode */
    uint32_t freq;                      /**< Specifies the PWM frequency.
                                             This parameter must be a number between
                                             0 ~ SystemFreq/2 (max = 32Mhz).*/
    uint32_t bperiod;                   /**< Specifies the PWM breath period in breath
 mode.
                                             Unit: ms.
                                             This parameter must be a number between
                                             0 ~ 0xFFFFFFFF/SystemFreq*1000. */
    uint32_t hperiod;                   /**< Specifies the PWM hold period in breath mode.
                                             Unit: ms.
                                             This parameter must be a number between
                                             0 ~ 0xFFFFFF/SystemFreq*1000. */
    uint32_t bstoplvl;                   /**< Specifies the PWM io level when stop.
                                             This parameter can be a value
                                             of @ref PWM_STOP_LVL */
    pwm_channel_init_t channel_a;       /**< Specifies the configuration parameters
                                             of channel A. */
    pwm_channel_init_t channel_b;       /**< Specifies the configuration parameters
                                             of channel B. */
    pwm_channel_init_t channel_c;       /**< Specifies the configuration parameters
                                             of channel C. */

} pwm_init_t;
```

1.5 APP_DRV配置

用户可通过修改*app_drv_config.h*中的宏定义配置，设置芯片类型和各外设睡眠唤醒优先级等。具体的宏定义描述，参见下表。

表 1-2 *app_drv_config.h*中的宏定义

宏定义	值	描述
APP_DRIVER_GR551X	0x0	GR551x系列芯片
APP_DRIVER_GR5525X	0x1	GR5x25系列芯片
APP_DRIVER_GR5526X	0x2	GR5526系列芯片

版权所有 © 2025 深圳市汇顶科技股份有限公司

宏定义	值	描述
APP_DRIVER_GR5332X	0x3	GR533x系列芯片
APP_DRIVER_GR5405	0x3	GR5405系列芯片
APP_DRIVER_ADC_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	ADC睡眠唤醒优先级
APP_DRIVER_AES_WAKEUP_PRIORITY	WAKEUP_PRIORITY_MID	AES睡眠唤醒优先级
APP_DRIVER_COMP_WAKEUP_PRIORITY	WAKEUP_PRIORITY_LOW	COMP睡眠唤醒优先级
APP_DRIVER_DUAL_TIM_WAKEUP_PRIORITY	WAKEUP_PRIORITY_MID	DUAL_TIM睡眠唤醒优先级
APP_DRIVER_DMA_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	DMA睡眠唤醒优先级
APP_DRIVER_UART_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	UART睡眠唤醒优先级
APP_DRIVER_HMAC_WAKEUP_PRIORITY	WAKEUP_PRIORITY_MID	HMAC睡眠唤醒优先级
APP_DRIVER_I2C_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	I2C睡眠唤醒优先级
APP_DRIVER_I2S_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	I2S睡眠唤醒优先级
APP_DRIVER_QSPI_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	QSPI睡眠唤醒优先级
APP_DRIVER_RNG_WAKEUP_PRIORITY	WAKEUP_PRIORITY_MID	RNG睡眠唤醒优先级
APP_DRIVER_SPI_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	SPI睡眠唤醒优先级
APP_DRIVER_TIM_WAKEUP_PRIORITY	WAKEUP_PRIORITY_MID	TIM睡眠唤醒优先级
APP_DRIVER_PWM_WAKEUP_PRIORITY	WAKEUP_PRIORITY_MID	PWM睡眠唤醒优先级
APP_DRIVER_ISO7816_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	ISO7816睡眠唤醒优先级
APP_DRIVER_PKC_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	PKC睡眠唤醒优先级
APP_DRIVER_DSPI_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	DSPI睡眠唤醒优先级
APP_DRIVER_PDM_WAKEUP_PRIORITY	WAKEUP_PRIORITY_HIGH	PDM睡眠唤醒优先级
SOC_GPIO_PINS_MAX	根据芯片引脚资源定义	普通GPIO Pin的数量
SOC_AON_PINS_MAX	根据芯片引脚资源定义	不掉电AON_GPIO Pin的数量

1.5.1 芯片类型配置

目前APP驱动支持以下芯片类型，宏定义如下：

表 1-3 芯片类型

宏定义	值	描述
APP_DRIVER_GR551X	0x0	GR551x系列芯片
APP_DRIVER_GR5525X	0x1	GR5x25系列芯片
APP_DRIVER_GR5526X	0x2	GR5526系列芯片
APP_DRIVER_GR5332X	0x3	GR533x系列芯片
APP_DRIVER_GR5405	0x3	GR5405系列芯片

用户可根据实际使用的芯片选择芯片类型，例如SOC_GR5515表示GR551x系列芯片，该宏定义在*custom_config.h*文件中。

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 1-4 芯片类型选择

芯片	芯片类型
SOC_GR5515	APP_DRIVER_GR551X
SOC_GR5x25	APP_DRIVER_GR5525X
SOC_GR5526	APP_DRIVER_GR5526X
SOC_GR533x	APP_DRIVER_GR5332X
SOC_GR5405	APP_DRIVER_GR5405

1.5.2 睡眠唤醒优先级配置

系统从睡眠中唤醒时，会按照高优先级到低优先级的顺序来唤醒各个外设。

1.5.2.1 wakeup_priority_t

外设睡眠唤醒优先级的枚举定义如下：

表 1-5 wakeup_priority_t枚举成员

成员	描述
WAKEUP_PRIORITY_LOW = 1	唤醒优先级低
WAKEUP_PRIORITY_MID = 2	唤醒优先级中
WAKEUP_PRIORITY_HIGH = 3	唤醒优先级高

用户可根据实际需求，配置外设模块的睡眠唤醒优先级。下面以ADC模块为例，配置睡眠唤醒优先级：

```
#ifndef APP_DRIVER_ADC_WAKEUP_PRIORITY
#define APP_DRIVER_ADC_WAKEUP_PRIORITY              WAKEUP_PRIORITY_HIGH  /**< ADC Wakeup
                                                                          priority High */
#endif
```

1.6 用户程序文件

表 1-6 用户程序文件

名称	描述
Startup.S	芯片的启动文件
gr_interrupt.c	定义一般的中断处理函数，开发者无需关心
gr_system.c	系统初始化实现
gr_dwt.c	dwt函数接口实现（仅GR5405系列芯片）
gr_mpu.c	mpu函数接口实现（仅GR5405系列芯片）
gr_nvic.c	nvic函数接口实现（仅GR5405系列芯片）
gr_platform.c	平台初始化，包括与平台相关的中断向量表重映射、SDK初始化、SoC初始化、冷热启动处理流程入口等

版权所有 © 2025 深圳市汇顶科技股份有限公司

名称	描述
gr_soc.c	芯片相关的一些初始化流程实现，如Flash、NVDS初始化、晶振配置、PMU校准等
board_SK.h	芯片基本资源初始化，包含串口、LED和Key等；定义了一些常用的GPIO引脚
custom_config.h	用户参数配置，包括RAM大小、代码加载地址、代码运行地址、系统时钟等
main.c	主函数

版权所有 © 2025 深圳市汇顶科技股份有限公司

2 APP驱动接口

本节将对APP驱动中各个外设模块对外使用的接口进行详细介绍。

2.1 睡眠管理

2.1.1 简介

Bluetooth LE SoC支持Active、Idle、Sleep三种功耗模式，在无活跃业务时会进入深度睡眠，将关闭除Always On域、SRAM外所有片上外设的电源，并切换到低速时钟域运行。每个外设模块的状态分为RESET（设备未初始化）、READY（设备就绪）、BUSY（设备忙），当配置为Deep Sleep时，系统会判断每个外设是否处于RESET或者READY状态，只有每个外设都不处于BUSY状态，即外设无活跃业务，系统才会进入深度睡眠模式。当触发系统唤醒后，会执行一次复位启动过程并恢复到睡眠前的上下文。

为保持外设在睡眠唤醒后和进入睡眠前的状态一致，在执行Deep Sleep流程之前，需要备份正在使用的外设模块的寄存器。在APP驱动中有一套备份和恢复的程序框架，实现了外设睡眠恢复的自动管理，用户使用APP驱动时无需自行备份恢复这些寄存器，显著降低了使用复杂度。

软件现有两种实现方式：

- 一种是备份和恢复函数在HAL层实现，且外设状态通过g_devices_state的bit位进行标识。APP驱动接口只实现备份和恢复函数的注册和调用，如pwr_register_sleep_cb、pwr_enter_sleep_check_new和pwr_wake_up_ind_new。该方式适用于GR5526、GR5x25、GR533x。
- 另一种备份和恢复函数在各个外设驱动进行实现，例如xxx_prepare_for_sleep和xxx_wake_up_ind，且外设状态通过外设句柄状态位标识，使用hal_xxx_get_state函数可获取外设状态。各模块在初始化时会调用pwr_register_sleep_cb函数，注册各个模块的备份恢复相关函数。该方式适用于GR551x、GR5405。

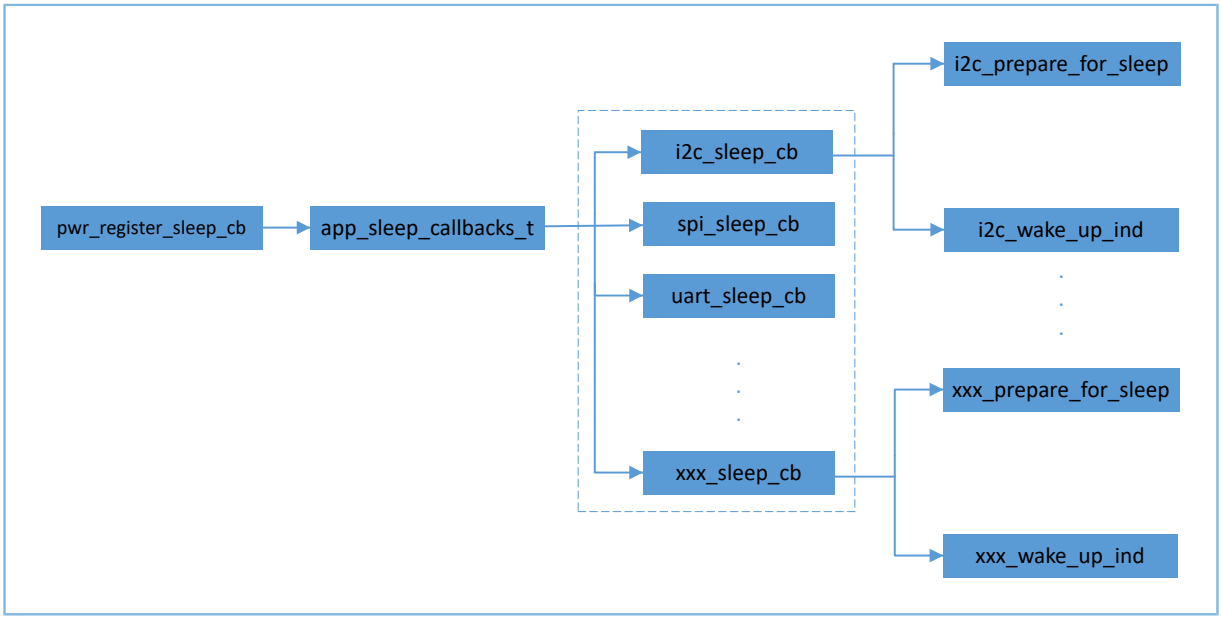

图 2-1 睡眠函数注册流程

版权所有 © 2025 深圳市汇顶科技股份有限公司

系统进入睡眠前，会调用pwr_enter_sleep_check，检查各个模块是否可以进入睡眠，可以睡眠则调用xxx_prepare_for_sleep备份正在使用的外设模块；系统唤醒后，调用pwr_wake_up_ind，从而调用各个模块的xxx_wake_up_ind，将备份的外设寄存器还原。

2.1.2 结构体

2.1.2.1 pwr_env_t

结构体定义如下：

表 2-1 pwr_env_t结构体成员

成员	描述	取值
app_sleep_callbacks_t *pwr_sleep_cb[APP_SLEEP_CB_MAX]	睡眠相关回调函数的数组	保存各模块睡眠相关回调函数
wakeup_priority_t wakeup_priority[APP_SLEEP_CB_MAX]	睡眠唤醒优先级数组	保存各个模块的睡眠唤醒优先级

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwr_mgmt.c。

2.1.2.2 app_sleep_callbacks_t

结构体定义如下：

表 2-2 app_sleep_callbacks_t结构体成员

成员	描述	取值
bool (*app_prepare_for_sleep)(void)	睡眠准备函数，该函数能够保存现场并配置相应模块相关寄存器，使得具体模块进入低功耗模式	注册时进行赋值
void (*app_wake_up_ind)(void)	唤醒后，恢复模块相关寄存器，激活相关模块	注册时进行赋值

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwr_mgmt.c。

2.1.3 枚举

2.1.3.1 pwr_mgmt_dev_state_t

枚举定义如下：

表 2-3 pwr_mgmt_dev_state_t枚举成员

成员	描述
DEVICE_BUSY	外设正忙，不可以进入深度睡眠模式
DEVICE_IDLE	外设空闲，可以进入深度睡眠模式

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.1.4 接口函数

2.1.4.1 pwr_register_sleep_cb

表 2-4 pwr_register_sleep_cb接口

函数原型	pwr_id_t pwr_register_sleep_cb(const app_sleep_callbacks_t *p_cb, wakeup_priority_t wakeup_priority, pwr_id_t id)
功能说明	外设注册睡眠管理
输入参数	• p_cb：回调函数结构体 • wakeup_priority：唤醒优先级 • id：需要注册睡眠管理的外设
返回值	pwr_id_t：模块ID
备注	

2.1.4.2 pwr_unregister_sleep_cb

表 2-5 pwr_unregister_sleep_cb接口

函数原型	void pwr_unregister_sleep_cb(pwr_id_t id)
功能说明	外设注销睡眠管理
输入参数	id：需要注销的模块ID
返回值	
备注	

2.1.4.3 pwr_wake_up_ind

表 2-6 pwr_wake_up_ind接口

函数原型	void pwr_wake_up_ind(void)
功能说明	唤醒时统一恢复外设
输入参数	
返回值	
备注	仅适用于GR551x、GR5405

2.1.4.4 pwr_wake_up_ind_new

表 2-7 pwr_wake_up_ind_new接口

函数原型	void pwr_wake_up_ind_new(void)
功能说明	唤醒时统一恢复外设
输入参数	
返回值	

版权所有 © 2025 深圳市汇顶科技股份有限公司

备注	适用于GR5526、GR5x25、GR533x

2.1.4.5 pwr_enter_sleep_check

表 2-8 pwr_enter_sleep_check接口

函数原型	pwr_mgmt_dev_state_t pwr_enter_sleep_check(void)
功能说明	进入睡眠前检查外设状态
输入参数	
返回值	系统工作状态
备注	仅适用于GR551x、GR5405

2.1.4.6 pwr_enter_sleep_check_new

表 2-9 **pwr_enter_sleep_check_new**接口

函数原型	pwr_mgmt_dev_state_t pwr_enter_sleep_check_new(void)
功能说明	进入睡眠前检查外设状态并备份外设寄存器
输入参数	
返回值	系统工作状态
备注	适用于GR5526、GR5x25、GR533x

2.1.5 接口使用方法

下文以I2C为例介绍GR551x/GR5405接口使用，GR5526、GR5x25、GR533x备份和恢复在HAL层实现，用户无需关注。

1. I2C模块初始化时，注册睡眠回调函数。

```
uint16_t app_i2c_init(app_i2c_params_t *p_params, app_i2c_evt_handler_t evt_handler)
{
    ...
   /*注册睡眠回调函数*/
    pwr_register_sleep_cb(&i2c_sleep_cb, APP_DRIVER_I2C_WAKEUP_PRIORITY, I2C_PWR_ID);
    ...
}
```

2. 实现睡眠回调函数。

```
static const app_sleep_callbacks_t i2c_sleep_cb =
{
    .app_prepare_for_sleep = i2c_prepare_for_sleep, /*进入睡眠，保存现场
                                    并配置i2c相关寄存器，进入低功耗模式*/

    .app_wake_up_ind       = i2c_wake_up_ind   /*唤醒后，恢复i2c相关寄存器，激活
                                                 i2c模块 */
};
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.2 APP IO驱动

2.2.1 结构体

2.2.1.1 app_io_init_t

结构体定义如下：

表 2-10 app_io_init_t结构体成员

成员	描述	取值
uint32_t pin	选择IO引脚	该参数的取值可以是下列值中的任意一个： APP_IO_PIN_0（PIN0）～ APP_IO_PIN_15（PIN15）
app_io_mode_t mode	IO模式配置	该参数的取值可以是下列值中的任意一个： • APP_IO_MODE_NONE（不配置） • APP_IO_MODE_INPUT（输入模式） • APP_IO_MODE_OUTPUT（输出模式） • APP_IO_MODE_MUX（外设功能复用模式） • APP_IO_MODE_IT_RISING（中断上升沿触发模式） • APP_IO_MODE_IT_FALLING（中断下降沿触发模式） • APP_IO_MODE_IT_BOTH_EDGE（中断上升下降沿触发模式） • APP_IO_MODE_IT_HIGH（中断高电平触发模式） • APP_IO_MODE_IT_LOW（中断低电平触发模式） • APP_IO_MODE_ANALOG（模拟输入模式） • APP_IO_MODE_MAX（参数检测）
app_io_pull_t pull	设置IO上拉/下拉/浮空	该参数的取值可以是下列值中的任意一个： • APP_IO_NOPULL（浮空） • APP_IO_PULLUP（上拉） • APP_IO_PULLDOWN（下拉） • APP_IO_PULL_MAX（参数检测）
app_io_mux_t mux	设置IO映射功能	该参数的取值可以是下列值中的任意一个： • APP_IO_MUX_0（MUX模式0） • APP_IO_MUX_1（MUX模式1） • APP_IO_MUX_2（MUX模式2） • APP_IO_MUX_3（MUX模式3） • APP_IO_MUX_4（MUX模式4） • APP_IO_MUX_5（MUX模式5） • APP_IO_MUX_6（MUX模式6） • APP_IO_MUX_7（MUX模式7） • APP_IO_MUX_MAX（MUX模式总数，用作参数检测）

详细信息请参考文件：`SDK_Folder\drivers\inc\app_io.h`。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.2.1.2 app_io_evt_t

结构体定义如下：

表 2-11 app_io_evt_t结构体成员

成员	描述	取值
app_io_type_t type	设置IO类型（GPIOA/NORMAL/GPIOB/GPIOC/AON_GPIO/MSIO）	该参数的取值可以是下列值中的任意一个： • APP_IO_TYPE_GPIOA（GPIO0 ～ GPIO15） • APP_IO_TYPE_GPIOB（GPIO16 ～ GPIO31） • APP_IO_TYPE_GPIOC（GPIO32 ～ GPIO33） • APP_IO_TYPE_AON • APP_IO_TYPE_MSIO • APP_IO_TYPE_MAX（参数检测）
uint32_t pin	选择IO引脚	同2.2.1.1 app_io_init_t中定义
void *arg	用户参数指针	通过2.2.3.6 app_io_event_register_cb传入用户参数，中断函数可用

详细信息请参考文件：SDK_Folder\drivers\inc\app_io.h。

2.2.2 枚举

2.2.2.1 app_io_pin_state_t

枚举定义如下：

表 2-12 app_io_pin_state_t枚举成员

成员	描述
APP_IO_PIN_RESET = 0	IO低电平
APP_IO_PIN_SET = 1	IO高电平

2.2.2.2 app_io_type_t

枚举定义如下：

表 2-13 app_io_type_t枚举成员

成员	描述
APP_IO_TYPE_GPIOA	普通IO类型：GPIO0 ～ GPIO15
APP_IO_TYPE_GPIOB	普通IO类型：GPIO16 ～ GPIO31
APP_IO_TYPE_GPIOC	普通IO类型：GPIO32 ～ GPIO33，仅适用于GR5526、GR5x25
APP_IO_TYPE_NORMAL	普通IO类型：GPIO0 ～ GPIO32，仅适用于GR551x
APP_IO_TYPE_AON	具有唤醒功能的IO类型：AON_GPIO0 ～ AON_GPIO7
APP_IO_TYPE_MSIO	具有模拟输入功能的IO类型：MSIO0 ～ MSIO7

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述
APP_IO_TYPE_MAX	IO类型总数，用作参数检测

2.2.2.3 app_io_mode_t

枚举定义如下：

表 2-14 app_io_mode_t枚举成员

成员	描述
APP_IO_MODE_NONE	不设置模式
APP_IO_MODE_INPUT	IO输入模式
APP_IO_MODE_OUTPUT	IO输出模式
APP_IO_MODE_MUX	外设功能复用模式
APP_IO_MODE_IT_RISING	中断上升沿触发模式
APP_IO_MODE_IT_FALLING	中断下降沿触发模式
APP_IO_MODE_IT_BOTH_EDGE	中断上升下降沿触发模式
APP_IO_MODE_IT_HIGH	中断高电平触发模式
APP_IO_MODE_IT_LOW	中断低电平触发模式
APP_IO_MODE_ANALOG	模拟输入模式
APP_IO_MODE_MAX	IO模式总数，用作参数检测

2.2.2.4 app_io_pull_t

枚举定义如下：

表 2-15 app_io_pull_t枚举成员

成员	描述
APP_IO_NOPULL	浮空
APP_IO_PULLUP	上拉
APP_IO_PULLDOWN	下拉
APP_IO_PULL_MAX	PULL模式总数，用作参数检测

2.2.2.5 app_io_mux_t

枚举定义如下：

表 2-16 app_io_mux_t枚举成员

成员	描述
APP_IO_MUX_0～APP_IO_MUX_54	MUX模式0～54，具体功能请参考芯片Datasheet

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.2.3 接口函数

表 2-17 IO驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_io_init	Y	Y	Y	Y	Y
app_io_deinit	Y	Y	Y	Y	Y
app_io_read_pin	Y	Y	Y	Y	Y
app_io_write_pin	Y	Y	Y	Y	Y
app_io_toggle_pin	Y	Y	Y	Y	Y
app_io_event_register_cb	Y	Y	Y	Y	Y
app_io_event_unregister	Y	Y	Y	Y	Y
app_io_set_speed	Y	Y	Y	Y	Y
app_io_set_strength	Y	Y	Y	Y	Y
app_io_set_intput_type	Y	Y	Y	Y	Y

2.2.3.1 app_io_init

表 2-18 app_io_init接口

函数原型	uint16_t app_io_init(app_io_type_t type, app_io_init_t *p_init)
功能说明	初始化IO
输入参数	• type：IO类型 • p_init：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.2.3.2 app_io_deinit

表 2-19 app_io_deinit接口

函数原型	uint16_t app_io_deinit(app_io_type_t type, uint32_t pin)
功能说明	反初始化IO
输入参数	• type：IO类型 • pin：IO引脚
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.2.3.3 app_io_read_pin

表 2-20 app_io_read_pin接口

函数原型	app_io_pin_state_t app_io_read_pin(app_io_type_t type, uint32_t pin)
功能说明	读取IO电平
输入参数	• type：IO类型 • pin：IO引脚
返回值	详见2.2.2.1 app_io_pin_state_t枚举
备注	

2.2.3.4 app_io_write_pin

表 2-21 app_io_write_pin接口

函数原型	uint16_t app_io_write_pin(app_io_type_t type, uint32_t pin, app_io_pin_state_t pin_state)
功能说明	使能IO输出高低电平
输入参数	• type：IO类型 • pin：IO引脚 • pin_state：输出高/低电平
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.2.3.5 app_io_toggle_pin

表 2-22 app_io_toggle_pin接口

函数原型	uint16_t app_io_toggle_pin(app_io_type_t type, uint32_t pin)
功能说明	切换IO电平： • 若IO电平为高电平，则调用此函数使能IO输出低电平 • 若IO电平为低电平，则调用此函数使能IO输出高电平
输入参数	• type：IO类型 • pin：IO引脚
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.2.3.6 app_io_event_register_cb

表 2-23 app_io_event_register_cb接口

函数原型	uint16_t app_io_event_register_cb(app_io_type_t type, app_io_init_t *p_init, app_io_callback_t io_evt_cb, void *arg)
功能说明	初始化IO，设置IO中断模式，并注册回调函数
输入参数	• type：IO类型

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• pin_init：初始化参数的结构体指针 • io_evt_cb：IO回调函数，由用户自定义 • arg：任意指针，用于用户传参
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	• 回调函数形式可参照typedef void (*app_io_callback_t)(app_io_evt_t *p_evt)。 • HAL层已实现中断标志位的清理，用户无需关心中断标志的清理。回调函数根据返回事件类型进行相应的处理即可。

2.2.3.7 app_io_event_unregister

表 2-24 app_io_event_unregister接口

函数原型	uint16_t app_io_event_unregister(app_io_type_t type, uint32_t pin)
功能说明	反初始化IO，并注销中断回调函数
输入参数	• type：IO类型 • pin：IO引脚
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.2.3.8 app_io_set_speed

表 2-25 app_io_set_speed接口

函数原型	uint16_t app_io_set_speed(app_io_type_t type, uint32_t pin, app_io_speed_t speed)
功能说明	设置IO速度
输入参数	• type：IO类型 • pin：IO引脚 • speed：IO速度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.2.3.9 app_io_set_strength

表 2-26 app_io_set_strength接口

函数原型	uint16_t app_io_set_strength(app_io_type_t type, uint32_t pin, app_io_strength_t strength)
功能说明	设置IO驱动强度
输入参数	• type：IO类型 • pin：IO引脚 • strength：IO驱动强度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.2.3.10 app_io_set_intput_type

表 2-27 app_io_set_intput_type接口

函数原型	uint16_t app_io_set_intput_type(app_io_type_t type, uint32_t pin, app_io_input_type_t input_type)
功能说明	设置IO输入特性（CMOS 输入或Schmitt 输入）
输入参数	• type：IO类型 • pin：IO引脚 • input_type：IO输入特性
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.2.4 示例工程

2.2.4.1 GPIO

本示例工程将IO复用成GPIO，实现GPIO输入输出和电平变换，工程路径：SDK_Folder\projects\peripheral\gpio\app_gpio。

2.2.4.1.1 简介

示例工程配置4个GPIO的PIN脚，PIN0和PIN1为输出模式，PIN2和PIN3为输入模式，连接方式如图所示：

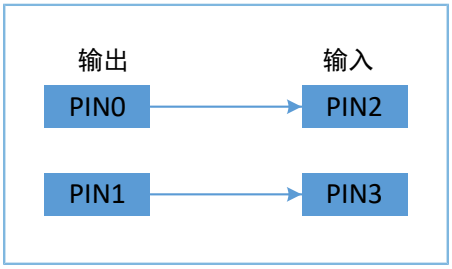

图 2-2 GPIO引脚配置及连接

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.2.4.1.2 流程图

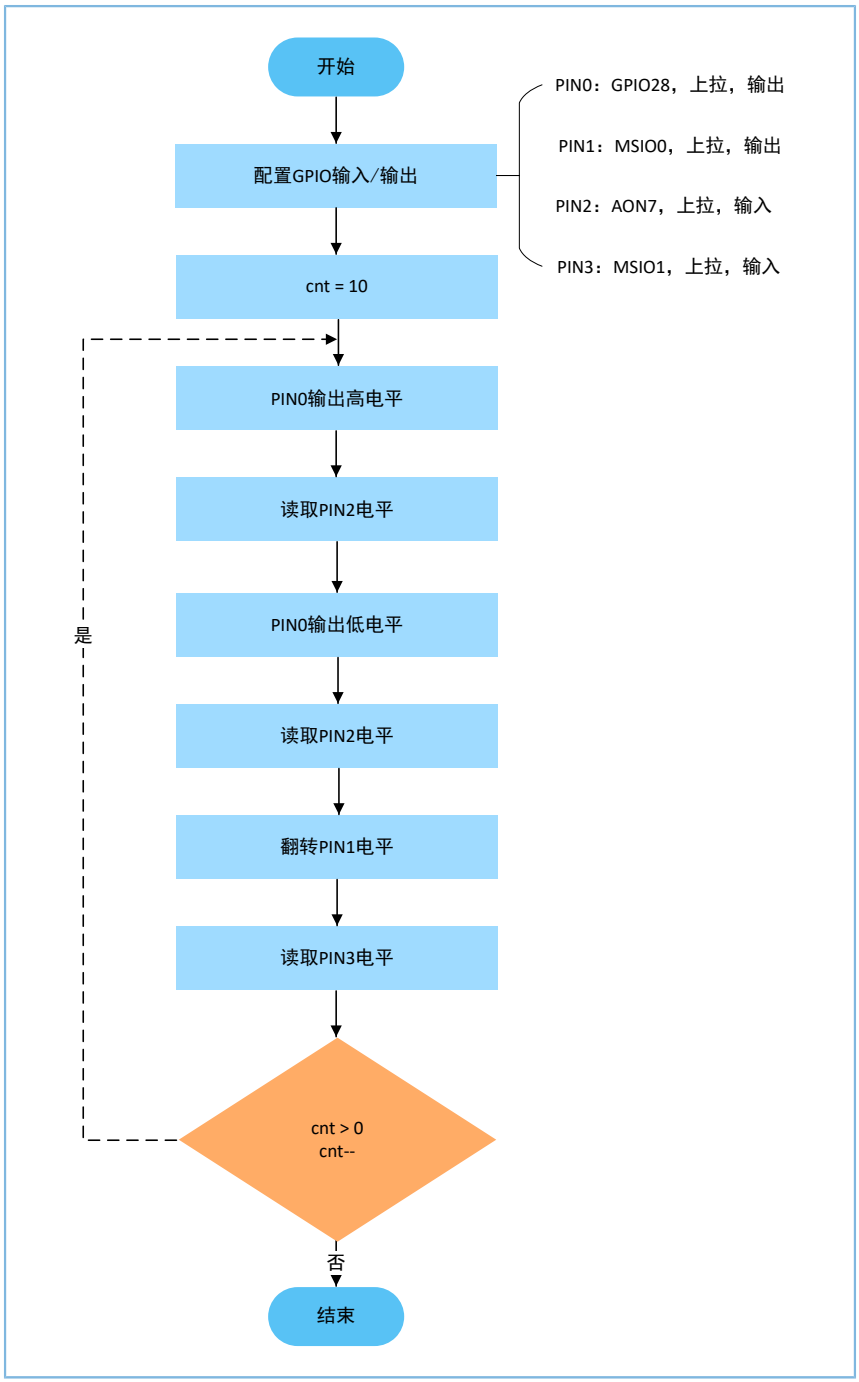

图 2-3 GPIO输入输出和电平变换流程

2.2.4.1.3 初始化参数

1. IO引脚定义

```
#define APP_GPIO_PIN0            APP_IO_PIN_12     /* GPIO28 */
#define APP_GPIO_PIN0_TYPE       APP_IO_TYPE_GPIOB

#define APP_GPIO_PIN1            APP_IO_PIN_0      /* MSIO0 */
#define APP_GPIO_PIN1_TYPE       APP_IO_TYPE_MSIO
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
#define APP_GPIO_PIN2            APP_IO_PIN_7      /* AON7 */
#define APP_GPIO_PIN2_TYPE       APP_IO_TYPE_AON

#define APP_GPIO_PIN3            APP_IO_PIN_1      /* MSIO1 */
#define APP_GPIO_PIN3_TYPE       APP_IO_TYPE_MSIO
```

为兼容GR551x SDK V1.7.0及之前的版本，GR551x保留APP_IO_TYPE_NORMAL参数，即GPIO28引脚配置有两种方式：

- 方式一：

```
#define APP_GPIO_PIN0            APP_IO_PIN_12     /* GPIO28 */
#define APP_GPIO_PIN0_TYPE       APP_IO_TYPE_GPIOB
```

- 方式二：

```
#define APP_GPIO_PIN0            APP_IO_PIN_28     /* GPIO28 */
#define APP_GPIO_PIN0_TYPE       APP_IO_TYPE_NORMAL
```

2. 初始化

   下面以GPIO28为例介绍IO初始化。

```
app_io_init_t io_init = APP_IO_DEFAULT_CONFIG;
io_init.pull = APP_IO_PULLUP;
io_init.mode = APP_IO_MODE_OUTPUT;
io_init.pin  = APP_GPIO_PIN0;
io_init.mux  = APP_IO_MUX;
ret = app_io_init(APP_GPIO_PIN0_TYPE, &io_init);
```

- pull：上/下拉电阻配置，选择APP_IO_PULLUP（上拉）。
- mode：I/O。
- pin：IO引脚选择。
- mux：引脚映射配置，具体选择可参考芯片Datasheet中Pin Mux配置表。示例中不同芯片复用模式的配置，请参考SDK_Folder\platform\boards\board_SK.h。

提示:

具体配置可参考2.2.1.1 app_io_init_t结构体描述。

2.2.4.1.4 重要函数

- 初始化IO配置

```
ret = app_io_init(APP_GPIO_PIN0_TYPE, &io_init);
```

- 设置IO输出电平

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
app_io_write_pin(APP_GPIO_PIN0_TYPE, APP_GPIO_PIN0, APP_IO_PIN_SET);
```

APP_IO_PIN_SET表示高电平；APP_IO_PIN_RESET表示低电平。

- 读取IO电平

```
pin_val = app_io_read_pin(APP_GPIO_PIN2_TYPE, APP_GPIO_PIN2)
```

pin_val为0表示低电平，为1表示高电平。

- 电平翻转

```
app_io_toggle_pin(APP_GPIO_PIN1_TYPE, APP_GPIO_PIN1)
```

翻转IO电平：高电平翻转为低电平，低电平翻转为高电平。

2.2.4.1.5 测试验证

1. 测试步骤

   (1) 根据示例工程代码连接引脚。

   (2) 连接开发板到PC，打开串口调试助手（如GRUart），连接串口，查看日志是否与预期结果一致。

2. 测试结果

   结果如下所示，GPIO的电平按照预期进行变化。

```
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=1
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=0
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=1
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=0
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=1
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=0
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=1
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
```

```
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=0
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=1
APP_GPIO_PIN0 set high, APP_GPIO_PIN2 input value=1
APP_GPIO_PIN0 set low, APP_GPIO_PIN2 input value=0
APP_GPIO_PIN1 toggle, APP_GPIO_PIN3 input value=0
```

2.2.4.2 GPIO Interrupt

示例工程实现按键输入功能，工程路径：SDK_Folder\projects\peripheral\gpio\app_gpio。

2.2.4.2.1 简介

配置GPIO为输入模式，低电平触发中断，即当按键按下，产生低电平输入，触发中断。

2.2.4.2.2 流程图

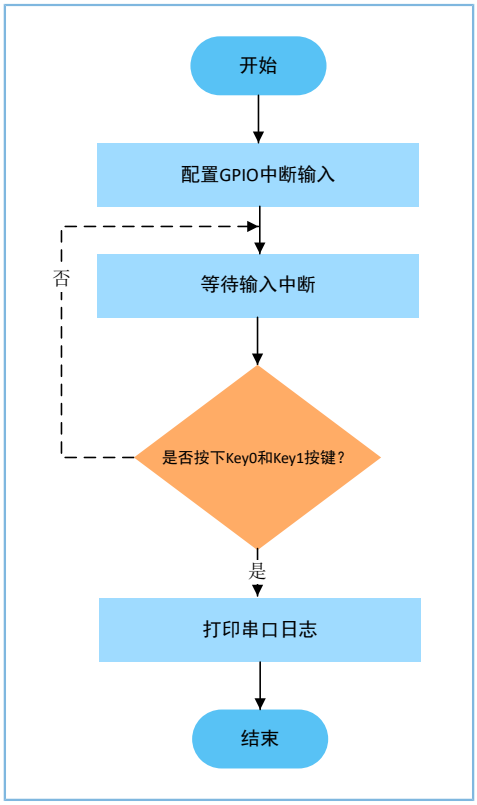

图 2-4 IO按键输入

2.2.4.2.3 初始化参数

1. IO引脚定义：

   定义key0和key1按键输入引脚。

```
#define APP_GPIO_KEY0_PIN         APP_KEY_UP_PIN
#define APP_GPIO_KEY0_TYPE        APP_KEY_UP_IO_TYPE
```

```
#define APP_GPIO_KEY1_PIN        APP_KEY_OK_PIN
#define APP_GPIO_KEY1_TYPE       APP_KEY_OK_IO_TYPE
```

具体引脚定义，参见*board_SK.h*。GR5526引脚定义如下，key0按键输入引脚为AON_GPIO_1，key1按键输入为AON_GPIO_0。

```
/*******KEY DRIVER IO CONFIG*******************/
#define APP_KEY_UP_IO_TYPE                  APP_IO_TYPE_AON
#define APP_KEY_DOWN_IO_TYPE                APP_IO_TYPE_AON
#define APP_KEY_UP_PIN                      APP_IO_PIN_1
#define APP_KEY_DOWN_PIN                    APP_IO_PIN_0
```

2. 中断配置

```
app_io_init_t io_init = APP_IO_DEFAULT_CONFIG;

    io_init.pull = APP_IO_PULLUP;
    io_init.mode = APP_IO_MODE_IT_FALLING;
    io_init.pin  = APP_GPIO_KEY0_PIN;
    io_init.mux  = APP_IO_MUX;
    ret = app_io_event_register_cb(APP_GPIO_KEY0_TYPE,&io_init, app_io_event_handler,
                                   "KEY0 pin interrupt");
```

详细GPIO配置参数，请参考2.2.4.1.3 初始化参数。其中，mode选择APP_IO_MODE_IT_FALLING（下降沿触发）。

2.2.4.2.4 重要函数

```
ret = app_io_event_register_cb(APP_GPIO_KEY0_TYPE,&io_init, app_io_event_handler,
                                                "KEY0 pin interrupt");
```

调用app_io_event_register_cb接口实现GPIO初始化和中断注册，支持参数传入。

2.2.4.2.5 测试验证

连接开发板到PC端，打开串口调试助手，连接串口。分别按下key0和key1，串口打印如下。

说明:

需根据原理图了解key0和key1对应的物理按键。

```
KEY0 pin interrupt
KEY0 pressed.
KEY1 pin interrupt
KEY1 pressed.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.3 APP GPIO_TE驱动

2.3.1 结构体

2.3.1.1 app_gpiote_param_t

结构体定义如下：

表 2-28 app_gpiote_param_t结构体成员

成员	描述	取值
app_io_type_t type	设置IO类型（GPIOA/NORMAL/GPIOB/GPIOC/AON_GPIO/MSIO）	参考2.2.1.2 app_io_evt_t中定义
uint32_t pin	选择IO引脚	参考2.2.1.1 app_io_init_t中定义
app_io_mode_t mode	中断触发模式	该参数的取值可以是下列值中的任意一个： • APP_IO_MODE_IT_RISING（上升沿触发） • APP_IO_MODE_IT_FALLING（下降沿触发） • APP_IO_MODE_IT_BOTH_EDGE（上升下降沿触发，GR551x没有该模式） • APP_IO_MODE_IT_HIGH（高电平触发） • APP_IO_MODE_IT_LOW（低电平触发）
app_io_pull_t pull	设置IO上拉/下拉/浮空	该参数的取值可以是下列值中的任意一个： • APP_IO_NOPULL（浮空） • APP_IO_PULLUP（上拉） • APP_IO_PULLDOWN（下拉）
app_io_callback_t io_evt_cb	中断回调函数指针	用户自定义

详细信息请参考文件：`SDK_Folder\drivers\inc\app_gpiote.h`。

2.3.2 接口函数

表 2-29 GPIO_TE驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_gpiote_init	Y	Y	Y	Y	Y
app_gpiote_config	Y	Y	Y	Y	Y
app_gpiote_deinit	Y	Y	Y	Y	Y

2.3.2.1 app_gpiote_init

表 2-30 app_gpiote_init接口

函数原型	uint16_t app_gpiote_init(const app_gpiote_param_t *p_params, uint8_t table_cnt)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	初始化多组GPIO，并配置多组GPIO中断，并注册中断回调函数
输入参数	• p_params：初始化参数的结构体指针 • table_cnt：配置的IO数量
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_io_callback_t)(app_io_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

2.3.2.2 app_gpiote_config

表 2-31 app_gpiote_config接口

函数原型	uint16_t app_gpiote_config(const app_gpiote_param_t *p_config)
功能说明	初始化GPIO，并配置GPIO中断，并注册中断回调函数（单个）
输入参数	p_config：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.3.2.3 app_gpiote_deinit

表 2-32 app_gpiote_deinit接口

函数原型	uint16_t app_gpiote_deinit(app_io_type_t type, uint32_t pin)
功能说明	反初始化GPIO，并注销中断回调函数（单个）
输入参数	• type：IO类型 • pin：IO引脚
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.3.3 示例工程

2.3.3.1 GPIO_TE

示例工程主要调用GPIO_TE驱动接口，配置GPIO中断输入模式，实现按键输入功能。工程路径：SDK_Folder\projects\peripheral\gpio\app_gpio。

2.3.3.1.1 简介

同2.2.4.2.1 简介。

2.3.3.1.2 流程图

同2.2.4.2.2 流程图。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.3.3.1.3 初始化参数

1. IO引脚定义：

   定义key0和key1按键输入引脚。

```
#define APP_GPIO_KEY0_PIN        APP_KEY_UP_PIN
#define APP_GPIO_KEY0_TYPE       APP_KEY_UP_IO_TYPE

#define APP_GPIO_KEY1_PIN        APP_KEY_OK_PIN
#define APP_GPIO_KEY1_TYPE       APP_KEY_OK_IO_TYPE
```

   具体引脚定义，参见*board_SK.h*，如下key0按键输入引脚为AON_GPIO_1；key1按键输入为AON_GPIO_0。

```
/*******KEY DRIVER IO CONFIG*******************/
#define APP_KEY_UP_IO_TYPE                  APP_IO_TYPE_AON
#define APP_KEY_DOWN_IO_TYPE                APP_IO_TYPE_AON
#define APP_KEY_UP_PIN                      APP_IO_PIN_1
#define APP_KEY_DOWN_PIN                    APP_IO_PIN_0
```

2. 定义初始化结构体，用于中断引脚、中断方式和中断回调函数。

```
const app_gpiote_param_t gpiote_param[] =
{
    {APP_GPIO_KEY0_TYPE, APP_GPIO_KEY0_PIN, APP_IO_MODE_IT_FALLING,
                            APP_IO_PULLUP, app_io_event_handler},
    {APP_GPIO_KEY1_TYPE, APP_GPIO_KEY1_PIN, APP_IO_MODE_IT_FALLING,
                            APP_IO_PULLUP, app_io_event_handler},
};
```

2.3.3.1.4 重要函数

实现GPIO中断配置

```
app_gpiote_init(gpiote_param, sizeof(gpiote_param) / sizeof(app_gpiote_param_t))
```

2.3.3.1.5 测试验证

连接开发板到PC端，打开串口调试助手，连接串口。分别按下key0和key1，串口打印如下。

说明:

需根据原理图了解key0和key1对应的物理按键。

```
KEY0 pressed.

KEY1 pressed.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.4 APP DMA驱动

2.4.1 结构体

2.4.1.1 app_dma_params_t

结构体定义如下：

表 2-33 app_dma_params_t结构体成员

成员	描述	取值
dma_regs_t *p_instance	DMA实例	该参数的取值可以是下列值中的任意一个： GR5526/GR5x25： • DMA0 • DMA1 GR551x/GR533x/GR5405： • DMA0
dma_channel_t channel_number	DMA通道	该参数的取值可以是下列值中的任意一个： GR5526/GR5x25： • DMA_Channel0 • DMA_Channel1 • DMA_Channel2 • DMA_Channel3 • DMA_Channel4 • DMA_Channel5 GR533x/GR5405： • DMA_Channel0 • DMA_Channel1 • DMA_Channel2 • DMA_Channel3 • DMA_Channel4 GR551x： • DMA_Channel0 • DMA_Channel1 • DMA_Channel2 • DMA_Channel3 • DMA_Channel4 • DMA_Channel5 • DMA_Channel6 • DMA_Channel7 **说明：** 不采用链式传输方式，每个通道一次最大搬运长度为4095拍。

成员	描述	取值
dma_init_t init	DMA配置	参考2.4.1.2 dma_init_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_dma.h。

各Bluetooth LE芯片中DMA支持的外设如下表，外设在使用DMA传输方式时，需要根据下表选择支持的DMA实例。其中xx_M表示仅可作Master端，xx_S表示仅可作Slave端。

表 2-34 DMA支持的外设

芯片	DMA0	DMA1
GR551x	QSPI0_M	/
	QSPI1_M	/
	SPI_M	/
	SPI_S	/
	UART0	/
	UART1	/
	I2C0	/
	I2C1	/
	I2S_M	/
	I2S_S	/
	ADC	/
GR5526	QSPI0_M	QSPI1_M
	QSPI1_M	QSPI2_M
	OSPI_M	OSPI_M
	SPI_M	SPI_M
	SPI_S	DSPI_M
	UART0	UART0
	UART1	UART3
	UART2	UART4
	UART3	UART5
	UART4	I2S_M
	I2C2	I2S_S
	I2C3	PDM
	I2C4	I2C0
	I2C5	I2C1
	ADC	/
GR5x25	QSPI0_M	QSPI1_M
	QSPI1_M	QSPI2_M
	SPI_M	SPI_M

版权所有 © 2025 深圳市汇顶科技股份有限公司

芯片	DMA0	DMA1
	SPI_S	DSPI_M
	UART0	UART0
	UART1	UART3
	UART2	I2S_M
	UART3	I2S_S
	I2C2	PDM
	I2C3	I2C0
	ADC	I2C1
GR533x/GR5405	SPI_M	/
	SPI_S	/
	UART0	/
	UART1	/
	I2C0	/
	I2C1	/
	ADC	/
	PWM0	/

2.4.1.2 dma_init_t

结构体定义如下：

表 2-35 dma_init_t结构体成员

成员	描述	取值
uint32_t src_request	源端硬件请求信号	0x0 ～ 0xF（默认0x0）
uint32_t dst_request	目的端硬件请求信号	0x0 ～ 0xF（默认0x0）
uint32_t direction	数据传输方向	该参数的取值可以是下列值中的任意一个： • DMA_MEMORY_TO_MEMORY（内存到内存） • DMA_MEMORY_TO_PERIPH（内存到外设） • DMA_PERIPH_TO_MEMORY（外设到内存） • DMA_PERIPH_TO_PERIPH（外设到外设）
uint32_t src_increment	源地址更新方式	该参数的取值可以是下列值中的任意一个： • DMA_SRC_INCREMENT（源地址递增） • DMA_SRC_DECREMENT（源地址递减） • DMA_SRC_NO_CHANGE（源地址不变）
uint32_t dst_increment	目的地址更新方式	该参数的取值可以是下列值中的任意一个： • DMA_DST_INCREMENT（目的地址递增） • DMA_DST_DECREMENT（目的地址递减） • DMA_DST_NO_CHANGE（目的地址不变）
uint32_t src_data_alignment	源数据传输位宽	该参数的取值可以是下列值中的任意一个：

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• DMA_SDATAALIGN_BYTE（8 bits） • DMA_SDATAALIGN_HALFWORD（16 bits） • DMA_SDATAALIGN_WORD（32 bits）
uint32_t dst_data_alignment	目的数据传输位宽	该参数的取值可以是下列值中的任意一个： • DMA_DDATAALIGN_BYTE（8 bits） • DMA_DDATAALIGN_HALFWORD（16 bits） • DMA_DDATAALIGN_WORD（32 bits）
uint32_t mode	传输模式	该参数的取值可以是下列值中的任意一个： • DMA_NORMAL（普通模式） • DMA_CIRCULAR（循环模式）
uint32_t priority	优先级	该参数的取值可以是下列值中的任意一个： • DMA_PRIORITY_LOW（低优先级） • DMA_PRIORITY_MEDIUM（中优先级） • DMA_PRIORITY_HIGH（高优先级） • DMA_PRIORITY_VERY_HIGH（最高优先级）

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_dma.h
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_dma.h
- GR5405：SDK_Folder\hal_drv\include\hal_dma.h

2.4.1.3 dma_sg_llp_config_t

结构体定义如下：

表 2-36 dma_sg_llp_config_t结构体成员

成员	描述	取值
dma_scatter_config_t scatter_config	分散传输配置	参考2.4.1.4 dma_scatter_config_t结构体
dma_gather_config_t gather_config	聚合传输配置	参考2.4.1.5 dma_gather_config_t结构体
dma_llp_config_t llp_config	传输链表配置	参考2.4.1.6 dma_llp_config_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_dma.h。

2.4.1.4 dma_scatter_config_t

结构体定义如下：

表 2-37 dma_scatter_config_t结构体成员

成员	描述	取值
uint32_t dst_scatter_en	目的端分散使能	该参数的取值可以是下列值中的任意一个：

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• DMA_DST_SCATTER_ENABLE • DMA_DST_SCATTER_DISABLE
uint32_t dst_dsi	目的端分散间隔	0x0 ～ 0xFFFFF
uint32_t dst_dsc	目的端分散计数	0x0 ～ 0xFFF

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_dma.h。

2.4.1.5 dma_gather_config_t

结构体定义如下：

表 2-38 dma_gather_config_t结构体成员

成员	描述	取值
uint32_t src_gather_en	源端聚合使能	该参数的取值可以是下列值中的任意一个： • DMA_SRC_GATHER_ENABLE • DMA_SRC_GATHER_DISABLE
uint32_t src_sgi	源端聚合间隔	0x0 ～ 0xFFFFF
uint32_t src_sgc	源端聚合计数	0x0 ～ 0xFFF

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_dma.h。

2.4.1.6 dma_llp_config_t

结构体定义如下：

表 2-39 dma_llp_config_t结构体成员

成员	描述	取值
uint32_t llp_src_en	源端传输链表使能	该参数的取值可以是下列值中的任意一个： • DMA_LLP_SRC_ENABLE • DMA_LLP_SRC_DISABLE
uint32_t llp_src_writeback	源端回写状态值	0x0 ～ 0xFFFFFFFF
uint32_t llp_dst_en	目的端传输链表使能	该参数的取值可以是下列值中的任意一个： • DMA_LLP_DST_ENABLE • DMA_LLP_DST_DISABLE
uint32_t llp_dst_writeback	目的端回写状态值	0x0 ～ 0xFFFFFFFF
struct dma_block_config *head_lli	传输链表头	参考2.4.1.7 dma_block_config结构体

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_dma.h。

2.4.1.7 dma_block_config

结构体定义如下：

表 2-40 dma_block_config结构体成员

成员	描述	取值
uint32_t src_address	源端地址	0x0 ～ 0xFFFFFFFF
uint32_t dst_address	目的端地址	0x0 ～ 0xFFFFFFFF
struct dma_block_config *p_lli	传输链表下一个节点地址	0x0 ～ 0xFFFFFFFF
uint32_t CTL_L	控制寄存器低32位配置	0x0 ～ 0xFFFFFFFF
uint32_t CTL_H	控制寄存器高32位配置	0x0 ～ 0xFFFFFFFF
uint32_t src_status	源端传输状态（回写）	0x0 ～ 0xFFFFFFFF
uint32_t dst_status	目的端传输状态（回写）	0x0 ～ 0xFFFFFFFF

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_dma.h。

2.4.2 枚举

2.4.2.1 app_dma_evt_type_t

枚举定义如下：

表 2-41 app_dma_evt_type_t枚举成员

成员	描述
APP_DMA_EVT_ERROR = 0	DMA传输错误事件
APP_DMA_EVT_TFR = 1	DMA传输完成事件
APP_DMA_EVT_BLK = 2	DMA块传输完成事件

2.4.2.2 app_dma_state_t

枚举定义如下：

表 2-42 app_dma_state_t枚举成员

成员	描述
APP_DMA_INVALID = 0	DMA不可用
APP_DMA_ACTIVITY = 1	DMA激活
APP_DMA_SLEEP = 2	DMA睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_dma.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.4.3 接口函数

表 2-43 DMA驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_dma_init	Y	Y	Y	Y	Y
app_dma_deinit	Y	Y	Y	Y	Y
app_dma_start	Y	Y	Y	Y	Y
app_dma_get_handle	Y	Y	Y	Y	Y
dma_wake_up	Y	Y	Y	Y	Y
app_dma_start_sg_llp	N	Y	Y	N	N

2.4.3.1 app_dma_init

表 2-44 app_dma_init接口

函数原型	dma_id_t app_dma_init(app_dma_params_t *p_params, app_dma_evt_handler_t evt_handler)
功能说明	初始化DMA
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自行实现的中断回调函数
返回值	dma_id_t：DMA通道ID
备注	回调函数形式参照typedef void (*app_dma_evt_handler_t)(app_dma_evt_type_t type)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

2.4.3.2 app_dma_deinit

表 2-45 app_dma_deinit接口

函数原型	uint16_t app_dma_deinit(dma_id_t id)
功能说明	反初始化DMA
输入参数	id：DMA通道ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.4.3.3 app_dma_start

表 2-46 app_dma_start接口

函数原型	uint16_t app_dma_start(dma_id_t id, uint32_t src_address, uint32_t dst_address, uint32_t data_length)
功能说明	开启DMA传输
输入参数	• id：DMA通道ID

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• src_address：源地址 • dst_address：目的地址 • data_length：数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.4.3.4 app_dma_get_handle

表 2-47 app_dma_get_handle接口

函数原型	dma_handle_t *app_dma_get_handle(dma_id_t id)
功能说明	获取DMA句柄
输入参数	id：DMA通道ID
返回值	DMA句柄,详见： • GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_dma.h • GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_dma.h • GR5405：SDK_Folder\hal_drv\include\hal_dma.h
备注	

2.4.3.5 dma_wake_up

表 2-48 dma_wake_up接口

函数原型	void dma_wake_up(dma_id_t id)
功能说明	系统唤醒后，唤醒DMA模块
输入参数	id：DMA通道ID
返回值	
备注	

2.4.3.6 app_dma_start_sg_llp

表 2-49 app_dma_start_sg_llp接口

函数原型	uint16_t app_dma_start_sg_llp(dma_id_t id, uint32_t src_address, uint32_t dst_address, uint32_t data_length, dma_sg_llp_config_t *sg_llp_config)
功能说明	开启DMA传输，带有分散聚合与链表传输配置
输入参数	• id：DMA通道ID • src_address：源地址 • dst_address：目的地址 • data_length：数据长度 • sg_llp_config：DMA分散聚合配置，详见2.4.1.3 dma_sg_llp_config_t结构体
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.4.4 示例工程

2.4.4.1 DMA Memory

示例工程实现DMA方式将数据从一块内存搬运到另一块内存，工程路径：SDK_Folder\projects\peripheral\dma\app_dma。

2.4.4.1.1 流程图

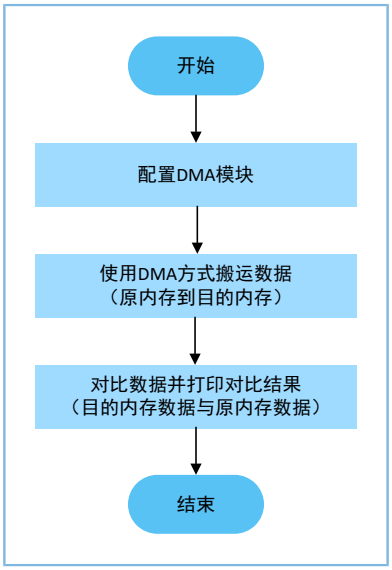

图 2-5 DMA实现数据拷贝流程

2.4.4.1.2 初始化参数

```
app_dma_params_t dma_params = {DMA0,DMA_Channel0, 0};
dma_params.p_instance                = DMA0;
dma_params.channel_number            = DMA_Channel0;
dma_params.init.direction            = DMA_MEMORY_TO_MEMORY;
dma_params.init.src_increment        = DMA_SRC_INCREMENT;
dma_params.init.dst_increment        = DMA_DST_INCREMENT;
dma_params.init.src_data_alignment   = DMA_SDATAALIGN_BYTE;
dma_params.init.dst_data_alignment   = DMA_DDATAALIGN_BYTE;
dma_params.init.priority             = DMA_PRIORITY_LOW;
s_dma_channel_id = app_dma_init(&dma_params, app_dma_callback);
```

- p_instance ： DMA模块ID，选择DMA0。
- channel_number：DMA通道，选择DMA_Channel0，即通道0。
- init.direction：DMA传输方向，选择DMA_MEMORY_TO_MEMORY，即传输方向内存到内存。
- init.src_increment：源地址更新方式，选择DMA_SRC_INCREMENT，即源端地址递增。
- init.dst_increment：目的地址更新方式，选择DMA_DST_INCREMENT，即目的端地址递增。
- init.src_data_alignment：源数据宽度，选择DMA_SDATAALIGN_BYTE，即源数据宽度8 bits。

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.dst_data_alignment：目标数据宽度，选择DMA_DDATAALIGN_BYTE，即目标数据宽度8 bits。

其他可选配置请参考2.4.1 结构体相关定义。

2.4.4.1.3 重要函数

- DMA初始化

```
s_dma_channel_id = app_dma_init(&dma_params, app_dma_callback);
```

  - dma_params：初始化参数结构体
  - app_dma_callback：用户自定义的回调函数
  - s_dma_channel_id：返回的DMA通道ID

- DMA数据传输

```
app_dma_start(s_dma_channel_id, (uint32_t)&g_src_data, (uint32_t)&g_dst_data,
 DMA_DATA_LEN);
```

  - g_src_data：源数据数组
  - g_dst_data：目的数据数组
  - DMA_DATA_LEN：传输数据的长度

2.4.4.1.4 测试验证

连接开发板到PC端，打开串口调试助手，连接串口。串口日志显示DMA搬运数据的对比结果，如下：

```
app_dma0_memory_to_memory: success
app_dma1_memory_to_memory: success

This example demo end.
```

2.5 APP UART驱动

2.5.1 宏定义

2.5.1.1 TX_ONCE_MAX_SIZE

表 2-50 TX_ONCE_MAX_SIZE宏定义

描述	值
定义一次传输的最大字节数	128 **说明：** • 该值表示单次发送缓冲区的最大长度。 • 异步发送时，数据会先放入环形缓冲区，再从环形缓冲区拷贝到单次发送缓冲区。 • 在DMA模式下，TX_ONCE_MAX_SIZE值将影响触发DMA中断的频率。

2.5.2 结构体

2.5.2.1 app_uart_pin_t

结构体定义如下：

表 2-51 app_uart_pin_t结构体成员

成员	描述	取值
app_io_type_t type	设置IO类型	参考2.2.1.2 app_io_evt_t中描述
app_io_mux_t mux	IO映射功能配置	参考2.2.1.1 app_io_init_t结构体，具体映射模式参考芯片Datasheet中的Pin Mux
uint32_t pin	选择IO引脚	参考2.2.1.1 app_io_init_t结构体，具体根据芯片Datasheet中UART资源进行选择
app_io_pull_t pull	设置IO上拉/下拉/浮空	参考2.2.1.1 app_io_init_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.2.2 app_uart_pin_cfg_t

结构体定义如下：

表 2-52 app_uart_pin_cfg_t结构体成员

成员	描述	取值
app_uart_pin_t tx	UART_TX引脚配置	参考2.5.2.1 app_uart_pin_t结构体
app_uart_pin_t rx	UART_RX引脚配置	参考2.5.2.1 app_uart_pin_t结构体
app_uart_pin_t cts	UART_CTS引脚配置	参考2.5.2.1 app_uart_pin_t结构体
app_uart_pin_t rts	UART_RTS引脚配置	参考2.5.2.1 app_uart_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.2.3 app_uart_dma_cfg_t

结构体定义如下：

表 2-53 app_uart_dma_cfg_t结构体成员

成员	描述	取值
dma_reg_t *tx_dma_instance	UART_TX指定DMA模块配置	该参数的取值可以是下列值中的任意一个： • DMA0 • DMA1 具体选择参考2.4.1.1 app_dma_params_t章节中，DMA支持的外设。
dma_reg_t *rx_dma_instance	UART_RX指定DMA模块配置	该参数的取值可以是下列值中的任意一个： • DMA0

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• DMA1 具体选择参考2.4.1.1 app_dma_params_t章节中，DMA支持的外设。
dma_channel_t tx_dma_channel	UART_TX的DMA通道选择	参考2.4.1.1 app_dma_params_t中通道描述
dma_channel_t rx_dma_channel	UART_RX的DMA通道选择	参考2.4.1.1 app_dma_params_t中通道描述

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.2.4 app_uart_params_t

结构体定义如下：

表 2-54 app_uart_params_t结构体成员

成员	描述	取值
app_uart_id_t id	设置使用的UART模块ID	• APP_UART_ID_0（UART0） • APP_UART_ID_1（UART1） • APP_UART_ID_2（UART2） • APP_UART_ID_3（UART3） • APP_UART_ID_4（UART4） • APP_UART_ID_5（UART5）
app_uart_pin_cfg_t pin_cfg	UART引脚功能设置	参考2.5.2.2 app_uart_pin_cfg_t结构体
app_uart_dma_cfg_t dma_cfg	UART DMA传输参数配置	参考2.5.2.3 app_uart_dma_cfg_t结构体
uart_init_t init	UART初始化参数配置	参考2.5.2.5 uart_init_t结构体
uart_env_t uart_dev	保存相关数据和句柄，不需要用户配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.2.5 uart_init_t

结构体定义如下：

表 2-55 uart_init_t结构体成员

成员	描述	取值
uint32_t baud_rate	波特率	0～4000000（默认115200），最大支持波特率请参考对应芯片Datasheet手册。
uint32_t data_bits	数据位	该参数的取值可以是下列值中的任意一个： • UART_DATABITS_5（5 bits） • UART_DATABITS_6（6 bits） • UART_DATABITS_7（7 bits）

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• UART_DATABITS_8（8 bits）
uint32_t stop_bits	停止位	该参数的取值可以是下列值中的任意一个： • UART_STOPBITS_1（1 bit） • UART_STOPBITS_1_5（1.5 bits） • UART_STOPBITS_2（2 bits）
uint32_t parity	奇偶校验位	该参数的取值可以是下列值中的任意一个： • UART_PARITY_NONE（无校验位） • UART_PARITY_ODD（奇校验） • UART_PARITY_EVEN（偶校验） • UART_PARITY_SP0（0校验） • UART_PARITY_SP1（1校验）
uint32_t hw_flow_ctrl	硬件流控使能位	该参数的取值可以是下列值中的任意一个： • UART_HWCONTROL_NONE（无硬件流控） • UART_HWCONTROL_RTS_CTS（硬件流控使能）
uint32_t rx_timeout_mode	接收超时使能位	该参数的取值可以是下列值中的任意一个： • UART_RECEIVER_TIMEOUT_ENABLE（使能） • UART_RECEIVER_TIMEOUT_DISABLE（禁用）

详细信息请参考文件：

- GR551x/GR5526/GR5x25：`SDK_Folder\drivers\inc\hal\gr55xx_hal_uart.h`
- GR533x：`SDK_Folder\drivers\inc\hal\gr533x_hal_uart.h`
- GR5405：`SDK_Folder\hal_drv\include\hal_uart.h`

2.5.2.6 app_uart_evt_t

结构体定义如下：

表 2-56 app_uart_evt_t结构体成员

成员	描述	取值
app_uart_evt_type_t type	串口事件类型	该参数的取值可以是下列值中的任意一个： • APP_UART_EVT_ERROR（UART传输错误） • APP_UART_EVT_TX_CPLT（UART发送完成） • APP_UART_EVT_RX_DATA（UART接收完成） • APP_UART_EVT_ABORT_TX（UART中断发送） • APP_UART_EVT_ABORT_RX（UART中断接收） • APP_UART_EVT_ABORT_TXRX （UART中断发送和接收）
union { uint32_t error_code; unt16_t size; }data;	• uint32_t error_code：串口事件返回的错误号 • uint16_t size：串口已接收/发送的数据长度 说明：	串口错误号： • HAL_UART_ERROR_NONE（无错误） • HAL_UART_ERROR_PE（奇偶校验错误） • HAL_UART_ERROR_FE （帧错误，一般发生在接收方检测到无效的停止位）

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
	若事件类型为传输错误，此处返回错误串口引脚；若事件类型为接收/发送完成，此处显示已接收/发送的数据长度	• HAL_UART_ERROR_OE（接收溢出错误） • HAL_UART_ERROR_BI （中断错误，一般发生在RX被拉低超过1帧数据时） • HAL_UART_ERROR_DMA（DMA传输错误） • HAL_UART_ERROR_BUSY（当前串口正在传输） • HAL_UART_ERROR_INVALID_PARAM（无效参数）

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.2.7 app_uart_tx_buf_t

结构体定义如下：

表 2-57 app_uart_tx_buf_t结构体成员

成员	描述	取值
uint8_t* tx_buf	串口发送缓存	串口发送缓存的地址
uint32_t tx_buf_size	串口发送缓存大小	发送缓存的大小

说明:

在串口的异步发送中，为了方便用户可以连续调用异步接口，使用环形缓存区缓存数据。上一次发送完成后自动从环形缓冲区取出数据，存放在tx_send_buf中。TX_ONCE_MAX_SIZE为tx_send_buf大小。

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.3 枚举

2.5.3.1 app_uart_id_t

枚举定义如下：

表 2-58 app_uart_id_t枚举成员

成员	描述
APP_UART_ID_0 = 0	UART模块0
APP_UART_ID_1 = 1	UART模块1
APP_UART_ID_2 = 2	UART模块2，仅适用于GR5526、GR5x25
APP_UART_ID_3 = 3	UART模块3，仅适用于GR5526、GR5x25
APP_UART_ID_4 = 4	UART模块4，仅适用于GR5526
APP_UART_ID_5 = 5	UART模块5，仅适用于GR5526
APP_UART_ID_MAX = 6	UART模块总个数，用作参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.5.3.2 app_uart_evt_type_t

枚举定义如下：

表 2-59 app_uart_evt_type_t枚举成员

成员	描述
APP_UART_EVT_ERROR	UART传输错误事件
APP_UART_EVT_TX_CPLT	UART发送完成事件
APP_UART_EVT_RX_DATA	UART接收完成事件
APP_UART_EVT_ABORT_TX	UART发送中止事件
APP_UART_EVT_ABORT_RX	UART接收中止事件
APP_UART_EVT_ABORT_TXRX	UART发送和接收中止事件

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.3.3 app_uart_dma_state_t

枚举定义如下：

表 2-60 app_uart_dma_state_t枚举成员

成员	描述
APP_UART_DMA_INVALID = 0	UART DMA不可用
APP_UART_DMA_ACTIVITY	UART DMA激活

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

2.5.3.4 app_uart_state_t

枚举定义如下：

表 2-61 app_uart_state_t枚举成员

成员	描述
APP_UART_INVALID = 0	UART不可用
APP_UART_ACTIVITY	UART激活
APP_UART_SLEEP	UART睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_uart.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.5.4 接口函数

表 2-62 UART驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_uart_init	Y	Y	Y	Y	Y
app_uart_deinit	Y	Y	Y	Y	Y
app_uart_transmit_async	Y	Y	Y	Y	Y
app_uart_transmit_sync	Y	Y	Y	Y	Y
app_uart_receive_async	Y	Y	Y	Y	Y
app_uart_receive_sync	Y	Y	Y	Y	Y
app_uart_flush	Y	Y	Y	Y	Y
app_uart_abort	Y	Y	Y	Y	Y
app_uart_abort_transmit	Y	Y	Y	Y	Y
app_uart_abort_receive	Y	Y	Y	Y	Y
app_uart_get_handle	Y	Y	Y	Y	Y

2.5.4.1 app_uart_init

表 2-63 app_uart_init接口

函数原型	uint16_t app_uart_init(app_uart_params_t *p_params, app_uart_evt_handler_t evt_handler, app_uart_tx_buf_t *tx_buffer)
功能说明	初始化UART
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数 • tx_buffer：发送缓存
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_uart_evt_handler_t)(app_uart_evt_t *p_evt)； 注意轮询运行模式下可以不指定中断回调函数（即evt_handler的入参可为NULL）；且一次发送的数据量不应超过app_uart_tx_buf_t中定义的缓存大小； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

2.5.4.2 app_uart_deinit

表 2-64 app_uart_deinit接口

函数原型	uint16_t app_uart_deinit(app_uart_id_t id)
功能说明	反初始化UART
输入参数	id：UART模块ID（见2.5.3.1 app_uart_id_t描述）

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.4.3 app_uart_transmit_async

表 2-65 app_uart_transmit_async接口

函数原型	uint16_t app_uart_transmit_async(app_uart_id_t id, uint8_t *p_data, uint16_t size)
功能说明	串口中断方式（异步）发送数据
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：待发送数据指针 • size：待发送数据长度，单位：Byte。此长度不应大于初始化的tx_buffer长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.4.4 app_uart_transmit_sync

表 2-66 app_uart_transmit_sync接口

函数原型	uint16_t app_uart_transmit_sync(app_uart_id_t id, uint8_t *p_data, uint16_t size, uint32_t timeout)
功能说明	串口轮询方式（同步）发送数据
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：待发送数据指针 • size：待发送数据长度，单位：Byte。此长度不应大于初始化的tx_buffer长度 • timeout：超时时间（单位：ms）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.4.5 app_uart_receive_async

表 2-67 app_uart_receive_async接口

函数原型	uint16_t app_uart_receive_async(app_uart_id_t id, uint8_t *p_data, uint16_t size)
功能说明	串口中断方式（异步）接收数据
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：接收数据的指针 • size：接收数据长度，单位：Byte
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.5.4.6 app_uart_receive_sync

表 2-68 app_uart_receive_sync接口

函数原型	uint16_t app_uart_receive_sync(app_uart_id_t id, uint8_t *p_data, uint16_t size, uint32_t timeout)
功能说明	串口轮询方式（同步）接收数据
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：接收数据的指针 • size：接收数据长度，单位：Byte • timeout：超时时间（单位：ms）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.4.7 app_uart_flush

表 2-69 app_uart_flush接口

函数原型	void app_uart_flush(app_uart_id_t id)
功能说明	立即发送TX FIFO中的数据。 该功能通常用于系统进入睡眠模式之前。当数据未全部发送完成时，需立即将TX FIFO中的数据发送出去。
输入参数	id：UART模块ID（见2.5.3.1 app_uart_id_t描述）
返回值	无
备注	

2.5.4.8 app_uart_abort

表 2-70 app_uart_abort接口

函数原型	uint16_t app_uart_abort(app_uart_id_t id)
功能说明	中止串口的数据发送和接收，会产生abort中断，调用注册的回调函数，事件类型为APP_UART_EVT_ABORT_TXRX
输入参数	id：UART模块ID（见2.5.3.1 app_uart_id_t描述）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.4.9 app_uart_abort_transmit

表 2-71 app_uart_abort_transmit接口

函数原型	uint16_t app_uart_abort_transmit(app_uart_id_t id)
功能说明	中止串口的数据发送，会产生abort中断，调用注册的回调函数，事件类型为APP_UART_EVT_ABORT_TX
输入参数	id：UART模块ID（见2.5.3.1 app_uart_id_t描述）

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.4.10 app_uart_abort_receive

表 2-72 app_uart_abort_receive接口

函数原型	uint16_t app_uart_abort_receive(app_uart_id_t id)
功能说明	中止串口的数据接收，会产生abort中断，调用注册的回调函数，事件类型为APP_UART_EVT_ABORT_RX
输入参数	id：UART模块ID（见2.5.3.1 app_uart_id_t描述）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.4.11 app_uart_get_handle

表 2-73 app_uart_get_handle接口

函数原型	uart_handle_t *app_uart_get_handle(app_uart_id_t id)
功能说明	获取串口句柄
输入参数	id：UART模块ID（见2.5.3.1 app_uart_id_t描述）
返回值	串口句柄指针
备注	

2.5.5 DMA接口函数

表 2-74 DMA接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_uart_dma_init	Y	Y	Y	Y	Y
app_uart_dma_deinit	Y	Y	Y	Y	Y
app_uart_dma_transmit_async	Y	Y	Y	Y	Y
app_uart_dma_receive_async	Y	Y	Y	Y	Y
app_uart_transmit_dma_sg_llp	N	Y	Y	N	N
app_uart_receive_dma_sg_llp	N	Y	Y	N	N

2.5.5.1 app_uart_dma_init

表 2-75 app_uart_dma_init接口

函数原型	uint16_t app_uart_dma_init(app_uart_params_t *p_params)
功能说明	初始化UART DMA传输
输入参数	p_params：初始化参数的结构体指针

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.5.2 app_uart_dma_deinit

表 2-76 app_uart_dma_deinit接口

函数原型	uint16_t app_uart_dma_deinit(app_uart_id_t id)
功能说明	反初始化UART DMA传输
输入参数	id：UART模块ID（见2.5.3.1 app_uart_id_t描述）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.5.3 app_uart_dma_transmit_async

表 2-77 app_uart_dma_transmit_async接口

函数原型	uint16_t app_uart_dma_transmit_async(app_uart_id_t id, uint8_t *p_data, uint16_t size)
功能说明	串口DMA方式（异步）发送数据
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：需要发送的数据，以Byte为单位 • size：需要发送的数据长度，此长度不应大于初始化的tx_buffer长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.5.4 app_uart_dma_receive_async

表 2-78 app_uart_dma_receive_async接口

函数原型	uint16_t app_uart_dma_receive_async(app_uart_id_t id, uint8_t *p_data, uint16_t size)
功能说明	串口DMA方式（异步）接收数据
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：存放接收到的数据，以Byte为单位 • size：设置需要接收的数据个数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.5.5 app_uart_transmit_dma_sg_llp

表 2-79 app_uart_transmit_dma_sg_llp接口

函数原型	uint16_t app_uart_transmit_dma_sg_llp(app_uart_id_t id, uint8_t *p_data, uint16_t size, dma_sg_llp_config_t *sg_llp_config)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	串口以DMA方式发送数据，支持scatter（分散）、gather（聚合）和llp（链表）特性，用于不连续内存之间的数据传输，以及使用链表传输提高单次传输的字节数
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：需要发送的数据 • size：需要发送的数据长度，以Byte为单位，此长度不应大于初始化的tx_buffer长度 • sg_llp_config：DMA分散聚合配置，详见2.4 APP DMA驱动
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.5.6 app_uart_receive_dma_sg_llp

表 2-80 app_uart_receive_dma_sg_llp接口

函数原型	uint16_t app_uart_receive_dma_sg_llp(app_uart_id_t id, uint8_t *p_data, uint16_t size, dma_sg_llp_config_t *sg_llp_config)
功能说明	串口以DMA方式接收数据，支持scatter（分散）、gather（聚合）和llp（链表）特性，用于不连续内存之间的数据传输，以及使用链表传输提高单次传输的字节数
输入参数	• id：UART模块ID（见2.5.3.1 app_uart_id_t描述） • p_data：存放接收到的数据 • size：设置需要接收的数据个数，以Byte为单位 • sg_llp_config：DMA分散聚合配置，详见2.4 APP DMA驱动
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.5.6 示例工程

2.5.6.1 UART Interrupt

示例工程实现UART以中断方式收发数据，工程路径：SDK_Folder\projects\peripheral\uart\app_uart。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.5.6.1.1 流程图

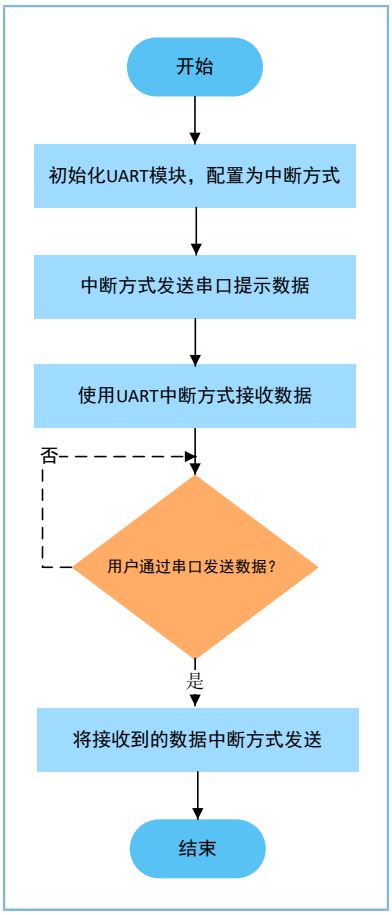

图 2-6 UART以中断方式收发数据流程

串口调试助手和开发板通过UART进行通信。开发板向串口发送提示数据，告知用户需要传输数据，开发板处于接收数据等待状态；当用户通过串口调试助手发送数据时，开发板接收发送的数据，并将其发送到串口进行显示。

2.5.6.1.2 初始化参数

初始化参数如下：

```
app_uart_params_t uart_params = {
    .id      = UART_ID,
    .pin_cfg = {
           .tx = {
              .type = APP_UART_TX_IO_TYPE,
              .mux  = APP_UART_TX_PINMUX,
              .pin  = APP_UART_TX_PIN,
              .pull = APP_IO_PULLUP,
           },
           .rx = {
              .type = APP_UART_RX_IO_TYPE,
              .mux  = APP_UART_RX_PINMUX,
              .pin  = APP_UART_RX_PIN,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
            .pull = APP_IO_PULLUP,
        },
    },

    .init = {
        .baud_rate = 115200,
        .data_bits = UART_DATABITS_8,
        .stop_bits = UART_STOPBITS_1,
        .parity    = UART_PARITY_NONE,
        .hw_flow_ctrl    = UART_HWCONTROL_NONE,
        .rx_timeout_mode = UART_RECEIVER_TIMEOUT_ENABLE,
    },
};
```

- id：定义串口ID

  APP_UART_ID可以为APP_UART_ID_0、APP_UART_ID_1、APP_UART_ID_2、APP_UART_ID_3、APP_UART_ID_4、APP_UART_ID_5，具体选择请参考*board_SK.h*，例如GR5526中APP_UART_ID为APP_UART_ID_3。

```
#define APP_UART_ID APP_UART_ID_3
```

- pin_cfg：配置串口读写引脚

  主要配置IO类型、复用MUX模式、使用的引脚和上下拉模式，具体参数定义请参考*board_SK.h*，以GR5526为例，定义如下，串口读写引脚分别为GPIO5和GPIO4，上拉模式。

```
/*******UART DRIVER IO CONFIG*******************/
#define APP_UART_TX_IO_TYPE APP_IO_TYPE_GPIOA
#define APP_UART_RX_IO_TYPE APP_IO_TYPE_GPIOA
#define APP_UART_TX_PIN     APP_IO_PIN_4
#define APP_UART_RX_PIN     APP_IO_PIN_5
#define APP_UART_TX_PINMUX          APP_IO_MUX_3
#define APP_UART_RX_PINMUX          APP_IO_MUX_3
#define APP_UART_TX_PULL            APP_IO_PULLUP
#define APP_UART_RX_PULL            APP_IO_PULLUP
```

- init：串口传输参数
  - init.baud_rate：波特率115200
  - init.data_bits：数据位UART_DATABITS_8（8 bits）
  - init.stop_bits：停止位UART_STOPBITS_1（1位）
  - init.parity：奇偶校验位UART_PARITY_NONE（无奇偶校验位）
  - init.hw_flow_ctrl：硬件流控使能位UART_HWCONTROL_NONE（无硬件流控）
  - init.rx_timeout_mode：接收超时使能位UART_RECEIVER_TIMEOUT_ENABLE（使能接收超时）

2.5.6.1.3 重要函数

- 初始化UART

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
ret = app_uart_init(&uart_params, app_uart_callback, &uart_buffer)
```

- uart_params：初始化参数结构体。
- app_uart_callback：用户自定义回调函数。
- uart_buffer：发送缓存。发送数据长度不能超过缓存大小，默认使用中断方式传输。

- 串口同步发送

```
app_uart_transmit_sync(UART_ID, g_message_0, sizeof(g_message_0), 5000)
```

串口中断方式同步发送数据。

- UART_ID：串口ID。
- g_message_0：待发送的数据。
- sizeof(g_message_0)：待发送数据的长度。
- 5000 ms：超时时间。

- 串口异步发送

```
app_uart_transmit_async(UART_ID, g_tx_buffer, rlen)
```

串口中断方式异步发送数据。

- UART_ID：串口ID。
- g_tx_buffer：待发送数据的buffer。
- rlen：待发送数据的长度。

- 串口异步接收

```
app_uart_receive_async(UART_ID, g_rx_buffer, sizeof(g_rx_buffer))
```

串口中断方式异步接收数据。

- UART_ID：串口ID。
- g_rx_buffer：待接收数据的buffer。
- sizeof(g_rx_buffer)：待接收数据的长度。

2.5.6.1.4 测试验证

1. 连接开发板到PC端，打开串口助手，连接串口。
2. 芯片上电串口日志打印提示信息，要求串口发送小于126字节的数据。
3. 当通过串口调试助手发送input data，开发板端串口接收到数据并将input data发送到串口，串口调试助手收到后进行显示。

```
Please input characters(<126) and end with newline.
Input:
input data
input data
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.5.6.2 UART DMA

示例工程实现UART以DMA方式收发数据，工程路径：SDK_Folder\projects\peripheral\uart\app_uart_dma。

2.5.6.2.1 流程图

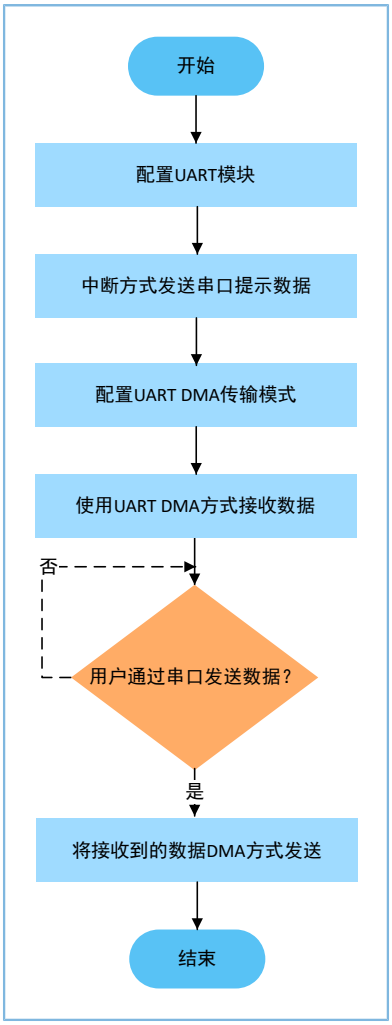

图 2-7 UART以DMA方式收发数据流程

串口调试助手和开发板通过UART进行通信。开发板向串口发送提示数据，告知用户需要传输数据，开发板处于接收数据等待状态；当用户通过串口调试助手发送数据时，开发板接收发送的数据，并将其发送到串口进行显示。

2.5.6.2.2 初始化参数

初始化参数如下：

```
app_uart_params_t uart_params = {
    .id      = UART_ID,
    .pin_cfg = {
           .tx = {
              .type = APP_UART_TX_IO_TYPE,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
            .mux  = APP_UART_TX_PINMUX,
            .pin  = APP_UART_TX_PIN,
            .pull = APP_IO_PULLUP,
        },
        .rx = {
            .type = APP_UART_RX_IO_TYPE,
            .mux  = APP_UART_RX_PINMUX,
            .pin  = APP_UART_RX_PIN,
            .pull = APP_IO_PULLUP,
        },
    },
    .dma_cfg = {
        .tx_dma_instance = DMA0,
        .rx_dma_instance = DMA0,
        .tx_dma_channel  = DMA_Channel0,
        .rx_dma_channel  = DMA_Channel1 ,
    },
    .init = {
        .baud_rate = 115200,
        .data_bits = UART_DATABITS_8,
        .stop_bits = UART_STOPBITS_1,
        .parity    = UART_PARITY_NONE,
        .hw_flow_ctrl    = UART_HWCONTROL_NONE,
        .rx_timeout_mode = UART_RECEIVER_TIMEOUT_ENABLE,
    },
};
```

- id：定义串口ID

```
#define UART_ID APP_UART_ID
```

APP_UART_ID可以为APP_UART_ID_0、APP_UART_ID_1、APP_UART_ID_2、APP_UART_ID_3、APP_UART_ID_4、APP_UART_ID_5，具体选择请参考*board_SK.h*，例如GR5526中APP_UART_ID为APP_UART_ID_3。

```
#define APP_UART_ID APP_UART_ID_3
```

- pin_cfg：配置串口读写引脚

主要配置IO类型、复用MUX模式、使用的引脚和上下拉模式，具体参数定义请参考*board_SK.h*，以GR5526为例，定义如下，串口读写引脚分别为GPIO5和GPIO4，上拉模式。

```
/*******UART DRIVER IO CONFIG*******************/
#define APP_UART_TX_IO_TYPE                 APP_IO_TYPE_GPIOA
#define APP_UART_RX_IO_TYPE                 APP_IO_TYPE_GPIOA
#define APP_UART_TX_PIN                     APP_IO_PIN_4
#define APP_UART_RX_PIN                     APP_IO_PIN_5
#define APP_UART_TX_PINMUX                  APP_IO_MUX_3
#define APP_UART_RX_PINMUX                  APP_IO_MUX_3
#define APP_UART_TX_PULL                    APP_IO_PULLUP
#define APP_UART_RX_PULL                    APP_IO_PULLUP
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init：串口传输参数
  - init.baud_rate：波特率115200
  - init.data_bits：数据位UART_DATABITS_8（8 bits）
  - init.stop_bits：停止位UART_STOPBITS_1（1位）
  - init.parity：奇偶校验位UART_PARITY_NONE（无奇偶校验位）
  - init.hw_flow_ctrl：硬件流控使能位UART_HWCONTROL_NONE（无硬件流控）
  - init.rx_timeout_mode：接收超时使能位UART_RECEIVER_TIMEOUT_ENABLE（使能接收超时）
- dma_cfg：DMA参数配置
  - dma_cfg.tx_dma_instance：串口发送使用的DMA实例，DMA0
  - dma_cfg.rx_dma_instance：串口接收使用的DMA实例，DMA0
  - dma_cfg.tx_dma_channel：串口发送使用的DMA通道，选用DMA_Channel0，即通道0
  - dma_cfg.rx_dma_channel：串口接收使用的DMA通道，选用DMA_Channel1，即通道1

2.5.6.2.3 重要函数

- 初始化UART

```
ret = app_uart_init(&uart_params, app_uart_callback, &uart_buffer)
```

  - uart_params：初始化参数结构体
  - app_uart_callback：用户自定义回调函数
  - uart_buffer：发送缓存，发送数据不超过缓存大小，默认中断方式传输。

- 配置串口DMA传输模式

```
ret = app_uart_dma_init(&uart_params);
```

- 串口同步发送

```
app_uart_transmit_sync(UART_ID, g_message_0, sizeof(g_message_0), 5000)
```

串口中断方式同步发送数据。

  - UART_ID：串口ID
  - g_message_0：待发送的数据
  - sizeof(g_message_0)：发送数据的长度
  - 5000 ms：超时时间

- 串口异步发送

```
app_uart_dma_transmit_async(UART_ID, g_tx_buffer, rlen)
```

串口DMA方式异步发送数据。

  - UART_ID：串口ID

版权所有 © 2025 深圳市汇顶科技股份有限公司

- g_tx_buffer：需要发送数据的buffer
- rlen：需要发送的数据长度

- 串口异步接收

```
app_uart_dma_receive_async(UART_ID, g_rx_buffer, sizeof(g_rx_buffer))
```

串口DMA方式异步接收数据。

- UART_ID：串口ID
- g_rx_buffer：需要接收数据的buffer
- sizeof(g_rx_buffer)：需要接收数据的长度

2.5.6.2.4 测试验证

同2.5.6.1.4 测试验证。

2.5.6.3 UART DMA（分散聚合特性）

示例工程实现带分散聚合特性的UART DMA方式收发数据，工程路径：`SDK_Folder\projects\peripheral\uart\app_uart_dma_sg_llp`。

说明:

该示例仅适用于GR5526和GR5x25。

2.5.6.3.1 流程图

对比示例2.5.6.1 UART Interrupt，需要配置DMA分散聚合结构体，用于带分散聚合特性的UART DMA方式收发数据。

版权所有 © 2025 深圳市汇顶科技股份有限公司

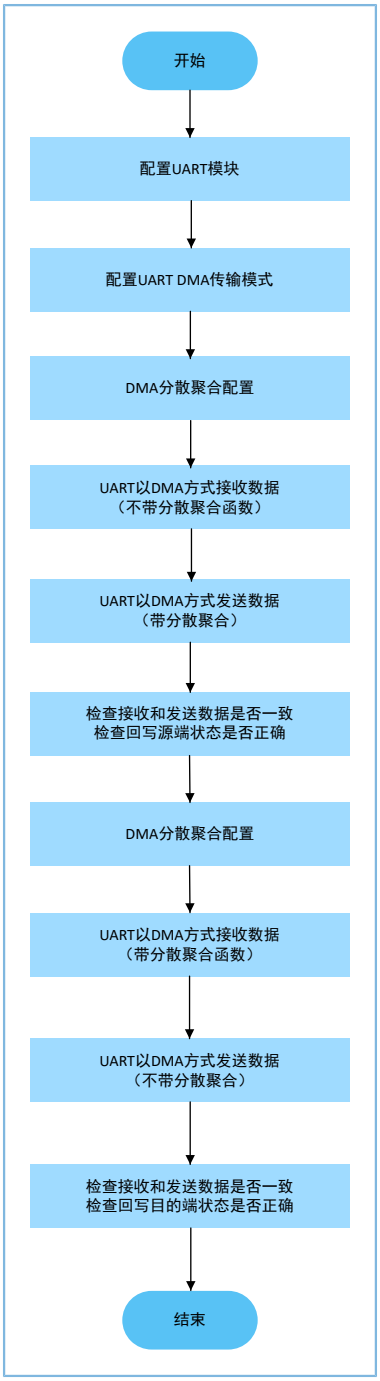

图 2-8 UART带分散聚合函数收发数据流程

2.5.6.3.2 初始化参数

初始化参数如下：

```
app_uart_params_t uart_params = {
    .id      = UART_ID,
    .pin_cfg = {
           .tx = {
              .type = APP_UART1_TX_IO_TYPE,
              .mux  = APP_UART1_TX_PINMUX,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
            .pin  = APP_UART1_TX_PIN,
            .pull = APP_IO_PULLUP,
        },
        .rx = {
            .type = APP_UART1_RX_IO_TYPE,
            .mux  = APP_UART1_RX_PINMUX,
            .pin  = APP_UART1_RX_PIN,
            .pull = APP_IO_PULLUP,
        },
    },
    .dma_cfg = {
        .tx_dma_instance = DMA0,
        .rx_dma_instance = DMA0,
        .tx_dma_channel  = DMA_Channel0,
        .rx_dma_channel  = DMA_Channel1 ,
    },
    .init = {
        .baud_rate = 115200,
        .data_bits = UART_DATABITS_8,
        .stop_bits = UART_STOPBITS_1,
        .parity    = UART_PARITY_NONE,
        .hw_flow_ctrl    = UART_HWCONTROL_NONE,
        .rx_timeout_mode = UART_RECEIVER_TIMEOUT_DISABLE,
    },
};
```

- id：定义串口ID

  具体选择请参考*board_SK.h*，例如GR5526中APP_UART_ID为APP_UART_ID_1。

```
#define APP_UART1_ID                        APP_UART_ID_1
```

- pin_cfg：配置串口读写引脚

  主要配置IO类型、复用MUX模式、使用的引脚和上下拉模式，具体参数定义请参考*board_SK.h*，以GR5526为例，定义如下，串口读写引脚分别为GPIO32和GPIO33，上拉模式，复用模式4。

```
#define APP_UART1_TX_IO_TYPE                APP_IO_TYPE_GPIOC
#define APP_UART1_RX_IO_TYPE                APP_IO_TYPE_GPIOC
#define APP_UART1_TX_PIN                    APP_IO_PIN_0
#define APP_UART1_RX_PIN                    APP_IO_PIN_1
#define APP_UART1_TX_PINMUX                 APP_IO_MUX_4
#define APP_UART1_RX_PINMUX                 APP_IO_MUX_4
```

- init：串口传输参数
  - init.baud_rate：波特率115200
  - init.data_bits：数据位UART_DATABITS_8（8 bits）
  - init.stop_bits：停止位UART_STOPBITS_1（1位）

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.parity：奇偶校验位UART_PARITY_NONE（无奇偶校验位）
- init.hw_flow_ctrl：硬件流控使能位UART_HWCONTROL_NONE（无硬件流控）
- init.rx_timeout_mode：接收超时使能位UART_RECEIVER_TIMEOUT_ENABLE（使能接收超时）

- dma_cfg：DMA参数配置
  - dma_cfg.tx_dma_instance：串口发送使用的DMA实例，DMA0
  - dma_cfg.rx_dma_instance：串口接收使用的DMA实例，DMA0
  - dma_cfg.tx_dma_channel：串口发送使用的DMA实例通道，选用DMA_Channel0，即通道0
  - dma_cfg.rx_dma_channel：串口接收使用的DMA实例通道，选用DMA_Channel1，即通道1

2.5.6.3.3 分散聚合结构体配置

```
dma_block_config_t block1 = {
    .src_address = (uint32_t)&src_data1,
    .dst_address = (uint32_t)0,
    .p_lli = NULL,
    .CTL_L = DMA_CTLL_INI_EN | DMA_DST_NO_CHANGE | DMA_SRC_INCREMENT |
                          DMA_SDATAALIGN_BYTE | DMA_DDATAALIGN_BYTE |\
                    DMA_SRC_GATHER_ENABLE | DMA_DST_SCATTER_DISABLE |\
                    DMA_LLP_SRC_ENABLE | DMA_LLP_DST_DISABLE | DMA_MEMORY_TO_PERIPH,
    .CTL_H = (uint32_t)UART_DATA_LEN/2,
    .src_status = 0x0,
    .dst_status = 0x0,
};

dma_block_config_t block0 = {
    .src_address = (uint32_t)&src_data0,
    .dst_address = (uint32_t)0,
    .p_lli = &block1,
    .CTL_L = DMA_CTLL_INI_EN | DMA_DST_NO_CHANGE | DMA_SRC_INCREMENT |
                          DMA_SDATAALIGN_BYTE | DMA_DDATAALIGN_BYTE |\
                    DMA_SRC_GATHER_ENABLE | DMA_DST_SCATTER_DISABLE |\
                    DMA_LLP_SRC_ENABLE | DMA_LLP_DST_DISABLE | DMA_MEMORY_TO_PERIPH,
    .CTL_H = (uint32_t)UART_DATA_LEN/2,
    .src_status = 0x0,
    .dst_status = 0x0,
};
dma_sg_llp_config_t sg_llp_config;
sg_llp_config.scatter_config.dst_scatter_en = DMA_DST_SCATTER_DISABLE;
sg_llp_config.scatter_config.dst_dsi = (uint32_t)0;
sg_llp_config.scatter_config.dst_dsc = (uint32_t)0;
sg_llp_config.gather_config.src_gather_en = DMA_SRC_GATHER_ENABLE;
sg_llp_config.gather_config.src_sgi = (uint32_t)1;
sg_llp_config.gather_config.src_sgc = (uint32_t)1;
sg_llp_config.llp_config.llp_src_en = DMA_LLP_SRC_ENABLE;
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
    sg_llp_config.llp_config.llp_src_writeback = 0xa;
    sg_llp_config.llp_config.llp_dst_en = DMA_LLP_DST_DISABLE;
    sg_llp_config.llp_config.head_lli = &block0;
```

- scatter_config.dst_scatter_en：选择DMA_DST_SCATTER_DISABLE，即目的端分散禁用
- scatter_config.dst_dsi：目的端分散间隔，以数据传输位宽为单位
- scatter_config.dst_dsc：目的端分散计数，代表连续的数据个数
- gather_config.src_gather_en：源端聚合使能
- gather_config.src_sgi：源端聚合间隔1个数据传输位宽
- gather_config.src_sgc：源端聚合计数1，代表连续的数据个数，数据单位和传输位宽有关。
- llp_config.llp_src_en：源端传输链表使能
- llp_config.llp_src_writeback：源端回写状态值
- llp_config.llp_dst_en：目的端传输链表使能
- llp_config.head_lli：传输链表头，指向block0
- src_address：源数据地址
- dst_address：目的数据地址
- p_lli：传输链表下一个节点地址
- CTL_L：控制寄存器低32位配置
- CTL_H：控制寄存器高32位配置
- src_status：回写源端传输状态
- dst_address：回写目的端传输状态
- DMA_CTLL_INI_EN：使能DMA中断
- DMA_DST_NO_CHANGE：目的端地址不变
- DMA_SRC_INCREMENT：源端地址递增
- DMA_SDATAALIGN_BYTE：源端数据传输位宽8 bits
- DMA_DDATAALIGN_BYTE：目的端数据传输位宽8 bits
- DMA_SRC_GATHER_ENABLE：源端聚合使能
- DMA_DST_SCATTER_DISABLE：目的端分散不使能
- DMA_LLP_SRC_ENABLE：源端传输链表使能
- DMA_LLP_DST_DISABLE：目的端传输链表不使能
- DMA_MEMORY_TO_PERIPH：数据传输方向为内存到外设

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.5.6.3.4 重要函数

- 串口以DMA分散聚合的方式接收数据

```
app_uart_receive_dma_sg_llp(APP_UART_ID_1, dst_data0,
                                    UART_DATA_LEN/2,&sg_llp_config)
```

- 串口以DMA分散聚合的方式发送数据

```
app_uart_transmit_dma_sg_llp(UART_ID, src_data0,
                                     UART_DATA_LEN/2,&sg_llp_config)
```

2.5.6.3.5 测试验证

1. 根据示例工程代码连接引脚。
2. 引脚连接后，连接开发板到PC端，打开串口调试助手，连接串口，查看日志对比接收数据和发送数据是否一致，回写状态是否正确。

```
[13:48:43.810] uart_transmit_dma_sg_llp transmit success
uart_transmit_dma_sg_llp data correct
uart_transmit_dma_sg_llp write back success
app_uart_receive_dma_sg_llp_test receive success
app_uart_receive_dma_sg_llp_test data0 is correct
app_uart_receive_dma_sg_llp_test data1 is correct
app_uart_receive_dma_sg_llp_test write back success
UART APP DMA SG & LLP example done!
```

2.6 APP TIM驱动

2.6.1 结构体

2.6.1.1 app_tim_params_t

结构体定义如下：

表 2-81 app_tim_params_t结构体成员

成员	描述	取值
app_tim_id_t id	Timer模块ID	该参数的取值可以是下列值中的任意一个： • APP_TIM_ID_0（TIMER0） • APP_TIM_ID_1（TIMER1）
timer_init_t init	Timer模块的初始化配置	参考2.6.1.2 timer_init_t结构体
tim_env_t tim_env	仅用于参数、句柄和信号量的保存，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_tim.h。

2.6.1.2 timer_init_t

结构体定义如下：

表 2-82 timer_init_t结构体成员

成员	描述	取值
uint32_t auto_reload	自动装载值，表示定时器的倒计时数。如果要配置特定的持续时间（例如，μs、ms或s），则需要使用以下公式： Expect_time (s) = RELOAD/(SystemCoreClock-1) 当自动装载值配置为0时，不会产生计数完成事件中断。其中RELOAD为自动装载值，即auto_reload，SystemCoreClock为系统时钟频率。	0 ～ 0xFFFFFFFF

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_tim.h
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_tim.h
- GR5405：SDK_Folder\hal_drv\include\hal_tim.h

2.6.2 枚举

2.6.2.1 app_tim_id_t

枚举定义如下：

表 2-83 app_tim_id_t枚举成员

成员	描述
APP_TIM_ID_0	TIMER模块0
APP_TIM_ID_1	TIMER模块1
APP_TIM_ID_MAX	TIMER模块总数，用作参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_tim.h。

2.6.2.2 app_tim_evt_t

枚举定义如下：

表 2-84 app_tim_evt_t枚举成员

成员	描述
APP_TIM_EVT_ERROR = 0	错误事件
APP_TIM_EVT_DONE = 1	计数完成事件
APP_TIM_EVT_CHANNEL0	CHANNEL0事件，仅适用于GR533x、GR5405
APP_TIM_EVT_CHANNEL1	CHANNEL1事件，仅适用于GR533x、GR5405
APP_TIM_EVT_CHANNEL2	CHANNEL2事件，仅适用于GR533x、GR5405
APP_TIM_EVT_CHANNEL3	CHANNEL3事件，仅适用于GR533x、GR5405

详细信息请参考文件：SDK_Folder\drivers\inc\app_tim.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.6.2.3 app_tim_state_t

枚举定义如下：

表 2-85 app_tim_state_t枚举成员

成员	描述
APP_TIM_INVALID = 0	TIMER不可用
APP_TIM_ACTIVITY = 1	TIMER激活
APP_TIM_SLEEP = 2	TIMER睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_tim.h。

2.6.3 接口函数

表 2-86 TIM驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_tim_init	Y	Y	Y	Y	Y
app_tim_deinit	Y	Y	Y	Y	Y
app_tim_start	Y	Y	Y	Y	Y
app_tim_stop	Y	Y	Y	Y	Y
app_tim_get_handle	Y	Y	Y	Y	Y
app_tim_get_channel0_val	N	N	N	Y	Y
app_tim_get_channel1_val	N	N	N	Y	Y
app_tim_get_channel2_val	N	N	N	Y	Y
app_tim_get_channel3_val	N	N	N	Y	Y

2.6.3.1 app_tim_init

表 2-87 app_tim_init接口

函数原型	uint16_t app_tim_init(app_tim_params_t *p_params, app_tim_evt_handler_t evt_handler)
功能说明	初始化Timer
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_tim_evt_handler_t)(app_tim_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.6.3.2 app_tim_deinit

表 2-88 app_tim_deinit接口

函数原型	uint16_t app_tim_deinit(app_tim_id_t id)
功能说明	反初始化Timer
输入参数	id：Timer模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.6.3.3 app_tim_start

表 2-89 app_tim_start接口

函数原型	uint16_t app_tim_start(app_tim_id_t id)
功能说明	开启Timer，使能中断并开始向下计数
输入参数	id：指定Timer模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.6.3.4 app_tim_stop

表 2-90 app_tim_stop接口

函数原型	uint16_t app_tim_stop(app_tim_id_t id)
功能说明	关闭Timer，停止计数
输入参数	id：指定Timer模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.6.3.5 app_tim_get_handle

表 2-91 app_tim_get_handle接口

函数原型	uint16_t app_tim_get_handle(app_tim_id_t id)
功能说明	获取Timer句柄
输入参数	id：指定Timer模块ID
返回值	Timer句柄，详见： • GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_tim.h • GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_tim.h • GR5405：SDK_Folder\hal_drv\include\hal_tim.h
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.6.3.6 app_tim_get_channel0_val

表 2-92 app_tim_get_channel0_val接口

函数原型	uint32_t app_tim_get_channel0_val(app_tim_id_t id)
功能说明	获取当前通道0的计数值
输入参数	id：Timer模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR533x、GR5405

2.6.3.7 app_tim_get_channel1_val

表 2-93 app_tim_get_channel1_val接口

函数原型	uint32_t app_tim_get_channel1_val(app_tim_id_t id)
功能说明	获取当前通道1的计数值
输入参数	id：Timer模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR533x、GR5405

2.6.3.8 app_tim_get_channel2_val

表 2-94 app_tim_get_channel2_val接口

函数原型	uint32_t app_tim_get_channel2_val(app_tim_id_t id)
功能说明	获取当前通道2的计数值
输入参数	id：Timer模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR533x、GR5405

2.6.3.9 app_tim_get_channel3_val

表 2-95 app_tim_get_channel3_val接口

函数原型	uint32_t app_tim_get_channel3_val(app_tim_id_t id)
功能说明	获取当前通道3的计数值
输入参数	id：Timer模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR533x、GR5405

2.6.4 示例工程

示例工程代码帮助开发者快速理解并使用TIMER驱动。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.6.4.1 TIMER

该示例工程实现TIMER的定时功能，工程路径：SDK_Folder\projects\peripheral\tim\app_tim。

2.6.4.1.1 流程图

示例设置TIMER0的reload值为SystemCoreClock - 1，因为TIMER0使用系统时钟，因此会看到每秒产生一次中断并打印。设置TIMER1的reload值为SystemCoreClok /100 - 1，TIMER1每10 ms产生中断，TIMER0和TIMER1计数将在10s后停止。

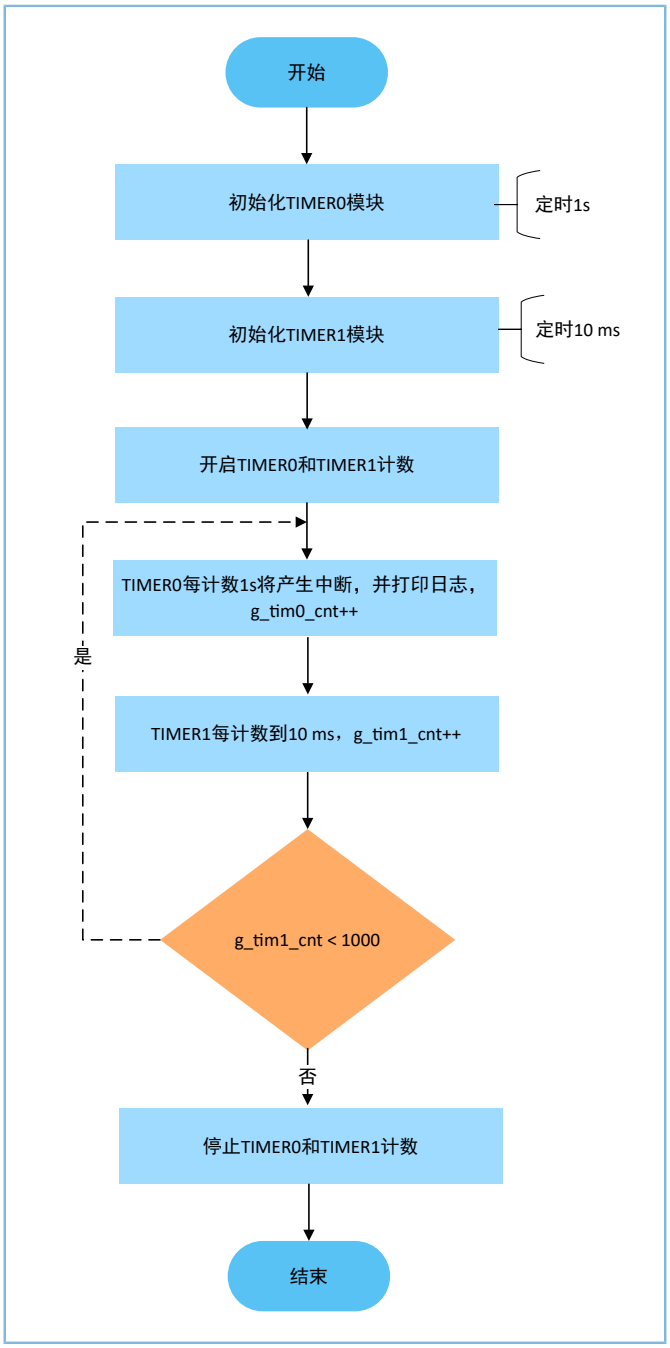

图 2-9 TIMER定时功能实现流程

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.6.4.1.2 初始化参数

```
    p_params_tim0.id = APP_TIM_ID_0;
    p_params_tim0.init.auto_reload = SystemCoreClock - 1;
```

- id：TIMER模块ID，选择APP_TIM_ID_0（TIMER0）
- auto_reload：自动装载值，设置为SystemCoreClock – 1，表示1s

2.6.4.1.3 重要函数

- 初始化模块

```
app_tim_init(&p_params_tim0, app_tim0_event_handler)
```

- 开启TIMER计数

```
app_tim_start(APP_TIM_ID_0)
```

- 停止TIMER计数

```
app_tim_stop(APP_TIM_ID_0)
```

2.6.4.1.4 测试验证

连接开发板到PC端，打开串口助手，连接串口，查看串口日志。TIMER0 1s打印一次串口，10s后停止打印。

```
This is 0th call TIMER0.

This is 1th call TIMER0.

This is 2th call TIMER0.

This is 3th call TIMER0.

This is 4th call TIMER0.

This is 5th call TIMER0.

This is 6th call TIMER0.

This is 7th call TIMER0.

This is 8th call TIMER0.

This is 9th call TIMER0.

This example demo end.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.7 APP DUAL_TIM驱动

2.7.1 结构体

2.7.1.1 app_dual_tim_params_t

结构体定义如下：

表 2-96 app_dual_tim_params_t结构体成员

成员	描述	取值
app_dual_tim_id_t id	使用的Dual Timer模块ID	该参数的取值可以是下列值中的任意一个： • APP_DUAL_TIM_ID_0（TIMER0） • APP_DUAL_TIM_ID_1（TIMER1）
dual_timer_init_t init	Dual Timer的初始化配置	参考2.7.1.2 dual_timer_init_t结构体
dual_tim_env_t dual_tim_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：

- GR551x/GR5526/GR5x25：`SDK_Folder\drivers\inc\hal\gr55xx_hal_dual_tim.h`
- GR533x：`SDK_Folder\drivers\inc\hal\gr533x_hal_dual_tim.h`
- GR5405：`SDK_Folder\hal_drv\include\hal_dual_tim.h`

2.7.1.2 dual_timer_init_t

结构体定义如下：

表 2-97 dual_timer_init_t结构体成员

成员	描述	取值
uint32_t prescaler	预分频系数	该参数的取值可以是下列值中的任意一个： • DUAL_TIMER_PRESCALER_DIV0（时钟除以1） • DUAL_TIMER_PRESCALER_DIV16（时钟除以16） • DUAL_TIMER_PRESCALER_DIV256（时钟除以256）
uint32_t counter_mode	计数模式	该参数的取值可以是下列值中的任意一个： • DUAL_TIMER_COUNTERMODE_LOOP（循环计数模式） • DUAL_TIMER_COUNTERMODE_ONESHOT（单次计数模式）
uint32_t auto_reload	自动装载值	自动装载值仅表示定时器的倒计时数。如果要配置特定的持续时间（例如，μs、ms或s），则需要使用以下公式： $$Expect\_time(s) = \frac{RELOAD}{\frac{SystemCoreClock}{PRE}} - 1$$

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		其中PRE代表预分频系数，SystemCoreClock为系统时钟频率，RELOAD指自动装载值，Expect_time代表换算的时间，单位：s，取值范围：0 ～ 0xFFFFFFFF。 当自动装载值配置为0时，不会产生计数完成事件中断。

详细信息请参考文件：SDK_Folder\drivers\inc\app_dual_tim.h。

2.7.2 枚举

2.7.2.1 app_dual_tim_id_t

枚举定义如下：

表 2-98 app_dual_tim_id_t枚举成员

成员	描述
APP_DUAL_TIM_ID_0 = 0	DUAL TIMER模块0
APP_DUAL_TIM_ID_1 = 1	DUAL TIMER模块1
APP_DUAL_TIM_ID_MAX = 2	DUAL TIMER模块总数，用作参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_dual_tim.h。

2.7.2.2 app_dual_tim_evt_t

枚举定义如下：

表 2-99 app_dual_tim_evt_t枚举成员

成员	描述
APP_DUAL_TIM_EVT_ERROR = 0	DUAL TIMER错误事件
APP_DUAL_TIM_EVT_DONE = 1	DUAL TIMER计数完成事件

详细信息请参考文件：SDK_Folder\drivers\inc\app_dual_tim.h。

2.7.2.3 app_dual_tim_state_t

枚举定义如下：

表 2-100 app_dual_tim_state_t枚举成员

成员	描述
APP_DUAL_TIM_INVALID = 0	DUAL TIMER不可用
APP_DUAL_TIM_ACTIVITY	DUAL TIMER激活
APP_DUAL_TIM_SLEEP	DUAL TIMER睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_dual_tim.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.7.3 接口函数

表 2-101 DUAL_TIMER驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_dual_tim_init	Y	Y	Y	Y	Y
app_dual_tim_deinit	Y	Y	Y	Y	Y
app_dual_tim_start	Y	Y	Y	Y	Y
app_dual_tim_stop	Y	Y	Y	Y	Y
app_dual_tim_set_params	Y	Y	Y	Y	Y
app_dual_tim_set_background_reload	Y	Y	Y	Y	Y
app_dual_tim_get_handle	Y	Y	Y	Y	Y
app_dual_tim_set_onetime_reload	N	N	N	Y	Y
app_dual_tim_io_crtl_config	N	N	N	Y	Y
app_dual_tim_set_period_count	N	N	N	Y	Y

2.7.3.1 app_dual_tim_init

表 2-102 app_dual_tim_init接口

函数原型	uint16_t app_dual_tim_init(app_dual_tim_params_t *p_params, app_dual_tim_evt_handler_t evt_handler)
功能说明	初始化DUAL TIMER
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_dual_tim_evt_handler_t)(app_dual_tim_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

2.7.3.2 app_dual_tim_deinit

表 2-103 app_dual_tim_deinit接口

函数原型	uint16_t app_dual_tim_deinit(app_dual_tim_id_t id)
功能说明	反初始化DUAL TIMER
输入参数	id：DUAL TIMER模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.7.3.3 app_dual_tim_start

表 2-104 app_dual_tim_start接口

函数原型	uint16_t app_dual_tim_start(app_dual_tim_id_t id)
功能说明	开启DUAL TIMER，使能中断并开始向下计数
输入参数	id：DUAL TIMER模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.7.3.4 app_dual_tim_stop

表 2-105 app_dual_tim_stop接口

函数原型	uint16_t app_dual_tim_stop(app_dual_tim_id_t id)
功能说明	关闭DUAL TIMER，停止计数
输入参数	id：DUAL TIMER模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.7.3.5 app_dual_tim_set_params

表 2-106 app_dual_tim_set_params接口

函数原型	uint16_t app_dual_tim_set_params(app_dual_tim_params_t *p_params, app_dual_tim_id_t id)
功能说明	运行过程中修改DUAL TIMER参数
输入参数	• id：DUAL TIMER模块ID • p_params：需要修改参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.7.3.6 app_dual_tim_set_background_reload

表 2-107 app_dual_tim_set_background_reload接口

函数原型	uint16_t app_dual_tim_set_background_reload(app_dual_tim_id_t id, uint32_t reload_value)
功能说明	设置自动装载值（下一周期生效）
输入参数	• id：DUAL TIMER模块ID • reload_value：自动装载值
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.7.3.7 app_dual_tim_get_handle

表 2-108 app_dual_tim_get_handle接口

函数原型	dual_timer_handle_t *app_dual_tim_get_handle(app_dual_tim_id_t id)
功能说明	获取DUAL TIMER句柄
输入参数	id：DUAL TIMER模块ID
返回值	Timer句柄，详见： • GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_dual_tim.h • GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_dual_tim.h • GR5405：SDK_Folder\hal_drv\include\hal_dual_tim.h
备注	

2.7.3.8 app_dual_tim_set_onetime_reload

表 2-109 app_dual_tim_set_onetime_reload接口

函数原型	uint16_t app_dual_tim_set_onetime_reload(app_dual_tim_id_t id, uint32_t reload_value)
功能说明	设置一次性装载值
输入参数	• id：Dual Timer模块ID • reload_value：自动装载值
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	• 一次性重载值仅在下一个周期生效；它比设置自动装载值（下一周期生效）具有更高的优先级。 • 仅适用于GR533x、GR5405

2.7.3.9 app_dual_tim_io_crtl_config

表 2-110 app_dual_tim_io_crtl_config接口

函数原型	uint16_t app_dual_tim_io_crtl_config(app_dual_tim_params_t *p_params, app_dual_tim_id_t id, app_dual_tim_io_crtl_params_t *io_crtl_params)
功能说明	配置Dual Timer I/O控制功能
输入参数	• p_params：初始化参数结构体指针 • id：DUAL TIMER模块ID • io_crtl_params：I/O控制参数（参考代码中的app_dual_tim_io_crtl_params_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	• 该函数调用必须在app_dual_tim_init之后 • 重新配置IO控制需要重新初始化Dual Timer • 仅适用于GR533x、GR5405

2.7.3.10 app_dual_tim_set_period_count

表 2-111 app_dual_tim_set_period_count接口

函数原型	uint16_t app_dual_tim_set_period_count(app_dual_tim_id_t id, uint32_t count_value)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	设置周期数功能
输入参数	• id：DUAL TIMER模块ID • count_value：周期数值（需大于1）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	• 该函数仅在带I/O控制的周期/循环模式下可用；函数调用必须在app_dual_tim_io_crtl_config之后 • 该函数将强制开启停止事件中断 • 如果当前计数值不是初始的周期值，可能会导致第一个周期缩短 • 仅适用于GR533x、GR5405

2.7.4 示例工程

2.7.4.1 Dual Timer

示例工程运用Dual Timer以中断方式实现定时打印程序中相关信息。Dual Timer0处于循环模式，第一个周期为1s，随后每5s产生一次中断，串口打印每次中断；Dual Timer1处于一次性模式，中断周期每次延长1s，中断程序将在15s停止，Dual Timer0和Dual Timer1停止计时。工程目录：SDK_Folder\projects\peripheral\dual_tim\app_dual_tim。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.7.4.1.1 流程图

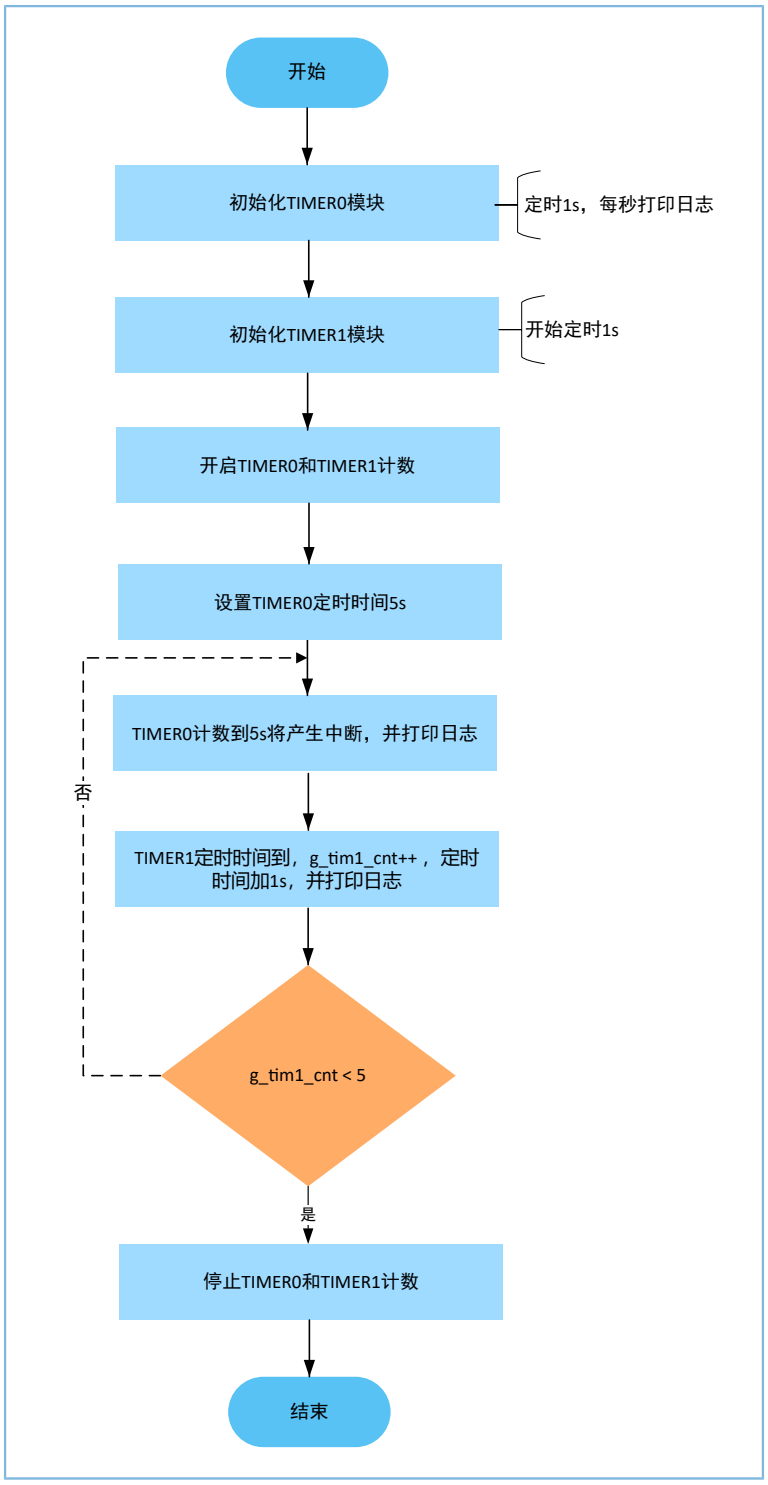

图 2-10 DUAL TIMER定时功能实现流程

2.7.4.1.2 初始化参数

以Dual Timer0为例，Dual Timer1类似。

```
#define DEFAULT_AUTO_RELOAD  63999999
#define TIMER_MS(X) (SystemCoreClock / 1000 *(X) - 1)
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
app_dual_tim_params_t p_params_tim0 = {
    .id = APP_DUAL_TIM_ID_0,
    .init = {
        .prescaler    = DUAL_TIMER_PRESCALER_DIV0,
        .counter_mode = DUAL_TIMER_COUNTERMODE_LOOP,
        .auto_reload  = DEFAULT_AUTO_RELOAD,
    },
};
```

- id：Dual Timer0模块ID选择APP_DUAL_TIM_ID_0
- init.prescaler：预分频系数DUAL_TIMER_PRESCALER_DIV0，时钟除以1
- init.counter_mode：采用循环模式DUAL_TIMER_COUNTERMODE_LOOP
- init.auto_reload：自动装载值DEFAULT_AUTO_RELOAD，默认63999999

2.7.4.1.3 重要函数

- 初始化Dual Timer函数

```
app_dual_tim_init(&p_params_tim0, app_dual_tim0_event_handler)
```

- 开启计数

```
app_dual_tim_start(APP_DUAL_TIM_ID_0)
```

- 设置自动装载值

```
app_dual_tim_set_background_reload(APP_DUAL_TIM_ID_0, TIMER_MS(5000))
```

- 重新设置计数参数

```
app_dual_tim_set_params(&p_params_tim1, APP_DUAL_TIM_ID_1)
```

- 停止计数

```
app_dual_tim_stop(APP_DUAL_TIM_ID_0)
```

2.7.4.1.4 测试验证

连接开发板到PC端，打开串口助手，连接串口，查看串口日志。DUAL TIMER0 在1s时打印日志，然后每5s打印一次日志，DUAL TIMER1 在1s、3s、6s、10s和15s时打印日志，DUAL TIMER0 和DUAL TIMER1在15s停止计数。

```
DUAL TIMER0 and DUAL TIMER1 start

This is 0th call DUAL TIMER0.

  This is 0th call DUAL TIMER1.

  This is 1th call DUAL TIMER1.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
This is 1th call DUAL TIMER0.

  This is 2th call DUAL TIMER1.

  This is 3th call DUAL TIMER1.

This is 2th call DUAL TIMER0.

  This is 4th call DUAL TIMER1.

This example demo end.
```

2.8 APP RTC驱动

2.8.1 宏定义

2.8.1.1 CFG_LPCLK_INTERNAL_EN

表 2-112 CFG_LPCLK_INTERNAL_EN宏定义

描述	值
是否使用芯片内部低速时钟源（RNG2时钟）	• 0：不使能，默认使用外部低速时钟源 • 1：使能

如果使能芯片内部低速时钟源，需要检查SDK_Folder\build\config\custom_config.h文件中PMU_CALIBRATION_ENABLE宏是否使能，不使能的情况下时钟源可能不准。

详细信息请参考文件：SDK_Folder\build\config\custom_config.h。

2.8.1.2 APP_RTC_ALARM_DISABLE_DATE

表 2-113 APP_RTC_ALARM_DISABLE_DATE宏定义

描述	值
不使能RTC的时间定时	CALENDAR_ALARM_DISABLE_DATE

详细信息请参考文件：SDK_Folder\drivers\inc\app_rtc.h。

2.8.1.3 APP_RTC_ALARM_DISABLE_TICK

表 2-114 APP_RTC_ALARM_DISABLE_TICK宏定义

描述	值
不使能RTC的TICK定时	CALENDAR_ALARM_DISABLE_TICK

详细信息请参考文件： SDK_Folder\drivers\inc\app_rtc.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.8.1.4 APP_RTC_ALARM_DISABLE_ALL

表 2-115 APP_RTC_ALARM_DISABLE_ALL宏定义

描述	值
不使能RTC的TICK定时和时间定时	CALENDAR_ALARM_DISABLE_ALL

详细信息请参考文件： SDK_Folder\drivers\inc\app_rtc.h。

2.8.2 结构体

2.8.2.1 app_rtc_time_t

结构体定义如下：

表 2-116 app_rtc_time_t结构体成员

成员	描述	取值
uint8_t sec	Calendar日历秒	取值范围：0 ～ 59
uint8_t min	Calendar日历分钟	取值范围：0 ～ 59
uint8_t hour	Calendar日历小时	取值范围：0 ～ 23
uint8_t date	Calendar日历日期	取值范围：1 ～ 31
uint8_t mon	Calendar日历月份	取值范围：1 ～ 12
uint8_t year	Calendar日历年	取值范围：0 ～ 99
uint8_t week	Calendar日历周	取值范围：0 ～ 6
uint16_t ms	Calendar日历毫秒	取值范围：0 ～ 999

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_calendar.h
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_calendar.h
- GR5405：SDK_Folder\hal_drv\include\hal_calendar.h

2.8.2.2 app_rtc_alarm_t

结构体定义如下：

表 2-117 app_rtc_alarm_t结构体成员

成员	描述	取值
uint8_t min	Calendar alarm分钟	取值范围：0 ～ 59
uint8_t hour	Calendar alarm小时	取值范围：0 ～ 23
uint8_t alarm_sel	Calendar alarm按周报警还是按月报警	取值范围： • CALENDAR_ALARM_SEL_DATE（按月为周期报警） • CALENDAR_ALARM_SEL_WEEKDAY（按周为周期报警）

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
uint16_t alarm_date_week_mask	Calendar alarm具体值，依赖于alarm_sel的值	• 如果alarm_sel=CALENDAR_ALARM_SEL_DATE： 取值范围：1 ～ 31 • 如果alarm_sel=CALENDAR_ALARM_SEL_WEEKDAY 取值为下面值的或操作： ◦ CALENDAR_ALARM_WEEKDAY_SUN ◦ CALENDAR_ALARM_WEEKDAY_MON ◦ CALENDAR_ALARM_WEEKDAY_TUE ◦ CALENDAR_ALARM_WEEKDAY_WED ◦ CALENDAR_ALARM_WEEKDAY_THU ◦ CALENDAR_ALARM_WEEKDAY_FRI ◦ CALENDAR_ALARM_WEEKDAY_SAT 例如选择星期一和星期二进行报警，设置参数如下： CALENDAR_ALARM_WEEKDAY_MON|CALENDAR_ALARM_WEEKDAY_TUE

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_calendar.h
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_calendar.h
- GR5405：SDK_Folder\hal_drv\include\hal_calendar.h

2.8.2.3 app_rtc_evt_t

结构体定义如下：

表 2-118 app_rtc_evt_t结构体成员

成员	描述	取值
app_rtc_evt_type_t type	RTC事件类型	该参数的取值可以是下列值中的任意一个： • APP_RTC_EVT_DATE_ALARM （RTC ALARM中断） • APP_RTC_EVT_TICK_ALARM （RTC TICK中断） • APP_RTC_EVT_OVERFLOW（RTC溢出中断）

详细信息请参考文件：SDK_Folder\drivers\inc\app_rtc.h。

2.8.3 枚举

2.8.3.1 app_rtc_evt_type_t

枚举定义如下：

表 2-119 app_rtc_evt_type_t枚举成员

成员	描述
APP_RTC_EVT_DATE_ALARM = 0	RTC ALARM中断

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述
APP_RTC_EVT_TICK_ALARM = 1	RTC TICK中断
APP_RTC_EVT_OVERFLOW = 2	RTC溢出中断

详细信息请参考文件：SDK_Folder\drivers\inc\app_rtc.h。

2.8.3.2 app_rtc_state_t

枚举定义如下：

表 2-120 app_rtc_state_t枚举成员

成员	描述
APP_RTC_INVALID = 0	RTC不可用
APP_RTC_ACTIVITY = 1	RTC激活

详细信息请参考文件：SDK_Folder\drivers\src\app_rtc.c。

2.8.4 接口函数

表 2-121 RTC驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_rtc_init	Y	Y	Y	Y	Y
app_rtc_deinit	Y	Y	Y	Y	Y
app_rtc_init_time	Y	Y	Y	Y	Y
app_rtc_get_time	Y	Y	Y	Y	Y
app_rtc_setup_alarm	Y	Y	Y	Y	Y
app_rtc_setup_tick	Y	Y	Y	Y	Y
app_rtc_disable_event	Y	Y	Y	Y	Y
app_rtc_time_sync	N	Y	Y	Y	Y

2.8.4.1 app_rtc_init

表 2-122 app_rtc_init接口

函数原型	uint16_t app_rtc_init(app_rtc_evt_handler_t evt_handler)
功能说明	初始化RTC
输入参数	evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_rtc_evt_handler_t)(app_rtc_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

版权所有 © 2025 深圳市汇顶科技股份有限公司

	使用芯片内部低速时钟源时，该接口无意义。

2.8.4.2 app_rtc_deinit

表 2-123 app_rtc_deinit接口

函数原型	uint16_t app_rtc_deinit(void)
功能说明	反初始化RTC
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	使用芯片内部低速时钟源，该接口无意义

2.8.4.3 app_rtc_init_time

表 2-124 app_rtc_init_time接口

函数原型	uint16_t app_rtc_init_time(app_rtc_time_t *p_time)
功能说明	初始化RTC时间
输入参数	p_time：初始时间结构体
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.8.4.4 app_rtc_get_time

表 2-125 app_rtc_get_time接口

函数原型	uint16_t app_rtc_get_time(app_rtc_time_t *p_time)
功能说明	获取RTC时间
输入参数	p_time：存放时间的结构体
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.8.4.5 app_rtc_setup_alarm

表 2-126 app_rtc_setup_alarm接口

函数原型	uint16_t app_rtc_setup_alarm(app_rtc_alarm_t *p_alarm)
功能说明	设置RTC alarm中断
输入参数	p_alarm：Alarm中断的配置参数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	使用芯片内部低速时钟源，该接口无意义

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.8.4.6 app_rtc_setup_tick

表 2-127 app_rtc_setup_tick接口

函数原型	uint16_t app_rtc_setup_tick(uint32_t interval)
功能说明	设置RTC TICK中断
输入参数	interval：RTC TICK值（单位：ms）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	使用芯片内部低速时钟源，该接口无意义

2.8.4.7 app_rtc_disable_event

表 2-128 app_rtc_disable_event接口

函数原型	uint16_t app_rtc_disable_event(uint32_t disable_mode)
功能说明	禁用RTC中断
输入参数	disable_mode：可取值APP_RTC_ALARM_DISABLE_DATE、APP_RTC_ALARM_DISABLE_TICK、APP_RTC_ALARM_DISABLE_ALL
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.8.4.8 app_rtc_time_sync

表 2-129 app_rtc_time_sync接口

函数原型	void app_rtc_time_sync(uint16_t SlowClockFreq)
功能说明	同步RTC时间
输入参数	SlowClockFreq：校准后的慢速时钟
返回值	
备注	适用于GR5526、GR5x25、GR533x和GR5405

2.8.5 示例工程

2.8.5.1 RTC

示例工程通过RTC实现闹钟功能，工程目录：SDK_Folder\projects\peripheral\rtc\app_rtc。

2.8.5.1.1 流程图

示例工程首先初始化RTC并注册回调函数，设置RTC时间，RTC开始计时，并通过接口获取当前的时间并打印，验证RTC已经正常计时。其次，设置闹钟，并启动RTC TICK中断，每10tick产生一次中断，每1000 tick打印日志，与此同时当到达闹钟定时时间8:01时，产生闹钟定时中断，并输出相应日志。

版权所有 © 2025 深圳市汇顶科技股份有限公司

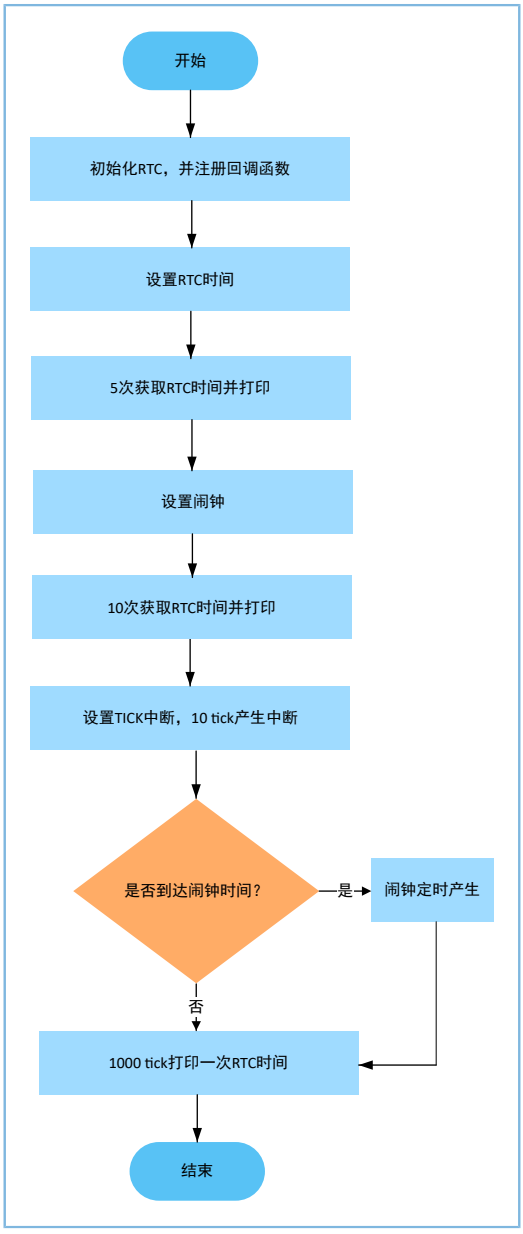

图 2-11 RTC示例工程流程

2.8.5.1.2 初始化参数

1. 初始日期设置：2019年5月20日8点0分0秒

```
app_rtc_time_t time;

time.year = 19;
time.mon  = 5;
time.date = 20;
time.hour = 8;
time.min  = 0;
time.sec  = 0;
time.week = 0;
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
app_rtc_init_time(&time);
```

2. 设置alarm为每周的上午8点01分。此函数必须在每次设置系统时间后重新调用。代码如下：

```
app_rtc_alarm_t alarm;

alarm.alarm_sel = CALENDAR_ALARM_SEL_WEEKDAY;
alarm.alarm_date_week_mask = 0xFF;
alarm.hour = 8;
alarm.min  = 1;
app_rtc_setup_alarm(&alarm)
```

- alarm_sel：选择按周或月报警，此处选择CALENDAR_ALARM_SEL_WEEKDAY（按周报警）。
- alarm_date_week_mask：每周具体哪天或者几天报警，0xFF表示每周每一天都报警。

2.8.5.1.3 重要函数

- 初始化RTC并注册回调函数

```
app_rtc_init(app_rtc_evt_handler)
```

- 设置初始时间

```
app_rtc_init_time(&time)
```

- 获取当前时间

```
app_rtc_get_time(&time)
```

- 设置闹钟

```
app_rtc_setup_alarm(&alarm)
```

- 设置RTC TICK中断

```
app_rtc_setup_tick(10)
```

每10 tick产生一次中断

2.8.5.1.4 测试验证

连接开发板到PC端，打开串口助手，连接串口，查看串口日志。App current time为获取当前时间并打印的日志，Tick alarm为RTC TICK中断打印的日志，设置闹钟8：01 am，在8:01 am时中断打印Date alarm。

每秒串口打印当前时间，并设置date alarm为8:01am：

```
[18:02:59.277] App current time: 05.20.19 Mon 08:00:01

[18:03:00.281] App current time: 05.20.19 Mon 08:00:02

[18:03:01.286] App current time: 05.20.19 Mon 08:00:03

[18:03:02.290] App current time: 05.20.19 Mon 08:00:04
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
[18:03:03.294] App current time: 05.20.19 Mon 08:00:05
Set an date alarm every day at 8.01 am.
```

设置tick定时，每计数1000 tick，打印一次当前时间：

```
Set an tick alarm every 10 ms.

[18:03:16.362] Tick alarm, 05.20.19 Mon 08:00:18, 100.

[18:03:17.366] Tick alarm, 05.20.19 Mon 08:00:19, 200.

[18:03:18.368] Tick alarm, 05.20.19 Mon 08:00:20, 300.

[18:03:19.373] Tick alarm, 05.20.19 Mon 08:00:21, 400.
```

定时闹钟时间到：

```
[18:03:57.519] Tick alarm, 05.20.19 Mon 08:00:59, 4200.

[18:03:58.331] Date alarm.

[18:03:58.503] Tick alarm, 05.20.19 Mon 08:01:00, 4300.
```

2.9 APP AON_WDT驱动

2.9.1 结构体

2.9.1.1 app_aon_wdt_params_t

结构体定义如下：

表 2-130 app_aon_wdt_params_t结构体成员

成员	描述	取值
aon_wdt_init_t init	AON_WDT需要的初始化配置	参考2.9.1.2 aon_wdt_init_t结构体
aon_aon_wdt_env_t aon_aon_wdt_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_aon_wdt.h_
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_aon_wdt.h
- GR5405：SDK_Folder\hal_drv\include\hal_aon_wdt.h

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.9.1.2 aon_wdt_init_t

表 2-131 aon_wdt_init_t结构体成员

成员	描述	取值
uint32_t counter	AON_WDT计数值。当计数值递减到0时，会复位系统	• GR551x：0x0U ～ 0xFFFFFFFF 说明： 计数值与ms的换算关系counter / (32768 / 1000) • GR5526/GR5x25/GR533x/GR5405：0 ～ 0xFFFFFFFFU/ (SystemRngClock / 1000)，单位：ms 说明： SystemRngClock为低速时钟频率（32.768 kHz），该时钟30s校准一次
uint32_t alarm_counter	闹钟值。当计数值递减到0时，会产生中断	• GR551x：无意义 • GR5526/GR5x25/GR533x/GR5405：0 ～ 0xFFFFFFFFU/ (SystemRngClock / 1000)，单位：ms 说明： SystemRngClock为低速时钟频率（32.768 kHz），该时钟30s校准一次

详细信息请参考文件：

- GR551x/GR5526/GR5x25：`SDK_Folder\drivers\inc\hal\gr55xx_hal_aon_wdt.h_`
- GR533x：`SDK_Folder\drivers\inc\hal\gr533x_hal_aon_wdt.h`
- GR5405：`SDK_Folder\hal_drv\include\hal_aon_wdt.h`

2.9.2 枚举

2.9.2.1 app_aon_wdt_state_t

枚举定义如下：

表 2-132 app_aon_wdt_state_t枚举成员

成员	描述
APP_AON_WDT_INVALID = 0	AON WDT不可用
APP_AON_WDT_ACTIVITY = 1	AON WDT激活

详细信息请参考文件：`SDK_Folder\drivers\inc\app_aon_wdt.h`。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.9.3 接口函数

表 2-133 AON_WDT驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_aon_wdt_init	Y	Y	Y	Y	Y
app_aon_wdt_deinit	Y	Y	Y	Y	Y
app_aon_wdt_refresh	Y	Y	Y	Y	Y

2.9.3.1 app_aon_wdt_init

表 2-134 app_aon_wdt_init接口

函数原型	uint16_t app_aon_wdt_init(app_aon_wdt_params_t *p_params, app_aon_wdt_evt_handler_t evt_handler)
功能说明	初始化AON WDT
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_aon_wdt_evt_handler_t)(void)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

2.9.3.2 app_aon_wdt_deinit

表 2-135 app_aon_wdt_deinit接口

函数原型	uint16_t app_aon_wdt_deinit(void)
功能说明	反初始化AON WDT
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.9.3.3 app_aon_wdt_refresh

表 2-136 app_aon_wdt_refresh接口

函数原型	uint16_t app_aon_wdt_refresh(void)
功能说明	AON WDT喂狗，重新装载计数值
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.9.4 示例工程

2.9.4.1 WDT

示例演示了WDT喂狗功能，工程路径：SDK_Folder\projects\peripheral\aon_wdt\app_aon_wdt_reset。

2.9.4.1.1 流程图

首先初始化AON WDT模块，设置喂狗时间最大间隔2s内，alarm为1s，当1s未喂狗，将产生中断进行提示喂狗，2s未喂狗，将复位系统。示例工程每300 ms喂狗一次，喂狗10次。喂完后，1s将产生中断，2s将复位系统。

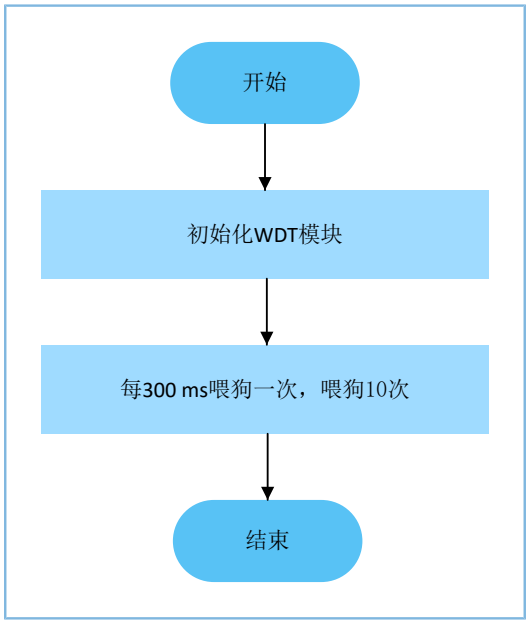

图 2-12 WDT喂狗功能实现流程

2.9.4.1.2 初始化参数

```
app_aon_wdt_params_t s_params;
#if (APP_DRIVER_CHIP_TYPE == APP_DRIVER_GR551X)
    s_params.init.counter = 32768 * 2 - 1;
    s_params.init.alarm_counter = 0x1F;
#else
    s_params.init.counter = 2000;
    s_params.init.alarm_counter = 1000;
#endif
    app_aon_wdt_init(&s_params, app_aon_wdt_evt_handler);
```

- counter：2s不进行喂狗，系统将复位。
- alarm_counter：闹钟值，当计数值递减到0时，产生中断。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.9.4.1.3 重要函数

- 初始化AON WDT

```
app_aon_wdt_init(&s_params, app_aon_wdt_evt_handler)
```

s_params：初始化参数结构体；app_aon_wdt_evt_handler：回调函数。

- 喂狗

```
app_aon_wdt_refresh()
```

Counter重新计数。

2.9.4.1.4 测试验证

连接开发板到PC端，打开串口助手，连接串口，查看串口日志。可以看到10次正常喂狗，10次以后未进行正常喂狗开始复位系统。

```
[15:37:17.554]
 0th feed dog.

[15:37:17.856]
 1th feed dog.

[15:37:18.159]
 2th feed dog.

[15:37:18.460]
 3th feed dog.

[15:37:18.763]
 4th feed dog.

[15:37:19.065]
 5th feed dog.

[15:37:19.368]
 6th feed dog.

[15:37:19.670]
 7th feed dog.

[15:37:19.971]
 8th feed dog.

[15:37:20.274]
 9th feed dog.

System will reset.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10 APP I2C驱动

2.10.1 结构体

2.10.1.1 app_i2c_pin_t

结构体定义如下：

表 2-137 app_i2c_pin_t结构体成员

成员	描述	取值
app_io_type_t type	IO类型	同2.2.1.2 app_io_evt_t结构体中描述
app_io_mux_t mux	IO映射功能	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
uint32_t pin	IO引脚	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
app_io_pull_t pull	IO上拉/下拉/浮空	同2.2.1.1 app_io_init_t结构体中描述

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.1.2 app_i2c_pin_cfg_t

结构体定义如下：

表 2-138 app_i2c_pin_cfg_t结构体成员

成员	描述	取值
app_i2c_pin_t scl	I2C SCL引脚配置	参考2.10.1.1 app_i2c_pin_t结构体
app_i2c_pin_t sda	I2C SDA引脚配置	参考2.10.1.1 app_i2c_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.1.3 app_i2c_dma_cfg_t

结构体定义如下：

表 2-139 app_i2c_dma_cfg_t结构体成员

成员	描述	取值
dma_reg_t *tx_dma_instance	I2C TX指定DMA模块配置	参考2.4.1.1 app_dma_params_t 章节中，DMA支持的外设
dma_reg_t *rx_dma_instance	I2C RX指定DMA模块配置	参考2.4.1.1 app_dma_params_t 章节中，DMA支持的外设
dma_channel_t tx_dma_channel	I2C TX的DMA通道选择	同2.4.1.1 app_dma_params_t中通道描述
dma_channel_t rx_dma_channel	I2C RX的DMA通道选择	同2.4.1.1 app_dma_params_t中通道描述

版权所有 © 2025 深圳市汇顶科技股份有限公司

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.1.4 app_i2c_params_t

结构体定义如下：

表 2-140 app_i2c_params_t结构体成员

成员	描述	取值
app_i2c_id_t id	I2C模块ID	• APP_I2C_ID_0（I2C 0） • APP_I2C_ID_1（I2C 1） • APP_I2C_ID_2（I2C 2） • APP_I2C_ID_3（I2C 3） • APP_I2C_ID_4（I2C 4） • APP_I2C_ID_5（I2C 5）
app_i2c_role_t role	I2C工作方式	该参数的取值可以是下列值中的任意一个： • APP_I2C_ROLE_MASTER（I2C配置为Master） • APP_I2C_ROLE_SLAVE（I2C配置为Slave）
app_i2c_pin_cfg_t pin_cfg	I2C引脚功能	参考2.10.1.2 app_i2c_pin_cfg_t结构体
app_i2c_dma_cfg_t dma_cfg	I2C操作模式	参考2.10.1.3 app_i2c_dma_cfg_t结构体
i2c_init_t init	I2C初始化参数配置	参考2.10.1.5 i2c_init_t结构体
i2c_env_t i2c_dev	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.1.5 i2c_init_t

结构体定义如下：

表 2-141 i2c_init_t结构体成员

成员	描述	取值
uint32_t speed	I2C传输速率	该参数的取值可以是下列值中的任意一个： • I2C_SPEED_100K（100 kHz, standard mode） • I2C_SPEED_400K（400 kHz, fast mode） • I2C_SPEED_1000K（1000 kHz, fast plus mode） • I2C_SPEED_2000K（2000 kHz, high-speed mode）
uint32_t own_address	I2C地址	作为Master主设备时，该参数无意义；作为Slave从设备时，由用户指定配置
uint32_t addressing_mode	I2C地址模式	该参数的取值可以是下列值中的任意一个： • I2C_ADDRESSINGMODE_7BIT（7 bits地址） • I2C_ADDRESSINGMODE_10BIT（10 bits地址）
uint32_t general_call_mode	通用广播模式	该参数的取值可以是下列值中的任意一个：

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• I2C_GENERALCALL_DISABLE（禁用） • I2C_GENERALCALL_ENABLE（使能）
uint32_t tx_hold_time	TX SDA保持时间，单位：ns	
uint32_t rx_hold_time	RX SDA保持时间，单位：ns	

2.10.1.6 app_i2c_evt_t

结构体定义如下：

表 2-142 app_i2c_evt_t结构体成员

成员	描述	取值
app_i2c_evt_type_t type	I2C事件类型	该参数的取值可以是下列值中的任意一个： • APP_I2C_EVT_ERROR（I2C传输错误） • APP_I2C_EVT_TX_CPLT（I2C发送完成） • APP_I2C_EVT_RX_DATA（I2C接收完成） • APP_I2C_ABORT（I2C传输中止）
union { uint32_t error_code; uint16_t size; }data;	• uint32_t error_code：I2C事件返回的错误号 • uint16_t size：I2C已接收/发送的数据长度 **说明：** 若事件类型为传输错误，此处返回错误号；若事件类型为接收/发送完成，此处显示已接收/发送的数据长度	I2C错误号： • HAL_I2C_ERROR_NONE（无错误） • HAL_I2C_ERROR_INVALID_PARAM（无效参数） • HAL_I2C_ERROR_ARB_LOST（仲裁丢失） • HAL_I2C_ERROR_NOACK（无回应） • HAL_I2C_ERROR_OVER（RX溢出） • HAL_I2C_ERROR_SCL_STUCK_AT_LOW（SCL被强制拉低） • HAL_I2C_ERROR_SDA_STUCK_AT_LOW（SDA被强制拉低） • HAL_I2C_ERROR_DMA（DMA错误） • HAL_I2C_ERROR_TIMEOUT（超时）
uint16_t slave_addr	I2C从设备地址	

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.2 枚举

2.10.2.1 app_i2c_id_t

枚举定义如下：

表 2-143 app_i2c_id_t枚举成员

成员	描述
APP_I2C_ID_0 = 0	I2C模块0
APP_I2C_ID_1 = 1	I2C模块1

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述
APP_I2C_ID_2 = 2	I2C模块2，仅适用于GR5526和GR5x25
APP_I2C_ID_3 = 3	I2C模块3，仅适用于GR5526和GR5x25
APP_I2C_ID_4 = 4	I2C模块4，仅适用于GR5526
APP_I2C_ID_5 = 5	I2C模块5，仅适用于GR5526
APP_I2C_ID_MAX = 6	I2C模块总个数，用于做参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.2.2 app_i2c_role_t

枚举定义如下：

表 2-144 app_i2c_role_t枚举成员

成员	描述
APP_I2C_ROLE_MASTER = 0	I2C作为Master
APP_I2C_ROLE_SLAVE = 1	I2C作为Slave
APP_I2C_ROLE_MAX = 2	I2C ROLE总个数，用于做参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.2.3 app_i2c_evt_type_t

枚举定义如下：

表 2-145 app_i2c_evt_type_t枚举成员

成员	描述
APP_I2C_EVT_ERROR = 0	I2C传输错误事件
APP_I2C_EVT_TX_CPLT = 1	I2C发送完成事件
APP_I2C_EVT_RX_DATA = 2	I2C接收完成事件
APP_I2C_ABORT = 3	I2C Master传输中止事件

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.2.4 app_i2c_state_t

枚举定义如下：

表 2-146 app_i2c_state_t枚举成员

成员	描述
APP_I2C_INVALID = 0	I2C不可用
APP_I2C_ACTIVITY = 1	I2C激活
APP_I2C_SLEEP = 2	I2C睡眠

版权所有 © 2025 深圳市汇顶科技股份有限公司

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.2.5 app_i2c_dma_state_t

枚举定义如下：

表 2-147 app_i2c_dma_state_t枚举成员

成员	描述
APP_I2C_DMA_INVALID = 0	I2C DMA不可用
APP_I2C_DMA_ACTIVITY = 1	I2C DMA激活

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2c.h。

2.10.3 接口函数

表 2-148 I2C驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_i2c_init	Y	Y	Y	Y	Y
app_i2c_deinit	Y	Y	Y	Y	Y
app_i2c_receive_sync	Y	Y	Y	Y	Y
app_i2c_receive_async	Y	Y	Y	Y	Y
app_i2c_transmit_sync	Y	Y	Y	Y	Y
app_i2c_transmit_async	Y	Y	Y	Y	Y
app_i2c_mem_read_sync	Y	Y	Y	Y	Y
app_i2c_mem_read_async	Y	Y	Y	Y	Y
app_i2c_mem_write_sync	Y	Y	Y	Y	Y
app_i2c_mem_write_async	Y	Y	Y	Y	Y
app_i2c_get_handle	Y	Y	Y	Y	Y
app_i2c_master_abort_it	Y	Y	Y	Y	Y
app_i2c_timing_adjust	Y	Y	Y	Y	Y
app_i2c_timing_get	Y	Y	Y	Y	Y
app_i2c_transmit_receive_sync	Y	N	N	N	N

2.10.3.1 app_i2c_init

表 2-149 app_i2c_init接口

函数原型	uint16_t app_i2c_init(app_i2c_params_t *p_params, app_i2c_evt_handler_t evt_handler)
功能说明	初始化I2C
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_i2c_evt_handler_t)(app_i2c_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

2.10.3.2 app_i2c_deinit

表 2-150 app_i2c_deinit接口

函数原型	uint16_t app_i2c_deinit(app_i2c_id_t id)
功能说明	反初始化I2C
输入参数	id：I2C模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.3 app_i2c_receive_sync

表 2-151 app_i2c_receive_sync接口

函数原型	uint16_t app_i2c_receive_sync(app_i2c_id_t id, uint16_t target_address, uint8_t *p_data, uint16_t size, uint32_t timeout)
功能说明	I2C轮询方式（同步）接收数据
输入参数	• id：I2C模块ID • target_address：I2C Slave地址（作为Slave时，该参数无效） • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位 • timeout：接收超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.4 app_i2c_receive_async

表 2-152 app_i2c_receive_async接口

函数原型	uint16_t app_i2c_receive_async(app_i2c_id_t id, uint16_t target_address, uint8_t *p_data, uint16_t size)
功能说明	I2C中断方式（异步）接收数据
输入参数	• id：I2C模块ID • target_address：I2C Slave地址（作为Slave时，该参数无效） • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10.3.5 app_i2c_transmit_sync

表 2-153 app_i2c_transmit_sync接口

函数原型	uint16_t app_i2c_receive_sync(app_i2c_id_t id, uint16_t target_address, uint8_t *p_data, uint16_t size, uint32_t timeout)
功能说明	I2C轮询方式（同步）发送数据
输入参数	• id：I2C模块ID • target_address：I2C Slave地址（作为Slave时，该参数无效） • p_data：需要发送的数据buffer指针 • size：需要发送的数据长度，以Byte为单位 • timeout：发送超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.6 app_i2c_transmit_async

表 2-154 app_i2c_transmit_async接口

函数原型	uint16_t app_i2c_transmit_async(app_i2c_id_t id, uint16_t target_address, uint8_t *p_data, uint16_t size)
功能说明	I2C中断方式（异步）发送数据
输入参数	• id：I2C模块ID • target_address：I2C Slave地址（作为Slave时，该参数无效） • p_data：接收数据的buffer • size：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.7 app_i2c_mem_read_sync

表 2-155 app_i2c_mem_read_sync接口

函数原型	uint16_t app_i2c_mem_read_sync(app_i2c_id_t id, uint16_t dev_address, uint16_t mem_address, uint16_t mem_addr_size, uint8_t *p_data, uint16_t size, uint32_t timeout)
功能说明	I2C轮询方式（同步）读取特定内存地址的数据
输入参数	• id：I2C模块ID • dev_address：I2C Slave地址（作为Slave时，该参数无效） • mem_address：需要读取的内存地址 • mem_addr_size：读取的内存地址长度 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位 • timeout：读取超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10.3.8 app_i2c_mem_read_async

表 2-156 app_i2c_mem_read_async接口

函数原型	uint16_t app_i2c_mem_read_async(app_i2c_id_t id, uint16_t dev_address, uint16_t mem_address, uint16_t mem_addr_size, uint8_t *p_data, uint16_t size)
功能说明	I2C中断方式（异步）读取特定内存地址的数据
输入参数	• id：I2C模块ID • dev_address：I2C Slave地址（作为Slave时，该参数无效） • mem_address：需要读取的内存地址 • mem_addr_size：读取的内存地址长度 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.9 app_i2c_mem_write_sync

表 2-157 app_i2c_mem_write_sync接口

函数原型	uint16_t app_i2c_mem_write_sync(app_i2c_id_t id, uint16_t dev_address, uint16_t mem_address, uint16_t mem_addr_size, uint8_t *p_data, uint16_t size, uint32_t timeout);
功能说明	I2C轮询方式（同步）向特定内存地址写入数据
输入参数	• id：I2C模块ID • dev_address：I2C Slave地址（作为Slave时，该参数无效） • mem_address：需要写入的内存地址 • mem_addr_size：写入的内存地址长度 • p_data：需要发送的数据buffer指针 • size：需要发送的数据长度，以Byte为单位 • timeout：写入超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.10 app_i2c_mem_write_async

表 2-158 app_i2c_mem_write_async接口

函数原型	uint16_t app_i2c_mem_write_async(app_i2c_id_t id, uint16_t dev_address, uint16_t mem_address, uint16_t mem_addr_size, uint8_t *p_data, uint16_t size);
功能说明	I2C中断方式向（异步）特定内存地址写入数据
输入参数	• id：I2C模块ID • dev_address：I2C Slave地址（作为Slave时，该参数无效） • mem_address：需要写入的内存地址 • mem_addr_size：写入的内存地址长度

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• p_data：需要发送的数据buffer指针 • size：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.11 app_i2c_get_handle

表 2-159 app_i2c_get_handle接口

函数原型	i2c_handle_t *app_i2c_get_handle(app_i2c_id_t id)
功能说明	获取I2C句柄
输入参数	id：I2C模块ID
返回值	I2C句柄指针
备注	

2.10.3.12 app_i2c_master_abort_it

表 2-160 app_i2c_master_abort_it接口

函数原型	uint16_t app_i2c_master_abort_it(app_i2c_id_t id)
功能说明	I2C中止Master当前的传输（非阻塞模式）
输入参数	id：I2C模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.13 app_i2c_timing_adjust

表 2-161 app_i2c_timing_adjust接口

函数原型	uint16_t app_i2c_timing_adjust(app_i2c_id_t id, uint32_t timing_type, int32_t delta)
功能说明	对I2C时序进行微调，包括SCL高电平长度、低电平长度以及SDA保持时间
输入参数	• id：I2C模块ID • timing_type：时序类型（I2C_TIMING_SS_SCL_LOW、I2C_TIMING_SS_SCL_HIGH、I2C_TIMING_FS_SCL_LOW、I2C_TIMING_FS_SCL_HIGH、I2C_TIMING_HS_SCL_LOW、I2C_TIMING_HS_SCL_HIGH、I2C_TIMING_FS_SPK、I2C_TIMING_HS_SPK、I2C_TIMING_SDA_TX_HOLD、I2C_TIMING_SDA_RX_HOLD） • delta：时序需要修改的值，单位为I2C的工作时钟（sclk）周期
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	若SCL通信时钟周期长度等于4倍的I2C的工作时钟周期长度，当高电平的时序值等于1时，即1个工作时钟（sclk）周期，则高电平在时钟周期占比1/4；如不设置，则默认高低电平时间相同 由于上拉电阻以及电路走线差异，可能会导致I2C时序无法满足标准，导致通信出现异常，可通过此接口进行微调，入参delta为相对变化量，在初始化完成后调用此接口。I2C工作时钟频率在GR551x上等于系统时钟频率，在GR5xx其他系列芯片上等于串行时钟频率

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10.3.14 app_i2c_timing_get

表 2-162 app_i2c_timing_get接口

函数原型	uint16_t app_i2c_timing_get(app_i2c_id_t id, uint32_t timing_type, uint32_t *p_timing_value)
功能说明	获取I2C时序，包括SCL高电平长度、低电平长度以及SDA保持时间
输入参数	• id：I2C模块ID • timing_type：时序类型 • p_timing_value：对应时序的绝对值，单位为I2C的工作时钟（sclk）周期
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.3.15 app_i2c_transmit_receive_sync

表 2-163 app_i2c_transmit_receive_sync接口

函数原型	uint16_t app_i2c_transmit_receive_sync(app_i2c_id_t id, uint16_t dev_address, uint8_t *p_tdata, uint16_t tsize, uint8_t *p_rdata, uint16_t rsize, uint32_t timeout)
功能说明	I2C Master轮询方式（同步）向特定内存地址写入数据后读取数据，中间没有检测停止位。 I2C Slave轮询方式（同步）接收数据后发送数据，中间没有停止位。
输入参数	• id：I2C模块ID • dev_address：I2C Slave地址（作为Slave时，该参数无效） • p_data：需要发送的数据buffer指针 • tsize：需要发送的数据长度，以Byte为单位 • p_rdata：需要接收的数据buffer指针 • rsize：需要接收的数据长度，以Byte为单位 • timeout：发送与接收的总超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.4 DMA接口函数

表 2-164 DMA接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_i2c_dma_init	Y	Y	Y	Y	Y
app_i2c_dma_deinit	Y	Y	Y	Y	Y
app_i2c_dma_receive_async	Y	Y	Y	Y	Y
app_i2c_dma_transmit_async	Y	Y	Y	Y	Y
app_i2c_dma_mem_read_async	Y	Y	Y	Y	Y
app_i2c_dma_mem_write_async	Y	Y	Y	Y	Y

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10.4.1 app_i2c_dma_init

表 2-165 app_i2c_dma_init接口

函数原型	uint16_t app_i2c_dma_init(app_i2c_params_t *p_params)
功能说明	初始化I2C的DMA方式传输
输入参数	p_params：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.4.2 app_i2c_dma_deinit

表 2-166 app_i2c_dma_deinit接口

函数原型	uint16_t app_i2c_dma_deinit(app_i2c_id_t id)
功能说明	反初始化I2C的DMA方式传输
输入参数	id：I2C模块ID
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.4.3 app_i2c_dma_receive_async

表 2-167 app_i2c_dma_receive_async接口

函数原型	uint16_t app_i2c_dma_receive_async(app_i2c_id_t id, uint16_t target_address, uint8_t *p_data, uint16_t size)
功能说明	I2C以DMA方式（异步）接收数据
输入参数	• id：I2C模块ID • target_address：I2C Slave地址（作为Slave时，该参数无效） • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.4.4 app_i2c_dma_transmit_async

表 2-168 app_i2c_dma_transmit_async接口

函数原型	uint16_t app_i2c_dma_transmit_async(app_i2c_id_t id, uint16_t target_address, uint8_t *p_data, uint16_t size)
功能说明	I2C以DMA方式（异步）发送数据
输入参数	• id：I2C模块ID • target_address：I2C Slave地址（作为Slave时，该参数无效） • p_data：接收数据的buffer • size：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义

版权所有 © 2025 深圳市汇顶科技股份有限公司

备注	

2.10.4.5 app_i2c_dma_mem_read_async

表 2-169 app_i2c_dma_mem_read_async接口

函数原型	uint16_t app_i2c_dma_mem_read_async(app_i2c_id_t id, uint16_t dev_address, uint16_t mem_address, uint16_t mem_addr_size, uint8_t *p_data, uint16_t size)
功能说明	I2C以DMA方式（异步）读取特定内存地址的数据
输入参数	• id：I2C模块ID • dev_address：I2C Slave地址（作为Slave时，该参数无效） • mem_address：需要读取的内存地址 • mem_addr_size：读取的内存地址长度 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.4.6 app_i2c_dma_mem_write_async

表 2-170 app_i2c_dma_mem_write_async接口

函数原型	uint16_t app_i2c_dma_mem_write_async(app_i2c_id_t id, uint16_t dev_address, uint16_t mem_address, uint16_t mem_addr_size, uint8_t *p_data, uint16_t size)
功能说明	I2C以DMA方式（异步）向特定内存地址写入数据
输入参数	• id：I2C模块ID • dev_address：I2C Slave地址（作为Slave时，该参数无效） • mem_address：需要写入的内存地址 • mem_addr_size：写入的内存地址长度 • p_data：需要发送的数据buffer指针 • size：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.10.5 示例工程

2.10.5.1 I2C Master和Slave相互收发数据（中断方式）

I2C Master和Slave示例实现I2C之间的通信，其中一端为Master主设备，另一端为Slave从设备。Master端工程路径：SDK_Folder\projects\peripheral\i2c\app_i2c_master，Slave端工程路径：SDK_Folder\projects\peripheral\i2c\app_i2c_slave。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10.5.1.1 流程图

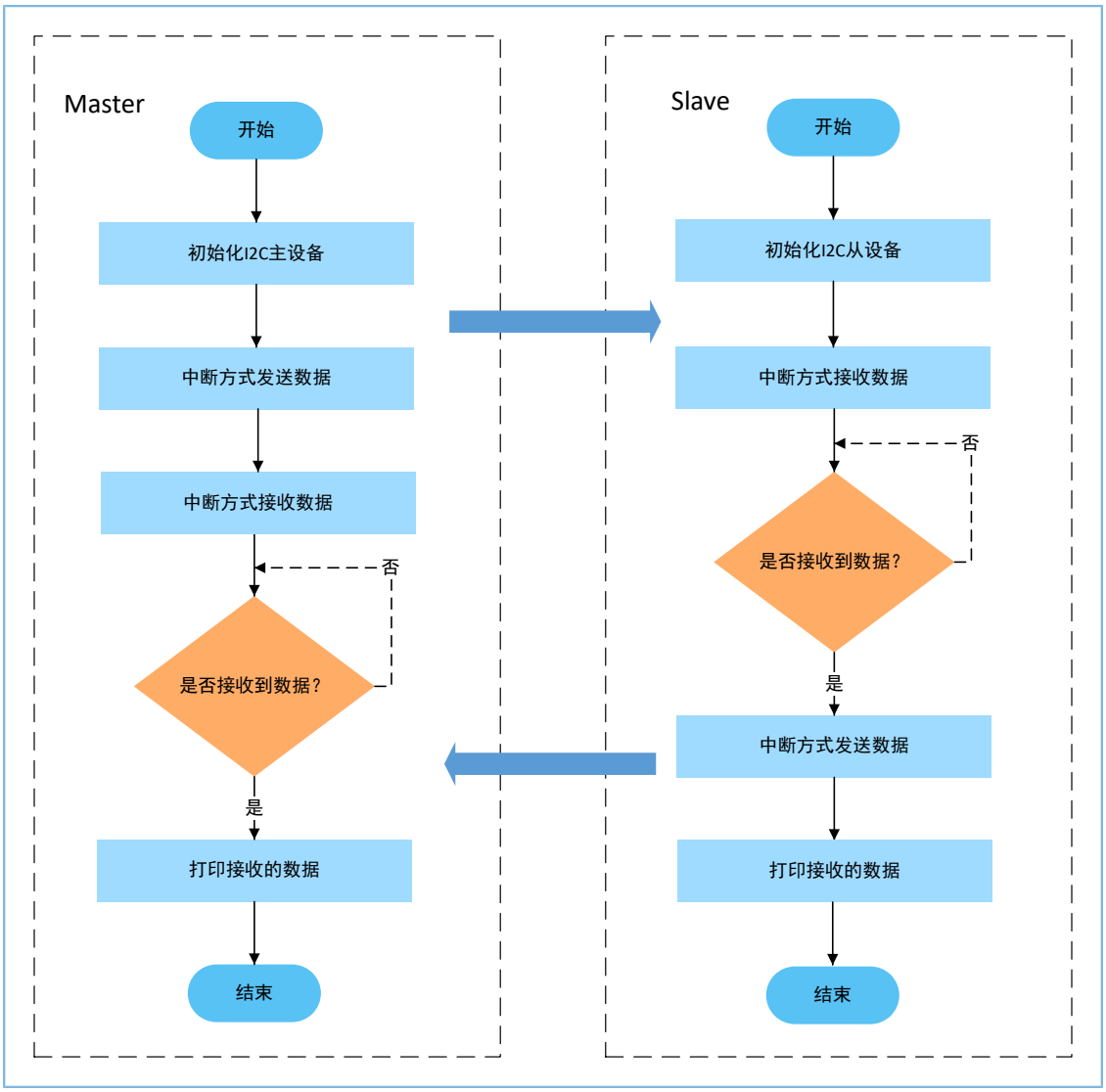

图 2-13 I2C Master和Slave对测（中断方式）示例流程

2.10.5.1.2 初始化参数

1. Master参数配置

```
app_i2c_params_t master_params = {
    .id = MASTER_I2C_ID,
    .role = APP_I2C_ROLE_MASTER,
    .pin_cfg = {
        .scl = {
            .type = APP_I2C_MASTER_SCL_IO_TYPE,
            .mux  = APP_I2C_MASTER_SCL_PINMUX,
            .pin  = APP_I2C_MASTER_SCL_PIN,
            .pull = APP_IO_NOPULL,
        },
        .sda = {
            .type = APP_I2C_MASTER_SDA_IO_TYPE,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
            .mux  = APP_I2C_MASTER_SDA_PINMUX,
            .pin  = APP_I2C_MASTER_SDA_PIN,
            .pull = APP_IO_NOPULL,
        },
    },
    .dma_cfg = {
        .tx_dma_instance = DMA0,
        .rx_dma_instance = DMA0,
        .tx_dma_channel  = DMA_Channel2,
        .rx_dma_channel  = DMA_Channel3,
    },
    .init = {
        .speed = I2C_SPEED_100K,
        .own_address = MASTER_DEV_ADDR,
        .addressing_mode = I2C_ADDRESSINGMODE_7BIT,
        .general_call_mode = I2C_GENERALCALL_DISABLE,
    },
};
```

- id：I2C Master的ID，具体配置值请参考*board_SK.h*。
- role：角色配置为APP_I2C_ROLE_MASTER。
- pin_cfg：引脚配置。
  - pin_cfg.scl：时钟引脚配置，包括IO类型、复用MUX模式、引脚和上下拉模式。具体配置值，请参考*board_SK.h*。
  - pin_cfg.sda：数据引脚配置，配置IO复用成IIC的复用模式，IO类型和引脚，上下拉模式，具体配置值，请参考*board_SK.h*。

  以GR5526为例，下面是*board_SK.h*中相关定义：

```
/*******I2C IO CONFIG*************************/
#define APP_I2C_MASTER_SCL_PIN              APP_IO_PIN_6
#define APP_I2C_MASTER_SDA_PIN              APP_IO_PIN_7
#define APP_I2C_MASTER_SCL_IO_TYPE          APP_IO_TYPE_GPIOA
#define APP_I2C_MASTER_SDA_IO_TYPE          APP_IO_TYPE_GPIOA
#define APP_I2C_MASTER_SCL_PINMUX           APP_IO_MUX_0
#define APP_I2C_MASTER_SDA_PINMUX           APP_IO_MUX_0
```

- dma_cfg：DMA参数配置
  - dma_cfg.tx_dma_instance：发送使用的DMA实例
  - dma_cfg.rx_dma_instance：接收使用的DMA实例
  - dma_cfg.tx_dma_channel：发送使用的DMA实例通道，选择通道2
  - dma_cfg.rx_dma_channel：接收使用的DMA实例通道，选择通道3
- init：串口传输参数配置
  - init.speed：传输速率，选择I2C_SPEED_100K，即为100 kHz

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.own_address：主设备地址，定义如下

```
#define MASTER_DEV_ADDR         0x4D
```

- init.addressing_mode：地址模式，7 bits地址
- init.general_call_mode：使能/禁用通用广播模式，选择I2C_GENERALCALL_DISABLE，表示禁用广播模式。

2. Slave参数配置

```
app_i2c_params_t slave_params = {
    .id = SLAVE_I2C_ID,
    .role = APP_I2C_ROLE_SLAVE,
    .pin_cfg = {
        .scl = {
            .type = APP_I2C_SLAVE_SCL_IO_TYPE,
            .mux  = APP_I2C_SLAVE_SCL_PINMUX,
            .pin  = APP_I2C_SLAVE_SCL_PIN,
            .pull = APP_IO_NOPULL,
        },
        .sda = {
            .type = APP_I2C_SLAVE_SDA_IO_TYPE,
            .mux  = APP_I2C_SLAVE_SDA_PINMUX,
            .pin  = APP_I2C_SLAVE_SDA_PIN,
            .pull = APP_IO_NOPULL,
        },
    },
    .dma_cfg = {
        .tx_dma_instance = DMA0,
        .rx_dma_instance = DMA0,
        .tx_dma_channel  = DMA_Channel2,
        .rx_dma_channel  = DMA_Channel3,
    },
    .init = {
        .speed = I2C_SPEED_100K,
        .own_address = SLAVE_DEV_ADDR,
        .addressing_mode = I2C_ADDRESSINGMODE_7BIT,
        .general_call_mode = I2C_GENERALCALL_DISABLE,
    },
};
```

- id：定义Slave使用ID，具体配置值请参考*board_SK.h*。
- role：角色配置为APP_I2C_ROLE_SLAVE。
- pin_cfg：引脚配置。
  - pin_cfg.scl：时钟引脚配置，配置IO复用成IIC的复用模式，IO类型和引脚，上下拉模式，具体配置值请参考*board_SK.h*。

版权所有 © 2025 深圳市汇顶科技股份有限公司

- pin_cfg.scl sda：数据引脚配置，配置IO复用成IIC的复用模式，IO类型和引脚，上下拉模式，具体配置值请参考*board_SK.h*。

以GR5526为例，下面是*board_SK.h*中相关定义：

```
#define APP_I2C_SLAVE_ID                    APP_I2C_ID_3
#define APP_I2C_SLAVE_SCL_PIN               APP_IO_PIN_6
#define APP_I2C_SLAVE_SDA_PIN               APP_IO_PIN_7
#define APP_I2C_SLAVE_SCL_IO_TYPE           APP_IO_TYPE_GPIOA
#define APP_I2C_SLAVE_SDA_IO_TYPE           APP_IO_TYPE_GPIOA
#define APP_I2C_SLAVE_SCL_PINMUX            APP_IO_MUX_0
#define APP_I2C_SLAVE_SDA_PINMUX            APP_IO_MUX_0
```

- dma_cfg：DMA参数配置
  - dma_cfg .tx_dma_instance：发送使用的DMA实例
  - dma_cfg .rx_dma_instance：接收使用的DMA实例
  - dma_cfg .tx_dma_channel：发送使用的DMA实例通道，选择通道2
  - dma_cfg .rx_dma_channel：接收使用的DMA实例通道，选择通道3
- init：串口传输参数配置
  - init.speed：传输速率，选择I2C_SPEED_100K，100 kHz
  - init. own_address：本设备地址，作为slave时有效，定义如下

```
#define SLAVE_DEV_ADDR                  0x55
```

  - init. addressing_mode：地址模式，7 bits地址
  - init. general_call_mode：配置使能/不使能通用广播模式，此处选择I2C_GENERALCALL_DISABLE表示不使能。

说明:

SCL和SDA都需要外部接上拉电阻。

2.10.5.1.3 重要函数

- 模块初始化

```
ret = app_i2c_init(&master_params, app_i2c_master_evt_handler)
```

- Slave接收数据，第二位I2C Slave地址无效，此处配置为0；rdata为接收数据存放的数组，接收数据长度256字节。

```
app_i2c_receive_async(SLAVE_I2C_ID, 0, rdata, 256)
```

- Master发送数据，其中从设备地址为SLAVE_DEV_ADDR，待发送数据为wdata，发送数据长度256字节。

```
ret = app_i2c_transmit_async(MASTER_I2C_ID, SLAVE_DEV_ADDR, wdata, 256)
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

- Slave发送数据，第二位I2C Slave地址无效，此处配置为0；wdata为发送数据存放的数组，发送数据长度256字节。

```
app_i2c_transmit_async(SLAVE_I2C_ID, 0, wdata, 256)
```

- Master接收数据，其中从设备地址为SLAVE_DEV_ADDR，待接收数据为rdata，接收数据长度256字节。

```
ret = app_i2c_receive_async(MASTER_I2C_ID, SLAVE_DEV_ADDR, rdata, 256)
```

2.10.5.1.4 测试验证

1. 硬件准备：2块开发板（A板和B板）
2. 引脚连接：参考示例工程代码使用IO进行连接

测试步骤：本示例采用Master和Slave对测方式。Master端代码路径：`projects\peripheral\i2c\app_i2c_maste`，Slave端代码路径：`projects\peripheral\i2c\app_i2c_slave`。连接引脚后，B板下载Slave程序，A板下载Master程序，需保证Slave先于Master准备就绪。

Master端打印日志如下：

```
app i2c interrrupt test.
I2C master send start.

[18:45:56.937] I2C master read start.
I2C master received:
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07
0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17
0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27
0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47
0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57
0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67
0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77
0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87
0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97
0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7
0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7
0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7
0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7
0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7
0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7
0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
```

Slave端打印日志如下：

```
app i2c slave interrrupt test.
I2C slave receive start.

[18:55:31.681] I2C slave received:
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07
0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17
0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27
0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47
0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57
0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67
0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77
0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87
0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97
0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7
0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7
0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7
0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7
0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7
0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7
0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
I2C slave send start.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10.5.2 TP报点

GR5526屏幕TP模块如下图所示，包含引脚有RESET、SCL、SDL和INT（中断）。RESET引脚主要通过复位信号复位TP模块，INT引脚在触摸屏幕时产生中断信号，触发报点中断。I2C的SCL和SDL读取触摸点的位置信息。工程路径：SDK_Folder\projects\peripheral\i2c\app_i2c_tp。该示例仅适用于GR5526。

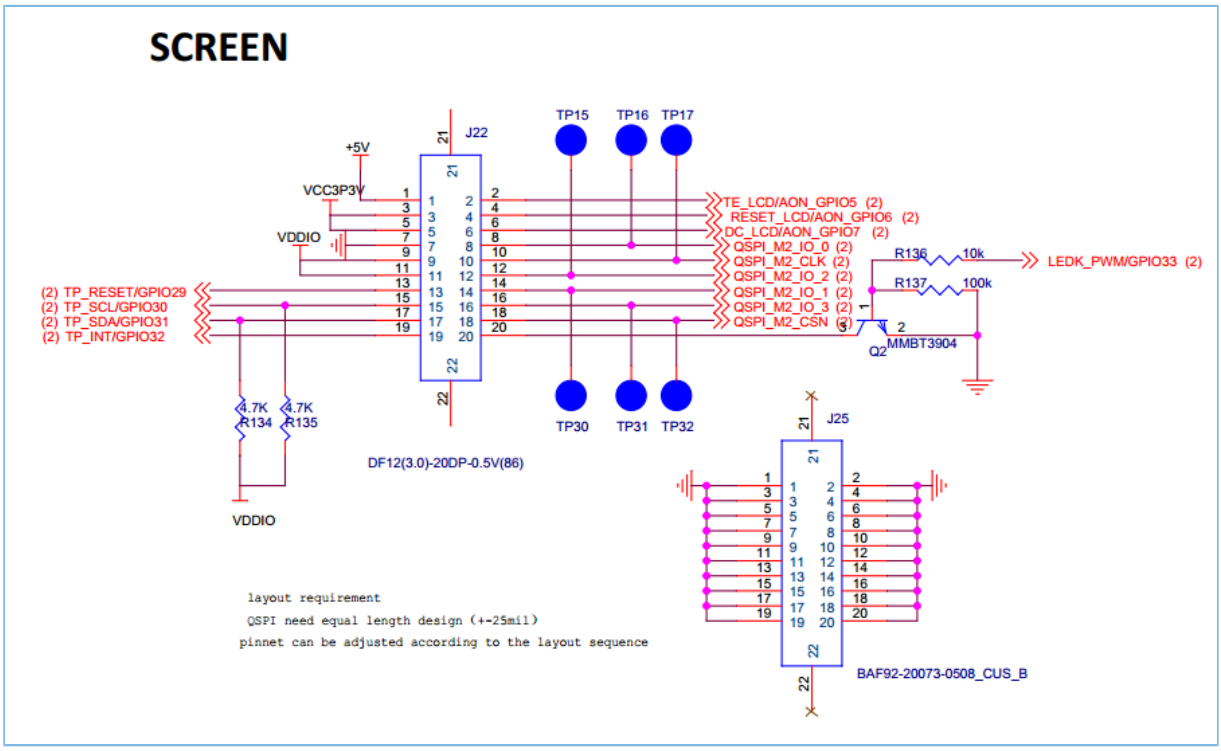

图 2-14 GR5526屏幕TP模块示意图

2.10.5.2.1 初始化参数

1. I2C配置

```
static app_i2c_params_t tp_params = {
    .id = TP_I2C_ID,
    .role = APP_I2C_ROLE_MASTER,
    .pin_cfg = {
        .scl = {
            .type = TP_SCL_IO_TYPE,
            .mux  = TP_SCL_IO_MUX,
            .pin  = TP_SCL_IO_PIN,
            .pull = APP_IO_PULLUP,
        },
        .sda = {
            .type = TP_SDA_IO_TYPE,
            .mux  = TP_SDA_IO_MUX,
            .pin  = TP_SDA_IO_PIN,
            .pull = APP_IO_PULLUP,
        },
    },
    .dma_cfg = {
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
    #if (APP_DRIVER_CHIP_TYPE == APP_DRIVER_GR5332X)
        .tx_dma_instance = DMA0,
        .rx_dma_instance = DMA0,
    #else
        .tx_dma_instance = DMA1,
        .rx_dma_instance = DMA1,
    #endif
        .tx_dma_channel  = DMA_Channel0,
        .rx_dma_channel  = DMA_Channel1,
    },
    .init = {
        .speed = I2C_SPEED_400K,
        .own_address = 0x00,
        .addressing_mode = I2C_ADDRESSINGMODE_7BIT,
        .general_call_mode = I2C_GENERALCALL_DISABLE,
    },
};
```

- id：定义Slave使用ID，具体配置值请参考*board_SK.h*。
- role：角色配置为APP_I2C_ROLE_SLAVE。
- pin_cfg：引脚配置
  - pin_cfg.scl：时钟引脚配置，配置IO复用成IIC的复用模式，IO类型和引脚、上下拉模式，具体配置值请参考*board_SK.h*。
  - pin_cfg.scl sda：数据引脚配置，配置IO复用成IIC的复用模式，IO类型和引脚、上下拉模式，具体配置值请参考*board_SK.h*。

  以GR5526为例，下面是*board_SK.h*中相关定义：

```
 #define TP_RST_IO_TYPE     APP_IO_TYPE_GPIOB
 #define TP_RST_IO_MUX      APP_IO_MUX_7
 #define TP_RST_IO_PIN      APP_IO_PIN_13

 #define TP_INT_IO_TYPE     APP_IO_TYPE_GPIOC
 #define TP_INT_IO_MUX      APP_IO_MUX_7
 #define TP_INT_IO_PIN      APP_IO_PIN_0

 #define TP_SCL_IO_TYPE     APP_IO_TYPE_GPIOB
 #define TP_SCL_IO_MUX      APP_IO_MUX_3
 #define TP_SCL_IO_PIN      APP_IO_PIN_14

 #define TP_SDA_IO_TYPE     APP_IO_TYPE_GPIOB
 #define TP_SDA_IO_MUX      APP_IO_MUX_3
 #define TP_SDA_IO_PIN      APP_IO_PIN_15
```

- dma_cfg：DMA参数配置
  - dma_cfg .tx_dma_instance：发送使用的DMA实例
  - dma_cfg .rx_dma_instance：接收使用的DMA实例

版权所有 © 2025 深圳市汇顶科技股份有限公司

- dma_cfg .tx_dma_channel：发送使用的DMA实例通道，选择通道2
- dma_cfg .rx_dma_channel：接收使用的DMA实例通道，选择通道3

- init：串口传输参数配置
  - init.speed：传输速率，选择I2C_SPEED_400K，400 kHz
  - init. own_address：主设备地址，0x00
  - init. addressing_mode：地址模式，7 bits地址
  - init. general_call_mode：配置使能/禁用通用广播模式，此处选择I2C_GENERALCALL_DISABLE表示不使能。

2. RESET引脚配置

   配置GPIO29为RESET引脚，GPIO29为上拉输出模式。

```
app_io_init_t io_init = APP_IO_DEFAULT_CONFIG;
io_init.pull = APP_IO_PULLUP;
io_init.mode = APP_IO_MODE_OUTPUT;
io_init.pin  = TP_RST_IO_PIN;
io_init.mux  = TP_RST_IO_MUX;
app_io_init(TP_RST_IO_TYPE, &io_init);
```

3. INT引脚配置

   配置GPIO32为中断输入引脚，GPIO32下降沿触发，上拉输入模式。

```
app_gpiote_param_t param[1] =
{
     TP_INT_IO_TYPE,
     TP_INT_IO_PIN,
     APP_IO_MODE_IT_FALLING,
     APP_IO_PULLUP,
     tp_interrupt_callback,
 };
app_gpiote_init(param, 1);
```

2.10.5.2.2 重要函数

- 初始化I2C

```
app_i2c_init(&tp_params, NULL)
```

- 初始化RESET和INT引脚，并注册中断函数

```
tp_pin_init()
```

- 使能TP模块

```
tp_rst()
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.10.5.2.3 测试验证

连接带屏幕的开发板到PC端，打开串口调试助手，连接串口，用手触摸屏幕，查看报点信息，如下显示触摸点的屏幕位置。

```
TP coordinates: x: 0x d4, y: 0x cd
TP coordinates: x: 0x d4, y: 0x cc
TP coordinates: x: 0x d3, y: 0x cb
TP coordinates: x: 0x d3, y: 0x cb
TP coordinates: x: 0x d2, y: 0x cb
TP coordinates: x: 0x d2, y: 0x cb
TP coordinates: x: 0x d2, y: 0x cb
```

2.11 APP I2S驱动

2.11.1 结构体

2.11.1.1 app_i2s_pin_t

结构体定义如下：

表 2-171 app_i2s_pin_t结构体成员

成员	描述	取值
app_io_type_t type	设置IO类型	同2.2.1.2 app_io_evt_t结构体定义
app_io_mux_t mux	设置IO映射功能	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
uint32_t pin	选择IO引脚	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
app_io_pull_t pull	设置IO上拉/下拉/浮空	同2.2.1.1 app_io_init_t结构体中描述

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.1.2 app_i2s_pin_cfg_t

结构体定义如下：

表 2-172 app_i2s_pin_cfg_t结构体成员

成员	描述	取值
app_i2s_pin_t ws	I2S WS引脚配置	参考2.11.1.1 app_i2s_pin_t结构体
app_i2s_pin_t sdo	I2S SDO引脚配置	参考2.11.1.1 app_i2s_pin_t结构体
app_i2s_pin_t sdi	I2S SDI引脚配置	参考2.11.1.1 app_i2s_pin_t结构体
app_i2s_pin_t sclk	I2S SCLK引脚配置	参考2.11.1.1 app_i2s_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.1.3 app_i2s_dma_cfg_t

结构体定义如下：

表 2-173 app_i2s_dma_cfg_t结构体成员

成员	描述	取值
dma_reg_t *tx_dma_instance	I2S TX指定DMA模块配置	参考2.4.1.1 app_dma_params_t中DMA实例描述
dma_reg_t *rx_dma_instance	I2S RX指定DMA模块配置	参考2.4.1.1 app_dma_params_t中DMA实例描述
dma_channel_t tx_dma_channel	I2S TX的DMA通道选择	参考2.4.1.1 app_dma_params_t中通道描述
dma_channel_t rx_dma_channel	I2S RX的DMA通道选择	参考2.4.1.1 app_dma_params_t中通道描述

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.1.4 app_i2s_params_t

结构体定义如下：

表 2-174 app_i2s_params_t结构体成员

成员	描述	取值
app_i2s_id_t id	设置使用的I2S模块ID	该参数的取值可以是下列值中的任意一个： • APP_I2S_ID_SLAVE（I2S Slave） • APP_I2S_ID_MASTER（I2S Master） • APP_I2S_ID_MAX（I2S模块总个数，无其他用途）
app_i2s_pin_cfg_t pin_cfg	I2S引脚功能设置	参考2.11.1.2 app_i2s_pin_cfg_t结构体
app_i2s_dma_cfg_t dma_cfg	I2S DMA参数配置	参考2.11.1.3 app_i2s_dma_cfg_t结构体
i2s_init_t init	I2S初始化参数配置	参考2.11.1.5 i2s_init_t结构体
i2s_env_t i2s_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.1.5 i2s_init_t

结构体定义如下：

表 2-175 i2s_init_t结构体成员

成员	描述	取值
uint32_t data_size	I2S数据位宽	该参数的取值可以是下列值中的任意一个：

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• I2S_DATASIZE_12BIT（12 bits） • I2S_DATASIZE_16BIT（16 bits） • I2S_DATASIZE_20BIT（20 bits） • I2S_DATASIZE_24BIT（24 bits） • I2S_DATASIZE_32BIT（32 bits）
uint32_t clock_source	I2S时钟源	该参数的取值可以是下列值中的任意一个： • I2S_CLOCK_SRC_96M（96 MHz） • I2S_CLOCK_SRC_64M（64 MHz） • I2S_CLOCK_SRC_32M（32 MHz）
uint32_t ws_cycles	I2S WS时钟长度	该参数的取值可以是下列值中的任意一个： • I2S_WS_CYCLES_16（16个BCLK时钟周期，代表传输16 bits的数据） • I2S_WS_CYCLES_24（24个BCLK时钟周期，代表传输24 bits的数据） • I2S_WS_CYCLES_32（32个BCLK时钟周期，代表传输32 bits的数据）
uint32_t audio_freq	I2S音频采样率，单位：Hz	最高96000（I2S_WS_CYCLES_16）
uint32_t channel_active	I2S激活通道（I2S_CHANNEL_NUM > 1）	暂不可用

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.1.6 app_i2s_evt_t

结构体定义如下：

表 2-176 app_i2s_evt_t结构体成员

成员	描述	取值
app_i2s_evt_type_t type	I2S事件类型	该参数的取值可以是下列值中的任意一个： • APP_I2S_EVT_ERROR（I2S传输错误） • APP_I2S_EVT_TX_CPLT（I2S发送完成） • APP_I2S_EVT_RX_DATA（I2S接收完成） • APP_I2S_EVT_TX_RX（I2S发送接收完成）
union { uint32_t error_code; uint16_t size; }data;	• uint32_t error_code：I2S事件返回的错误号 • uint16_t size：I2S已接收/发送的数据长度 说明： 若事件类型为传输错误，此处返回错误号； 若事件类型为接收/发送完成，此处显示已接收/发送的数据长度	I2S错误号如下： • HAL_I2S_ERROR_NONE（无错误） • HAL_I2S_ERROR_TIMEOUT（操作超时） • HAL_I2S_ERROR_TRANSFER（传输中断错误） • HAL_I2S_ERROR_DMA（DMA传输错误） • HAL_I2S_ERROR_INVALID_PARAM（非法传参错误）

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.2 枚举

2.11.2.1 app_i2s_id_t

枚举定义如下：

表 2-177 app_i2s_id_t枚举成员

成员	描述
APP_I2S_ID_MASTER = 0	I2S Master模块
APP_I2S_ID_SLAVE = 1	I2S Slave模块
APP_I2S_ID_MAX = 2	I2S模块总个数，用于参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.2.2 app_i2s_evt_type_t

枚举定义如下：

表 2-178 app_i2s_evt_type_t枚举成员

成员	描述
APP_I2S_EVT_ERROR = 0	I2S传输错误事件
APP_I2S_EVT_TX_CPLT = 1	I2S发送完成事件
APP_I2S_EVT_RX_DATA = 2	I2S接收完成事件
APP_I2S_EVT_TX_RX = 3	I2S发送接收完成事件

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.2.3 app_i2s_dma_state_t

枚举定义如下：

表 2-179 app_i2s_dma_state_t枚举成员

成员	描述
APP_I2S_DMA_INVALID	I2S DMA不可用
APP_I2S_DMA_ACTIVITY	I2S DMA激活

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.2.4 app_i2s_state_t

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-180 app_i2s_state_t枚举成员

成员	描述
APP_I2S_INVALID = 0	I2S不可用
APP_I2S_ACTIVITY = 1	I2S激活
APP_I2S_SLEEP = 2	I2S睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_i2s.h。

2.11.3 接口函数

表 2-181 I2S驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_i2s_init	Y	Y	Y	N	N
app_i2s_deinit	Y	Y	Y	N	N
app_i2s_receive_async	Y	Y	Y	N	N
app_i2s_receive_sync	Y	Y	Y	N	N
app_i2s_transmit_async	Y	Y	Y	N	N
app_i2s_transmit_sync	Y	Y	Y	N	N
app_i2s_transmit_receive_sync	Y	Y	Y	N	N
app_i2s_transmit_receive_async	Y	Y	Y	N	N
app_i2s_enable	Y	Y	Y	N	N
app_i2s_disable	Y	Y	Y	N	N
app_i2s_enable_clock	Y	Y	Y	N	N
app_i2s_disable_clock	Y	Y	Y	N	N
app_i2s_flush_tx_fifo	Y	Y	Y	N	N
app_i2s_flush_rx_fifo	Y	Y	Y	N	N
app_i2s_get_handle	Y	Y	Y	N	N
app_i2s_abort	Y	Y	Y	N	N

2.11.3.1 app_i2s_init

表 2-182 app_i2s_init接口

函数原型	uint16_t app_i2s_init(app_i2s_params_t *p_params, app_i2s_evt_handler_t evt_handler)
功能说明	初始化I2S
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义

版权所有 © 2025 深圳市汇顶科技股份有限公司

备注	回调函数形式参照typedef void (*app_i2s_evt_handler_t)(app_i2s_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

2.11.3.2 app_i2s_deinit

表 2-183 app_i2s_deinit接口

函数原型	uint16_t app_i2s_deinit(app_i2s_id_t id)
功能说明	反初始化I2S
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.3.3 app_i2s_receive_sync

表 2-184 app_i2s_receive_sync接口

函数原型	uint16_t app_i2s_receive_sync(app_i2s_id_t id, uint16_t *p_data, uint16_t size, uint32_t timeout)
功能说明	I2S轮询方式（同步）接收数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_data：接收数据的buffer • size：需要接收的数据长度，以半字（Half-Word）为单位 • timeout：接收超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	I2S同步方式接收数据，如果被打断，可能会由于FIFO溢出导致数据丢失，建议使用DMA方式，或通过关闭中断等方式避免被打断。

2.11.3.4 app_i2s_receive_async

表 2-185 app_i2s_receive_async接口

函数原型	uint16_t app_i2s_receive_async(app_i2s_id_t id, uint16_t *p_data, uint16_t size)
功能说明	I2S中断方式（异步）接收数据
输入参数	• id：I2S模块ID （参考2.11.2.1 app_i2s_id_t） • p_data：接收数据的buffer • size：需要接收的数据长度，以半字（Half-Word）为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	在RTOS睡眠模式下，有溢出风险，推荐使用DMA接口

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.3.5 app_i2s_transmit_sync

表 2-186 app_i2s_transmit_sync接口

函数原型	uint16_t app_i2s_transmit_sync(app_i2s_id_t id, uint16_t *p_data, uint16_t size, uint32_t timeout)
功能说明	I2S轮询方式（同步）发送数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_data：需要发送的数据buffer指针 • size：需要发送的数据长度，以半字（Half-Word）为单位 • timeout：发送超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	I2S同步发送时，如果被打断，可能会产生无意义的0数据，建议使用DMA方式，或者通过关闭中断的方式避免被打断。

2.11.3.6 app_i2s_transmit_async

表 2-187 app_i2s_transmit_async接口

函数原型	uint16_t app_i2s_transmit_async(app_i2s_id_t id, uint16_t *p_data, uint16_t size)
功能说明	I2S中断方式（异步）发送数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_data：接收数据的buffer • size：需要发送的数据长度，以半字（Half-Word）为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	从设备调用这个接口发送时，如果要发送的数据小于36个Half-Word的长度，不管是否发送成功都会进入发送完成中断

2.11.3.7 app_i2s_transmit_receive_sync

表 2-188 app_i2s_transmit_receive_sync接口

函数原型	uint16_t app_i2s_transmit_receive_sync(app_i2s_id_t id, uint16_t *p_tx_data, uint16_t *p_rx_data, uint32_t length, uint32_t timeout)
功能说明	I2S轮询方式（同步）发送接收数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • length：需要接收的数据长度，以半字（Half-Word）为单位 • timeout：接收超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	同步接口如果被打断则存在数据异常风险，建议使用DMA接口。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.3.8 app_i2s_transmit_receive_async

表 2-189 app_i2s_transmit_receive_async接口

函数原型	uint16_t app_i2s_transmit_receive_async(app_i2s_id_t id, uint16_t *p_tx_data, uint16_t *p_rx_data, uint32_t length)
功能说明	I2S使用中断方式（异步）发送接收数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • length：需要接收的数据长度，以半字（Half-Word）为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	在RTOS睡眠模式下，有溢出风险，推荐使用DMA接口

2.11.3.9 app_i2s_enable

表 2-190 app_i2s_enable接口

函数原型	uint16_t app_i2s_enable(app_i2s_id_t id)
功能说明	使能I2S模块
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.3.10 app_i2s_disable

表 2-191 app_i2s_disable接口

函数原型	uint16_t app_i2s_disable(app_i2s_id_t id)
功能说明	禁用I2S模块
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.3.11 app_i2s_enable_clock

表 2-192 app_i2s_enable_clock接口

函数原型	uint16_t app_i2s_enable_clock(app_i2s_id_t id)
功能说明	使能I2S Clock，仅Master可使用
输入参数	id：I2S模块ID。参数只能传APP_I2S_ID_MASTER，否则会返回参数错误
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.3.12 app_i2s_disable_clock

表 2-193 app_i2s_disable_clock接口

函数原型	uint16_t app_i2s_disable_clock(app_i2s_id_t id)
功能说明	禁用I2S Clock，仅Master可使用
输入参数	id：I2S模块ID，参数只能传APP_I2S_ID_MASTER，否则会返回参数错误
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.3.13 app_i2s_flush_tx_fifo

表 2-194 app_i2s_flush_tx_fifo接口

函数原型	uint16_t app_i2s_flush_tx_fifo(app_i2s_id_t id)
功能说明	清除I2S写缓冲区
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.3.14 app_i2s_flush_rx_fifo

表 2-195 app_i2s_flush_rx_fifo接口

函数原型	uint16_t app_i2s_flush_rx_fifo(app_i2s_id_t id)
功能说明	清除I2S读缓冲区
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.3.15 app_i2s_get_handle

表 2-196 app_i2s_get_handle接口

函数原型	i2s_handle_t *app_i2s_get_handle(app_i2s_id_t id)
功能说明	获取I2S句柄
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	I2S句柄指针
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.3.16 app_i2s_abort

表 2-197 app_i2s_abort接口

函数原型	uint16_t app_i2s_abort(app_i2s_id_t id)
功能说明	中止I2S数据传输
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.4 DMA接口函数

表 2-198 DMA接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_i2s_dma_init	Y	Y	Y	N	N
app_i2s_dma_deinit	Y	Y	Y	N	N
app_i2s_dma_receive_async	Y	Y	Y	N	N
app_i2s_dma_transmit_async	Y	Y	Y	N	N
app_i2s_dma_transmit_receive_async	Y	Y	Y	N	N

2.11.4.1 app_i2s_dma_init

表 2-199 app_i2s_dma_init接口

函数原型	uint16_t app_i2s_dma_init(app_i2s_params_t *p_params)
功能说明	初始化I2S的DMA方式传输
输入参数	p_params：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.4.2 app_i2s_dma_deinit

表 2-200 app_i2s_dma_deinit接口

函数原型	uint16_t app_i2s_dma_deinit(app_i2s_id_t id)
功能说明	反初始化I2S的DMA方式传输
输入参数	id：I2S模块ID（参考2.11.2.1 app_i2s_id_t）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.4.3 app_i2s_dma_receive_async

表 2-201 app_i2s_dma_receive_async接口

函数原型	uint16_t app_i2s_dma_receive_async(app_i2s_id_t id, uint16_t *p_data, uint16_t size)
功能说明	I2S DMA方式（异步）接收数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_data：接收数据的buffer • size：需要接收的数据长度，以半字（Half-Word）为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.4.4 app_i2s_dma_transmit_async

表 2-202 app_i2s_dma_transmit_async接口

函数原型	uint16_t app_i2s_dma_transmit_async(app_i2s_id_t id, uint16_t *p_data, uint16_t size)
功能说明	I2S DMA方式（异步）发送数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_data：接收数据的buffer • size：需要发送的数据长度，以半字（Half-Word）为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.4.5 app_i2s_dma_transmit_receive_async

表 2-203 app_i2s_dma_transmit_receive_async接口

函数原型	uint16_t app_i2s_dma_transmit_receive_async(app_i2s_id_t id, uint16_t *p_tx_data, uint16_t *p_rx_data, uint32_t length)
功能说明	I2S使用DMA方式（异步）发送接收数据
输入参数	• id：I2S模块ID（参考2.11.2.1 app_i2s_id_t） • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • length：需要接收的数据长度，以半字（Half-Word）为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.11.5 示例工程

2.11.5.1 I2S Master和Slave中断传输

示例实现I2S之间的通信，其中一端为Master主设备，另一端为Slave从设备。工程路径：SDK_Folder\projects\peripheral\i2s\app_i2s。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.5.1.1 流程图

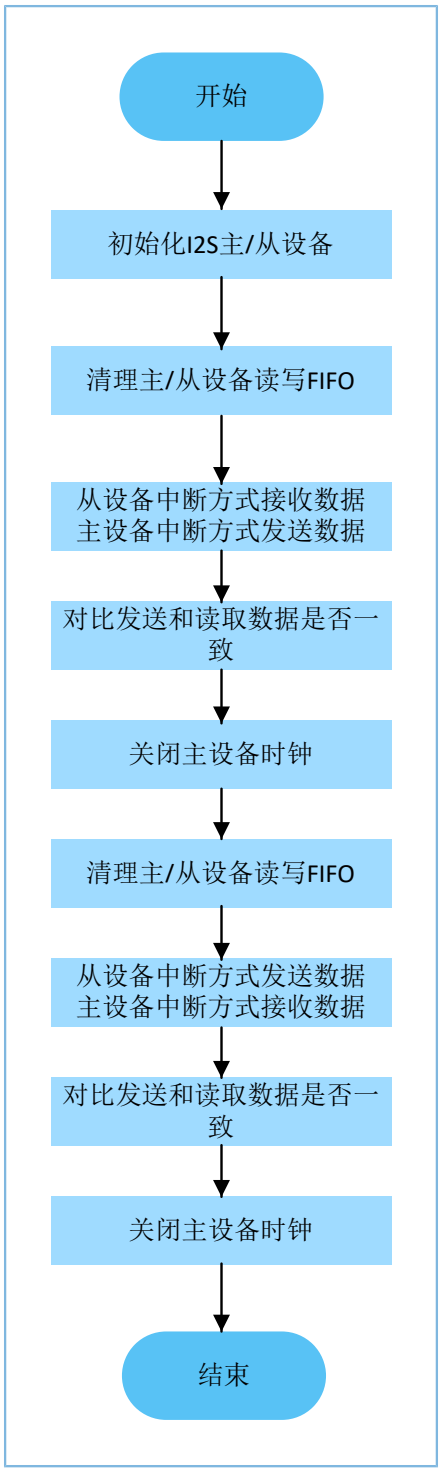

图 2-15 I2S Master和Slave中断传输流程

2.11.5.1.2 初始化参数

I2S主设备初始化参数，从设备类似：

```
app_i2s_params_t i2sm_params = {
    .id = APP_I2S_ID_MASTER,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
    .pin_cfg = {
        .ws = {
            .type = APP_I2S_MASTER_WS_IO_TYPE,
            .mux  = APP_I2S_MASTER_WS_PINMUX,
            .pin  = APP_I2S_MASTER_WS_PIN,
            .pull = APP_IO_NOPULL,
        },
        .sdo = {
            .type = APP_I2S_MASTER_SDO_IO_TYPE,
            .mux  = APP_I2S_MASTER_SDO_PINMUX,
            .pin  = APP_I2S_MASTER_SDO_PIN,
            .pull = APP_IO_NOPULL,
        },
        .sdi = {
            .type = APP_I2S_MASTER_SDI_IO_TYPE,
            .mux  = APP_I2S_MASTER_SDI_PINMUX,
            .pin  = APP_I2S_MASTER_SDI_PIN,
            .pull = APP_IO_NOPULL,
        },
        .sclk = {
            .type = APP_I2S_MASTER_SCLK_IO_TYPE,
            .mux  = APP_I2S_MASTER_SCLK_PINMUX,
            .pin  = APP_I2S_MASTER_SCLK_PIN,
            .pull = APP_IO_NOPULL,
        },
    },
     .dma_cfg = {
#if (APP_DRIVER_CHIP_TYPE != APP_DRIVER_GR551X)
         .tx_dma_instance = DMA1,
         .rx_dma_instance = DMA1,
#else
         .tx_dma_instance = DMA,
         .rx_dma_instance = DMA,
#endif
         .tx_dma_channel  = DMA_Channel2,
         .rx_dma_channel  = DMA_Channel3,
     },
     .init = {
         .data_size    = I2S_DATASIZE_32BIT,
         .clock_source = I2S_CLOCK_SRC_96M,
 #if (APP_DRIVER_CHIP_TYPE != APP_DRIVER_GR551X)
         .ws_cycles    = I2S_WS_CYCLES_32,
 #endif
         .audio_freq = 48000,
     }, };
```

- id：角色配置

  APP_I2S_ID_MASTER：主设备

版权所有 © 2025 深圳市汇顶科技股份有限公司

- pin_cfg：引脚配置

  主要配置IO类型、复用MUX模式、引脚和上下拉模式，具体选择的参数请参考*board_SK.h*。

  以GR5526为例，下面是具体IO配置：

```
/*******I2S IO CONFIG*************************/
#define APP_I2S_MASTER_WS_PIN               APP_IO_PIN_6
#define APP_I2S_MASTER_SDO_PIN              APP_IO_PIN_7
#define APP_I2S_MASTER_SDI_PIN              APP_IO_PIN_8
#define APP_I2S_MASTER_SCLK_PIN             APP_IO_PIN_9
#define APP_I2S_MASTER_WS_IO_TYPE           APP_IO_TYPE_GPIOA
#define APP_I2S_MASTER_SDO_IO_TYPE          APP_IO_TYPE_GPIOA
#define APP_I2S_MASTER_SDI_IO_TYPE          APP_IO_TYPE_GPIOA
#define APP_I2S_MASTER_SCLK_IO_TYPE         APP_IO_TYPE_GPIOA
#define APP_I2S_MASTER_WS_PINMUX            APP_IO_MUX_3
#define APP_I2S_MASTER_SDO_PINMUX           APP_IO_MUX_3
#define APP_I2S_MASTER_SDI_PINMUX           APP_IO_MUX_3
#define APP_I2S_MASTER_SCLK_PINMUX          APP_IO_MUX_3

#define APP_I2S_SLAVE_WS_PIN                APP_IO_PIN_3
#define APP_I2S_SLAVE_SDO_PIN               APP_IO_PIN_2
#define APP_I2S_SLAVE_SDI_PIN               APP_IO_PIN_1
#define APP_I2S_SLAVE_SCLK_PIN              APP_IO_PIN_0
#define APP_I2S_SLAVE_WS_IO_TYPE            APP_IO_TYPE_MSIO
#define APP_I2S_SLAVE_SDO_IO_TYPE           APP_IO_TYPE_MSIO
#define APP_I2S_SLAVE_SDI_IO_TYPE           APP_IO_TYPE_MSIO
#define APP_I2S_SLAVE_SCLK_IO_TYPE          APP_IO_TYPE_MSIO
#define APP_I2S_SLAVE_WS_PINMUX             APP_IO_MUX_4
#define APP_I2S_SLAVE_SDO_PINMUX            APP_IO_MUX_4
#define APP_I2S_SLAVE_SDI_PINMUX            APP_IO_MUX_4
#define APP_I2S_SLAVE_SCLK_PINMUX           APP_IO_MUX_4
```

  使用MSIO3作为Slave端WS引脚，MSIO0作为Slave端SCL引脚，MSIO2作为Slave端SDO引脚，MSIO1作为Slave端SDI引脚。使用GPIO6作为Master端WS引脚，GPIO9作为Master端SCL引脚，GPIO7作为Master端SDO引脚，GPIO8作为Master端SDI引脚。

- dma_cfg：DMA配置
  - dma_cfg.tx_dma_instance：I2S发送使用的DMA实例
  - dma_cfg.rx_dma_instance：I2S接收使用的DMA实例
  - dma_cfg.tx_dma_channel：I2S发送使用的DMA实例通道
  - dma_cfg.rx_dma_channel：I2S接收使用的DMA实例通道
- init：I2S传输参数配置
  - init.data_size：I2S数据位宽I2S_DATASIZE_32BIT，数据位宽32 bits
  - init.clock_source：I2S时钟源I2S_CLOCK_SRC_96M，选择96 MHz的时钟源
  - init.ws_cycles：I2S WS时钟长度I2S_WS_CYCLES_32，32个BCLK时钟周期代表传输32 bits的数据

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.audio_freq：I2S音频采样频率48000 Hz

2.11.5.1.3 重要函数

- I2S初始化

```
ret = app_i2s_init(&i2sm_params, app_i2sm_callback)
```

- I2S清理TX FIFO

```
app_i2s_flush_tx_fifo(APP_I2S_ID_MASTER);
```

- I2S清理RX FIFO

```
app_i2s_flush_rx_fifo(APP_I2S_ID_MASTER)
```

- I2S接收数据

```
app_i2s_receive_async(APP_I2S_ID_SLAVE, rdata, sizeof(rdata) >> 2)
```

- I2S发送数据

```
app_i2s_transmit_async(APP_I2S_ID_MASTER, wdata, sizeof(wdata) >> 2)
```

2.11.5.1.4 测试验证

1. 按代码配置连接引脚。
2. 连接开发板到PC端，打开串口助手，连接串口，查看串口日志如下：

```
I2SM interrupt write start.
Please check I2SS Received:
0x00010000 0x00030002 0x00050004 0x00070006
0x00090008 0x000B000A 0x000D000C 0x000F000E
0x00110010 0x00130012 0x00150014 0x00170016
0x00190018 0x001B001A 0x001D001C 0x001F001E
0x00210020 0x00230022 0x00250024 0x00270026
0x00290028 0x002B002A 0x002D002C 0x002F002E
0x00310030 0x00330032 0x00350034 0x00370036
0x00390038 0x003B003A 0x003D003C 0x003F003E
0x00410040 0x00430042 0x00450044 0x00470046
0x00490048 0x004B004A 0x004D004C 0x004F004E
0x00510050 0x00530052 0x00550054 0x00570056
0x00590058 0x005B005A 0x005D005C 0x005F005E
0x00610060 0x00630062 0x00650064 0x00670066
0x00690068 0x006B006A 0x006D006C 0x006F006E
0x00710070 0x00730072 0x00750074 0x00770076
0x00790078 0x007B007A 0x007D007C 0x007F007E
0x00810080 0x00830082 0x00850084 0x00870086
0x00890088 0x008B008A 0x008D008C 0x008F008E
0x00910090 0x00930092 0x00950094 0x00970096
0x00990098 0x009B009A 0x009D009C 0x009F009E
0x00A100A0 0x00A300A2 0x00A500A4 0x00A700A6
0x00A900A8 0x00AB00AA 0x00AD00AC 0x00AF00AE
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0x00B100B0 0x00B300B2 0x00B500B4 0x00B700B6
0x00B900B8 0x00BB00BA 0x00BD00BC 0x00BF00BE
0x00C100C0 0x00C300C2 0x00C500C4 0x00C700C6
0x00C900C8 0x00CB00CA 0x00CD00CC 0x00CF00CE
0x00D100D0 0x00D300D2 0x00D500D4 0x00D700D6
0x00D900D8 0x00DB00DA 0x00DD00DC 0x00DF00DE
0x00E100E0 0x00E300E2 0x00E500E4 0x00E700E6
0x00E900E8 0x00EB00EA 0x00ED00EC 0x00EF00EE
0x00F100F0 0x00F300F2 0x00F500F4 0x00F700F6
0x00F900F8 0x00FB00FA 0x00FD00FC 0x00FF00FE
0x01010100 0x01030102 0x01050104 0x01070106
0x01090108 0x010B010A 0x010D010C 0x010F010E
0x01110110 0x01130112 0x01150114 0x01170116
0x01190118 0x011B011A 0x011D011C 0x011F011E
0x01210120 0x01230122 0x01250124 0x01270126
0x01290128 0x012B012A 0x012D012C 0x012F012E
0x01310130 0x01330132 0x01350134 0x01370136
0x01390138 0x013B013A 0x013D013C 0x013F013E
0x01410140 0x01430142 0x01450144 0x01470146
0x01490148 0x014B014A 0x014D014C 0x014F014E
0x01510150 0x01530152 0x01550154 0x01570156
0x01590158 0x015B015A 0x015D015C 0x015F015E
0x01610160 0x01630162 0x01650164 0x01670166
0x01690168 0x016B016A 0x016D016C 0x016F016E
0x01710170 0x01730172 0x01750174 0x01770176
0x01790178 0x017B017A 0x017D017C 0x017F017E
0x01810180 0x01830182 0x01850184 0x01870186
0x01890188 0x018B018A 0x018D018C 0x018F018E
0x01910190 0x01930192 0x01950194 0x01970196
0x01990198 0x019B019A 0x019D019C 0x019F019E
0x01A101A0 0x01A301A2 0x01A501A4 0x01A701A6
0x01A901A8 0x01AB01AA 0x01AD01AC 0x01AF01AE
0x01B101B0 0x01B301B2 0x01B501B4 0x01B701B6
0x01B901B8 0x01BB01BA 0x01BD01BC 0x01BF01BE
0x01C101C0 0x01C301C2 0x01C501C4 0x01C701C6
0x01C901C8 0x01CB01CA 0x01CD01CC 0x01CF01CE
0x01D101D0 0x01D301D2 0x01D501D4 0x01D701D6
0x01D901D8 0x01DB01DA 0x01DD01DC 0x01DF01DE
0x01E101E0 0x01E301E2 0x01E501E4 0x01E701E6
0x01E901E8 0x01EB01EA 0x01ED01EC 0x01EF01EE
0x01F101F0 0x01F301F2 0x01F501F4 0x01F701F6
0x01F901F8 0x01FB01FA 0x01FD01FC 0x01FF01FE


I2SM interrupt read start.
Please check I2SM Received:
0x00010000 0x00030002 0x00050004 0x00070006
0x00090008 0x000B000A 0x000D000C 0x000F000E
0x00110010 0x00130012 0x00150014 0x00170016
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0x00190018 0x001B001A 0x001D001C 0x001F001E
0x00210020 0x00230022 0x00250024 0x00270026
0x00290028 0x002B002A 0x002D002C 0x002F002E
0x00310030 0x00330032 0x00350034 0x00370036
0x00390038 0x003B003A 0x003D003C 0x003F003E
0x00410040 0x00430042 0x00450044 0x00470046
0x00490048 0x004B004A 0x004D004C 0x004F004E
0x00510050 0x00530052 0x00550054 0x00570056
0x00590058 0x005B005A 0x005D005C 0x005F005E
0x00610060 0x00630062 0x00650064 0x00670066
0x00690068 0x006B006A 0x006D006C 0x006F006E
0x00710070 0x00730072 0x00750074 0x00770076
0x00790078 0x007B007A 0x007D007C 0x007F007E
0x00810080 0x00830082 0x00850084 0x00870086
0x00890088 0x008B008A 0x008D008C 0x008F008E
0x00910090 0x00930092 0x00950094 0x00970096
0x00990098 0x009B009A 0x009D009C 0x009F009E
0x00A100A0 0x00A300A2 0x00A500A4 0x00A700A6
0x00A900A8 0x00AB00AA 0x00AD00AC 0x00AF00AE
0x00B100B0 0x00B300B2 0x00B500B4 0x00B700B6
0x00B900B8 0x00BB00BA 0x00BD00BC 0x00BF00BE
0x00C100C0 0x00C300C2 0x00C500C4 0x00C700C6
0x00C900C8 0x00CB00CA 0x00CD00CC 0x00CF00CE
0x00D100D0 0x00D300D2 0x00D500D4 0x00D700D6
0x00D900D8 0x00DB00DA 0x00DD00DC 0x00DF00DE
0x00E100E0 0x00E300E2 0x00E500E4 0x00E700E6
0x00E900E8 0x00EB00EA 0x00ED00EC 0x00EF00EE
0x00F100F0 0x00F300F2 0x00F500F4 0x00F700F6
0x00F900F8 0x00FB00FA 0x00FD00FC 0x00FF00FE
0x01010100 0x01030102 0x01050104 0x01070106
0x01090108 0x010B010A 0x010D010C 0x010F010E
0x01110110 0x01130112 0x01150114 0x01170116
0x01190118 0x011B011A 0x011D011C 0x011F011E
0x01210120 0x01230122 0x01250124 0x01270126
0x01290128 0x012B012A 0x012D012C 0x012F012E
0x01310130 0x01330132 0x01350134 0x01370136
0x01390138 0x013B013A 0x013D013C 0x013F013E
0x01410140 0x01430142 0x01450144 0x01470146
0x01490148 0x014B014A 0x014D014C 0x014F014E
0x01510150 0x01530152 0x01550154 0x01570156
0x01590158 0x015B015A 0x015D015C 0x015F015E
0x01610160 0x01630162 0x01650164 0x01670166
0x01690168 0x016B016A 0x016D016C 0x016F016E
0x01710170 0x01730172 0x01750174 0x01770176
0x01790178 0x017B017A 0x017D017C 0x017F017E
0x01810180 0x01830182 0x01850184 0x01870186
0x01890188 0x018B018A 0x018D018C 0x018F018E
0x01910190 0x01930192 0x01950194 0x01970196
0x01990198 0x019B019A 0x019D019C 0x019F019E
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0x01A101A0 0x01A301A2 0x01A501A4 0x01A701A6
0x01A901A8 0x01AB01AA 0x01AD01AC 0x01AF01AE
0x01B101B0 0x01B301B2 0x01B501B4 0x01B701B6
0x01B901B8 0x01BB01BA 0x01BD01BC 0x01BF01BE
0x01C101C0 0x01C301C2 0x01C501C4 0x01C701C6
0x01C901C8 0x01CB01CA 0x01CD01CC 0x01CF01CE
0x01D101D0 0x01D301D2 0x01D501D4 0x01D701D6
0x01D901D8 0x01DB01DA 0x01DD01DC 0x01DF01DE
0x01E101E0 0x01E301E2 0x01E501E4 0x01E701E6
0x01E901E8 0x01EB01EA 0x01ED01EC 0x01EF01EE
0x01F101F0 0x01F301F2 0x01F501F4 0x01F701F6
0x01F901F8 0x01FB01FA 0x01FD01FC 0x01FF01FE
```

2.11.5.2 I2S Master和Slave DMA传输

示例实现I2S之间的DMA通信，其中一端为Master主设备，另一端为Slave从设备。工程路径：SDK_Folder\projects\peripheral\i2s\app_i2s。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.5.2.1 流程图

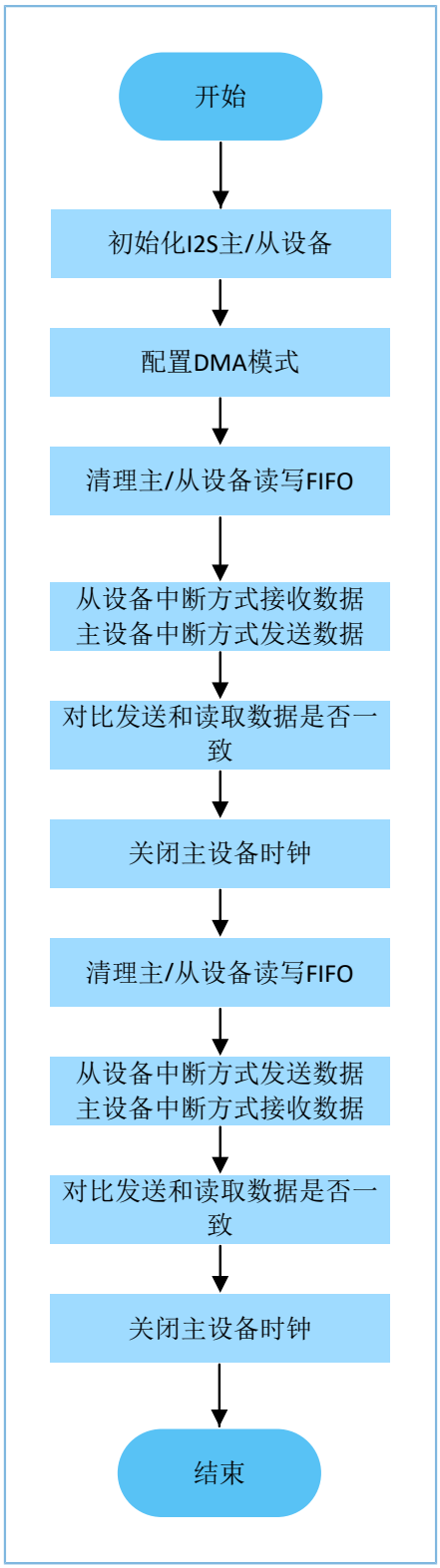

图 2-16 I2S Master和Slave DMA传输流程

2.11.5.2.2 初始化参数

I2S初始化参数请参见2.11.5.1.2 初始化参数。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.11.5.2.3 重要函数

- I2S初始化

```
ret = app_i2s_init(&i2sm_params, app_i2sm_callback);
```

- 配置DMA模式

```
ret = app_i2s_dma_init(&i2sm_params);
```

- I2S清理TX FIFO

```
app_i2s_flush_tx_fifo(APP_I2S_ID_MASTER);
```

- I2S清理RX FIFO

```
app_i2s_flush_rx_fifo(APP_I2S_ID_MASTER)
```

- I2S接收数据

```
app_i2s_receive_async(APP_I2S_ID_SLAVE, rdata, sizeof(rdata) >> 2)
```

- I2S发送数据

```
app_i2s_transmit_async(APP_I2S_ID_MASTER, wdata, sizeof(wdata) >> 2)
```

2.11.5.2.4 测试验证

1. 按代码配置连接引脚。
2. 连接开发板到PC端，打开串口助手，连接串口，查看串口日志如下：

```
I2SM dma write start.
Please check I2SS Received:
0x00010000 0x00030002 0x00050004 0x00070006
0x00090008 0x000B000A 0x000D000C 0x000F000E
0x00110010 0x00130012 0x00150014 0x00170016
0x00190018 0x001B001A 0x001D001C 0x001F001E
0x00210020 0x00230022 0x00250024 0x00270026
0x00290028 0x002B002A 0x002D002C 0x002F002E
0x00310030 0x00330032 0x00350034 0x00370036
0x00390038 0x003B003A 0x003D003C 0x003F003E
0x00410040 0x00430042 0x00450044 0x00470046
0x00490048 0x004B004A 0x004D004C 0x004F004E
0x00510050 0x00530052 0x00550054 0x00570056
0x00590058 0x005B005A 0x005D005C 0x005F005E
0x00610060 0x00630062 0x00650064 0x00670066
0x00690068 0x006B006A 0x006D006C 0x006F006E
0x00710070 0x00730072 0x00750074 0x00770076
0x00790078 0x007B007A 0x007D007C 0x007F007E
0x00810080 0x00830082 0x00850084 0x00870086
0x00890088 0x008B008A 0x008D008C 0x008F008E
0x00910090 0x00930092 0x00950094 0x00970096
0x00990098 0x009B009A 0x009D009C 0x009F009E
0x00A100A0 0x00A300A2 0x00A500A4 0x00A700A6
0x00A900A8 0x00AB00AA 0x00AD00AC 0x00AF00AE
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0x00B100B0 0x00B300B2 0x00B500B4 0x00B700B6
0x00B900B8 0x00BB00BA 0x00BD00BC 0x00BF00BE
0x00C100C0 0x00C300C2 0x00C500C4 0x00C700C6
0x00C900C8 0x00CB00CA 0x00CD00CC 0x00CF00CE
0x00D100D0 0x00D300D2 0x00D500D4 0x00D700D6
0x00D900D8 0x00DB00DA 0x00DD00DC 0x00DF00DE
0x00E100E0 0x00E300E2 0x00E500E4 0x00E700E6
0x00E900E8 0x00EB00EA 0x00ED00EC 0x00EF00EE
0x00F100F0 0x00F300F2 0x00F500F4 0x00F700F6
0x00F900F8 0x00FB00FA 0x00FD00FC 0x00FF00FE
0x01010100 0x01030102 0x01050104 0x01070106
0x01090108 0x010B010A 0x010D010C 0x010F010E
0x01110110 0x01130112 0x01150114 0x01170116
0x01190118 0x011B011A 0x011D011C 0x011F011E
0x01210120 0x01230122 0x01250124 0x01270126
0x01290128 0x012B012A 0x012D012C 0x012F012E
0x01310130 0x01330132 0x01350134 0x01370136
0x01390138 0x013B013A 0x013D013C 0x013F013E
0x01410140 0x01430142 0x01450144 0x01470146
0x01490148 0x014B014A 0x014D014C 0x014F014E
0x01510150 0x01530152 0x01550154 0x01570156
0x01590158 0x015B015A 0x015D015C 0x015F015E
0x01610160 0x01630162 0x01650164 0x01670166
0x01690168 0x016B016A 0x016D016C 0x016F016E
0x01710170 0x01730172 0x01750174 0x01770176
0x01790178 0x017B017A 0x017D017C 0x017F017E
0x01810180 0x01830182 0x01850184 0x01870186
0x01890188 0x018B018A 0x018D018C 0x018F018E
0x01910190 0x01930192 0x01950194 0x01970196
0x01990198 0x019B019A 0x019D019C 0x019F019E
0x01A101A0 0x01A301A2 0x01A501A4 0x01A701A6
0x01A901A8 0x01AB01AA 0x01AD01AC 0x01AF01AE
0x01B101B0 0x01B301B2 0x01B501B4 0x01B701B6
0x01B901B8 0x01BB01BA 0x01BD01BC 0x01BF01BE
0x01C101C0 0x01C301C2 0x01C501C4 0x01C701C6
0x01C901C8 0x01CB01CA 0x01CD01CC 0x01CF01CE
0x01D101D0 0x01D301D2 0x01D501D4 0x01D701D6
0x01D901D8 0x01DB01DA 0x01DD01DC 0x01DF01DE
0x01E101E0 0x01E301E2 0x01E501E4 0x01E701E6
0x01E901E8 0x01EB01EA 0x01ED01EC 0x01EF01EE
0x01F101F0 0x01F301F2 0x01F501F4 0x01F701F6
0x01F901F8 0x01FB01FA 0x01FD01FC 0x01FF01FE


I2SM dma read start.
Please check I2SM Received:
0x00010000 0x00030002 0x00050004 0x00070006
0x00090008 0x000B000A 0x000D000C 0x000F000E
0x00110010 0x00130012 0x00150014 0x00170016
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0x00190018 0x001B001A 0x001D001C 0x001F001E
0x00210020 0x00230022 0x00250024 0x00270026
0x00290028 0x002B002A 0x002D002C 0x002F002E
0x00310030 0x00330032 0x00350034 0x00370036
0x00390038 0x003B003A 0x003D003C 0x003F003E
0x00410040 0x00430042 0x00450044 0x00470046
0x00490048 0x004B004A 0x004D004C 0x004F004E
0x00510050 0x00530052 0x00550054 0x00570056
0x00590058 0x005B005A 0x005D005C 0x005F005E
0x00610060 0x00630062 0x00650064 0x00670066
0x00690068 0x006B006A 0x006D006C 0x006F006E
0x00710070 0x00730072 0x00750074 0x00770076
0x00790078 0x007B007A 0x007D007C 0x007F007E
0x00810080 0x00830082 0x00850084 0x00870086
0x00890088 0x008B008A 0x008D008C 0x008F008E
0x00910090 0x00930092 0x00950094 0x00970096
0x00990098 0x009B009A 0x009D009C 0x009F009E
0x00A100A0 0x00A300A2 0x00A500A4 0x00A700A6
0x00A900A8 0x00AB00AA 0x00AD00AC 0x00AF00AE
0x00B100B0 0x00B300B2 0x00B500B4 0x00B700B6
0x00B900B8 0x00BB00BA 0x00BD00BC 0x00BF00BE
0x00C100C0 0x00C300C2 0x00C500C4 0x00C700C6
0x00C900C8 0x00CB00CA 0x00CD00CC 0x00CF00CE
0x00D100D0 0x00D300D2 0x00D500D4 0x00D700D6
0x00D900D8 0x00DB00DA 0x00DD00DC 0x00DF00DE
0x00E100E0 0x00E300E2 0x00E500E4 0x00E700E6
0x00E900E8 0x00EB00EA 0x00ED00EC 0x00EF00EE
0x00F100F0 0x00F300F2 0x00F500F4 0x00F700F6
0x00F900F8 0x00FB00FA 0x00FD00FC 0x00FF00FE
0x01010100 0x01030102 0x01050104 0x01070106
0x01090108 0x010B010A 0x010D010C 0x010F010E
0x01110110 0x01130112 0x01150114 0x01170116
0x01190118 0x011B011A 0x011D011C 0x011F011E
0x01210120 0x01230122 0x01250124 0x01270126
0x01290128 0x012B012A 0x012D012C 0x012F012E
0x01310130 0x01330132 0x01350134 0x01370136
0x01390138 0x013B013A 0x013D013C 0x013F013E
0x01410140 0x01430142 0x01450144 0x01470146
0x01490148 0x014B014A 0x014D014C 0x014F014E
0x01510150 0x01530152 0x01550154 0x01570156
0x01590158 0x015B015A 0x015D015C 0x015F015E
0x01610160 0x01630162 0x01650164 0x01670166
0x01690168 0x016B016A 0x016D016C 0x016F016E
0x01710170 0x01730172 0x01750174 0x01770176
0x01790178 0x017B017A 0x017D017C 0x017F017E
0x01810180 0x01830182 0x01850184 0x01870186
0x01890188 0x018B018A 0x018D018C 0x018F018E
0x01910190 0x01930192 0x01950194 0x01970196
0x01990198 0x019B019A 0x019D019C 0x019F019E
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0x01A101A0 0x01A301A2 0x01A501A4 0x01A701A6
0x01A901A8 0x01AB01AA 0x01AD01AC 0x01AF01AE
0x01B101B0 0x01B301B2 0x01B501B4 0x01B701B6
0x01B901B8 0x01BB01BA 0x01BD01BC 0x01BF01BE
0x01C101C0 0x01C301C2 0x01C501C4 0x01C701C6
0x01C901C8 0x01CB01CA 0x01CD01CC 0x01CF01CE
0x01D101D0 0x01D301D2 0x01D501D4 0x01D701D6
0x01D901D8 0x01DB01DA 0x01DD01DC 0x01DF01DE
0x01E101E0 0x01E301E2 0x01E501E4 0x01E701E6
0x01E901E8 0x01EB01EA 0x01ED01EC 0x01EF01EE
0x01F101F0 0x01F301F2 0x01F501F4 0x01F701F6
0x01F901F8 0x01FB01FA 0x01FD01FC 0x01FF01FE


This example demo end.
```

2.11.5.3 I2S和PDM麦克风

开发板集成PDM模块和编解码Codec芯片，示例使用PDM模块作为音频输入，通过I2S传输音频数据到Codec芯片，播放音乐。工程路径：SDK_Folder\projects\peripheral\pdm\app_loopback。该示例仅适用于GR5526。

2.11.5.3.1 流程图

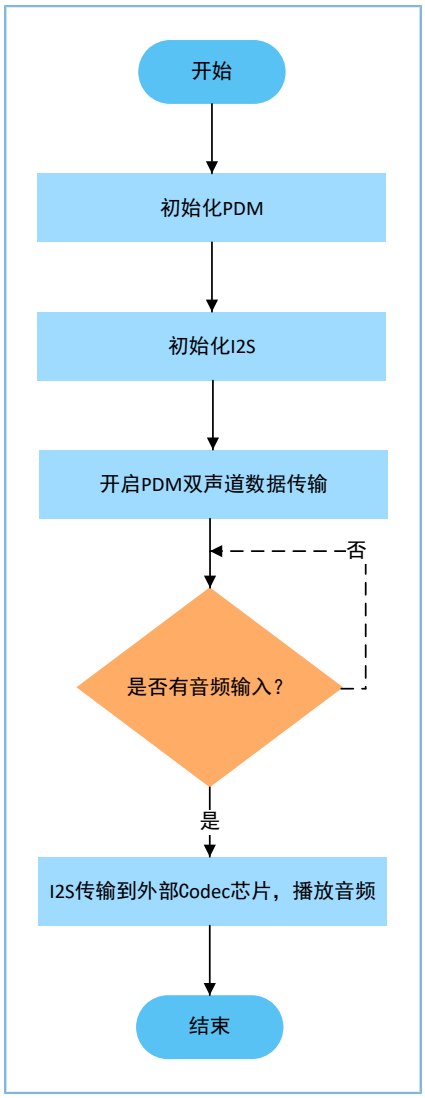

图 2-17 简易麦克风实现流程

2.11.5.3.2 初始化参数

1. PDM初始化参数

```
app_pdm_params_t g_pdm_params = {
    .pin_cfg = {
        .clk = {
            .type = APP_IO_TYPE_GPIOA,
            .mux  = APP_IO_MUX_1,
            .pin  = APP_IO_PIN_2,
            .pull = APP_IO_NOPULL,
        },
        .data = {
            .type = APP_IO_TYPE_GPIOA,
            .mux  = APP_IO_MUX_1,
            .pin  = APP_IO_PIN_3,
            .pull = APP_IO_NOPULL,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
        },
    },
    .dma_cfg = {
        .dma_instance = DMA1,
        .dma_channel  = DMA_Channel0,
    },
    .init = {
        .mode   = PDM_MODE_STEREO,
        .gain_l = 0x400U, // gain(db): 20 * log(0x400/1024) = 0db, 20 * log(0x800/1024) =
 6db
        .gain_r = 0x400U, // gain(db): 20 * log(0x400/1024) = 0db, 20 * log(0x800/1024) =
 6db
        .sample_rate = PDM_SAMPLE_RATE_16K,
    },
};
```

- pin_cfg：引脚配置
  - pin_cfg.clk：PDM时钟引脚配置，IO类型为APP_IO_TYPE_GPIOA，引脚为APP_IO_PIN_2，复用方式为APP_IO_MUX_1，并配置为上拉模式APP_IO_NOPULL
  - pin_cfg.data：PDM数据引脚配置，IO类型为APP_IO_TYPE_GPIOA，引脚为APP_IO_PIN_3，复用方式为APP_IO_MUX_1，并配置为上拉模式APP_IO_NOPULL，即PDM数据引脚为GPIO3
- dma_cfg：DMA参数配置
  - dma_cfg.dma_instance：指定使用DMA1实例
  - dma_cfg.dma_channel：指定使用DMA1实例通道，选择通道0
- init：传输参数配置
  - init.mode：立体声采样PDM_MODE_STEREO
  - init.gain_l：左声道增益
  - init.gain_r：右声道增益
  - init.sample_rate：采样频率PDM_SAMPLE_RATE_16K，即16 kHz

2. I2S初始化参数

```
app_i2s_params_t g_i2s_master_params = {
   .id = APP_I2S_ID_MASTER,
   .pin_cfg = {
       .ws = {
           .type = APP_IO_TYPE_GPIOA,
           .mux  = APP_IO_MUX_3,
           .pin  = APP_IO_PIN_6,
           .pull = APP_IO_NOPULL,
       },
       .sdo = {
           .type = APP_IO_TYPE_GPIOA,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
            .mux  = APP_IO_MUX_3,
            .pin  = APP_IO_PIN_7,
            .pull = APP_IO_NOPULL,
        },
        .sdi = {
            .type = APP_IO_TYPE_GPIOA,
            .mux  = APP_IO_MUX_3,
            .pin  = APP_IO_PIN_8,
            .pull = APP_IO_NOPULL,
        },
        .sclk = {
            .type = APP_IO_TYPE_GPIOA,
            .mux  = APP_IO_MUX_3,
            .pin  = APP_IO_PIN_9,
            .pull = APP_IO_NOPULL,
        },
    },
    .dma_cfg = {
        .tx_dma_instance = I2S_DMA,
        .rx_dma_instance = I2S_DMA,
        .tx_dma_channel  = DMA_Channel2,
        .rx_dma_channel  = DMA_Channel3,
    },
    .init = {
        .data_size    = I2S_DATASIZE_16BIT,
        .clock_source = I2S_CLOCK_SRC_64M,
        .ws_cycles    = I2S_WS_CYCLES_16,
        .audio_freq   = 16000,
    },
};
```

- id：I2S角色定义，为APP_I2S_ID_MASTER
- pin_cfg：I2S引脚定义，包含WS、SDO、SDI、SCLK引脚的IO类型、引脚、复用模式和上下拉模式
- dma_cfg：DMA参数配置
  - dma_cfg.tx_dma_instance：I2S发送使用的DMA实例
  - dma_cfg.rx_dma_instance：I2S接收使用的DMA实例，定义如下：

```
#if (APP_DRIVER_CHIP_TYPE == APP_DRIVER_GR551x)
#define I2S_DMA                    DMA
#else
#define I2S_DMA                    DMA1
#endif
```

  - dma_cfg.tx_dma_channel：I2S发送使用的DMA通道，选择DMA_Channel2，即通道2
  - dma_cfg.rx_dma_channel：I2S接收使用的DMA通道，选择DMA_Channel3，即通道3
- init：传输参数配置

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.data_size：I2S数据位宽I2S_DATASIZE_16BIT，即数据位宽为16 bits

init.clock_source：I2S时钟源I2S_CLOCK_SRC_64M，选择64 MHz的时钟源

init.ws_cycles：I2S WS时钟长度I2S_WS_CYCLES_16，16个BCLK时钟周期代表传输16 bits的数据

init.audio_freq：I2S音频采样频率，16000 Hz

2.11.5.3.3 重要函数

- 初始化PDM

```
uint16_t ret = loopback_pdm_init()
```

- 初始化I2S

```
ret = loopback_i2s_master_init(APP_I2S_TYPE_DMA)
```

- PDM开启音频数据采集传输

```
ret = app_pdm_stereo_start_dma_sg_llp(loopback_ctrl_p->pdm_audio_buf, PDM_AUDIO_FRAME_LEN,
 &loopback_ctrl_p->pdm_dma.sg_llp_config)
```

- I2S将音频数据传输到Codec芯片

```
ret = loopback_i2s_stereo_start(block_num, &loopback_ctrl_p->i2s_audio_buf[block_num *
 I2S_AUDIO_FRAME_LEN * I2S_CHANNEL_NUM], I2S_AUDIO_FRAME_LEN)
```

2.11.5.3.4 测试验证

1. 开发板连接到PC端，打开串口助手，打印如下，表明初始化成功，开始数据传输。

```
loopback_pdm_init ret = 0
loopback_i2s_master_init ret = 0
loopback_pdm_dma_start!
```

2. 连接耳机到开发板，对着麦克风讲话，通过耳机听到所讲内容。

2.12 APP SPI驱动

2.12.1 宏定义

2.12.1.1 APP_SPI_PIN_ENABLE

表 2-204 APP_SPI_PIN_ENABLE宏定义

描述	值
SPI引脚使能	1

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.12.1.2 APP_SPI_PIN_DISABLE

表 2-205 APP_SPI_PIN_DISABLE宏定义

描述	值
SPI引脚不使能	0

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.2 结构体

2.12.2.1 app_spi_params_t

结构体定义如下：

表 2-206 app_spi_params_t结构体成员

成员	描述	取值
app_spi_id_t id	设备类型	该参数的取值可以是下列值中的任意一个： • APP_SPI_ID_SLAVE（从设备） • APP_SPI_ID_MASTER（主设备）
app_spi_pin_cfg_t pin_cfg	引脚配置	参考2.12.2.2 app_spi_pin_cfg_t结构体
app_spi_dma_cfg_t dma_cfg	模式设置	参考2.12.2.4 app_spi_dma_cfg_t结构体
spi_init_t init	初始化结构体	参考2.12.2.5 spi_init_t结构体
bool is_soft_cs	是否通过软件控制CS	• 1：通过软件控制 • 0：不通过软件控制（GR551x该配置项不生效，固定使用软件CS）
spi_env_t spi_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

说明:

- 作为SPI Slave设备时，CS引脚需根据时钟极性配置上/下拉模式：当时钟极性为CS低电平有效时，CS引脚需配置为上拉模式；当时钟极性为CS高电平有效时，CS引脚需配置为下拉模式。
- 建议在初始化SPI Slave设备之前，确保SPI Master设备已完成初始化且处于稳定状态，以防止Master设备初始化时产生的I/O信号变化对Slave设备造成干扰。

2.12.2.2 app_spi_pin_cfg_t

结构体定义如下：

表 2-207 app_spi_pin_cfg_t结构体成员

成员	描述	取值
app_spi_pin_t cs	片选引脚配置	参考2.12.2.3 app_spi_pin_t结构体
app_spi_pin_t clk	时钟引脚配置	参考2.12.2.3 app_spi_pin_t结构体
app_spi_pin_t mosi	主输出从输入引脚配置	参考2.12.2.3 app_spi_pin_t结构体
app_spi_pin_t miso	主输入从输出引脚配置	参考2.12.2.3 app_spi_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.2.3 app_spi_pin_t

结构体定义如下：

表 2-208 app_spi_pin_t结构体成员

成员	描述	取值
app_io_type_t type	IO类型	参考2.2.1.2 app_io_evt_t中描述
app_io_mux_t mux	IO映射功能	参考2.2.1.1 app_io_init_t结构体，具体映射模式根据芯片Datasheet中的Pin Mux进行选择
uint32_t pin	IO引脚	参考2.2.1.1 app_io_init_t结构体，具体根据芯片Datasheet中SPI资源进行选择
app_io_mode_t mode	IO模式配置	APP_IO_MODE_MUX
app_io_pull_t pull	设置IO上拉/下拉/浮空	参考2.2.1.1 app_io_init_t结构体中描述
uint8_t enable	使能/不使能引脚	• true：使能 • false：不使能

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.2.4 app_spi_dma_cfg_t

结构体定义如下：

表 2-209 app_spi_dma_cfg_t结构体成员

成员	描述	取值
dma_regs_t * tx_dma_instance	SPI TX指定DMA模块配置	参考2.4.1.1 app_dma_params_t中DMA支持的外设
dma_regs_t * rx_dma_instance	SPI RX指定DMA模块配置	参考2.4.1.1 app_dma_params_t中DMA支持的外设
dma_channel_t tx_dma_channel	SPI TX的DMA通道选择	参考2.4.1.1 app_dma_params_t中通道描述
dma_channel_t rx_dma_channel	SPI RX的DMA通道选择	参考2.4.1.1 app_dma_params_t中通道描述

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
uint32_t wait_timeout_ms	轮询和死机等待的超时时间，以ms为单位	

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.2.5 spi_init_t

结构体定义如下：

表 2-210 spi_init_t结构体成员

成员	描述	取值
uint32_t data_size	SPI数据长度	该参数的取值可以是下列值中的任意一个： • SPI_DATASIZE_4BIT（4 bits） ～ SPI_DATASIZE_32BIT（32 bits）
uint32_t clock_polarity	串行时钟有效极性	该参数的取值可以是下列值中的任意一个： • SPI_POLARITY_LOW（CLK非活动状态时为低电平） • SPI_POLARITY_HIGH（CLK非活动状态时为高电平）
uint32_t clock_phase	位捕获的时钟活动边沿	该参数的取值可以是下列值中的任意一个： • SPI_PHASE_1EDGE（在第一个时钟变化沿捕获数据） • SPI_PHASE_2EDGE（在第二个时钟变化沿捕获数据）
uint32_t baudrate_prescaler	发送和接收SCK时钟波特率的预分频器值	取值范围：0 ～ 65534之间的偶数，如果是0，表示关闭clk。 分频值与波特率关系：clock rate = baudrate= peripheral CLK / baudrate_prescaler。
uint32_t ti_mode	TI mode使能/不使能	该参数的取值可以是下列值中的任意一个： • SPI_TIMODE_DISABLE（不使能） • SPI_TIMODE_ENABLE（使能）
uint32_t slave_select	选择的从设备	该参数的取值可以是下列值中的任意一个： • SPI_SLAVE_SELECT_0 • SPI_SLAVE_SELECT_1 • SPI_SLAVE_SELECT_ALL
uint32_t rx_sample_delay	RX输入端口的采样延时，以系统外设时钟周期为单位（仅适用于GR5526）	0 ～ 0x7

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_spi.h_
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_spi.h

版权所有 © 2025 深圳市汇顶科技股份有限公司

- GR5405：SDK_Folder\hal_drv\include\hal_spi.h

2.12.2.6 app_spi_evt_t

结构体定义如下：

表 2-211 app_spi_evt_t结构体成员

成员	描述	取值
app_spi_evt_type_t type	串口事件类型	该参数的取值可以是下列值中的任意一个： • APP_SPI_EVT_ERROR（SPI传输错误） • APP_SPI_EVT_TX_CPLT（SPI发送完成） • APP_SPI_EVT_RX_CPLT（SPI接收完成） • APP_SPI_EVT_TX_RX_CPLT（双工模式下，收发都完成） • APP_SPI_ABORT（SPI Master传输被中断）
union { uint32_t error_code; uint16_t size; }data;	• uint32_t error_code：SPI事件返回的错误号 • uint16_t size：SPI已接收/发送的数据长度 **说明：** 若事件类型为传输错误，此处返回错误号； 若事件类型为接收/发送完成，此处显示已接收/发送的数据长度	SPI错误号及对应含义如下： • HAL_SPI_ERROR_NONE（无错误） • HAL_SPI_ERROR_TIMEOUT（超时） • HAL_SPI_ERROR_TRANSFER（传输错误） • HAL_SPI_ERROR_DMA（DMA传输错误） • HAL_SPI_ERROR_INVALID_PARAM（无效参数）

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.3 枚举

2.12.3.1 app_spi_id_t

枚举定义如下：

表 2-212 app_spi_id_t枚举成员

成员	描述
APP_SPI_ID_SLAVE = 0	SPI作为Slave
APP_SPI_ID_MASTER = 1	SPI作为Master
APP_SPI_ID_MAX = 2	SPI Role总个数，用于参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.3.2 app_spi_evt_type_t

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-213 app_spi_evt_type_t枚举成员

成员	描述
APP_SPI_EVT_ERROR	SPI传输错误事件
APP_SPI_EVT_TX_CPLT	SPI发送完成事件
APP_SPI_EVT_RX_CPLT	SPI接收完成事件
APP_SPI_EVT_TX_RX_CPLT	双工模式下，SPI发送/接收完成
APP_SPI_ABORT	SPI Master传输中止事件

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.3.3 app_spi_dma_state_t

枚举定义如下：

表 2-214 app_spi_dma_state_t枚举成员

成员	描述
APP_SPI_DMA_INVALID = 0	SPI DMA不可用
APP_SPI_DMA_ACTIVITY = 1	SPI DMA激活

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.3.4 app_spi_state_t

枚举定义如下：

表 2-215 app_spi_state_t枚举成员

成员	描述
APP_SPI_INVALID = 0	SPI不可用
APP_SPI_ACTIVITY = 1	SPI激活
APP_SPI_SLEEP = 2	SPI睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_spi.h。

2.12.4 接口函数

表 2-216 SPI驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_spi_init	Y	Y	Y	Y	Y
app_spi_deinit	Y	Y	Y	Y	Y
app_spi_receive_async	Y	Y	Y	Y	Y
app_spi_receive_sync	Y	Y	Y	Y	Y

版权所有 © 2025 深圳市汇顶科技股份有限公司

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_spim_transmit_with_ia	N	Y	Y	Y	Y
app_spim_receive_with_ia	N	Y	Y	Y	Y
app_spi_transmit_sync	Y	Y	Y	Y	Y
app_spi_transmit_async	Y	Y	Y	Y	Y
app_spi_transmit_receive_sync	Y	Y	Y	Y	Y
app_spi_transmit_receive_async	Y	Y	Y	Y	Y
app_spi_read_eeprom_sync	Y	Y	Y	Y	Y
app_spi_read_eeprom_async	Y	Y	Y	Y	Y
app_spi_write_memory_async	Y	N	N	N	N
app_spi_read_memory_async	Y	N	N	N	N
app_spi_get_handle	Y	Y	Y	Y	Y
app_spi_transmit_high_speed_sync	Y	N	N	N	N
app_spi_receive_high_speed_sync	Y	N	N	N	N
app_spi_abort	Y	Y	Y	Y	Y

2.12.4.1 app_spi_init

表 2-217 app_spi_init接口

函数原型	uint16_t app_spi_init(app_spi_params_t *p_params, app_spi_evt_handler_t evt_handler)
功能说明	初始化SPI
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_spi_evt_handler_t)(app_spi_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理。

2.12.4.2 app_spi_deinit

表 2-218 app_spi_deinit接口

函数原型	uint16_t app_spi_deinit(app_spi_id_t id)
功能说明	反初始化SPI
输入参数	id：SPI主设备/从设备
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.12.4.3 app_spi_receive_sync

表 2-219 app_spi_receive_sync接口

函数原型	uint16_t app_spi_receive_sync(app_spi_id_t id, uint8_t *p_data, uint16_t size, uint32_t timeout)
功能说明	SPI轮询方式（同步）接收数据
输入参数	• id：SPI主设备/从设备 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位 • timeout：接收超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.4 app_spi_receive_async

表 2-220 app_spi_receive_async接口

函数原型	uint16_t app_spi_receive_async(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI中断方式（异步）接收数据
输入参数	• id：SPI主设备/从设备 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.5 app_spim_receive_with_ia

表 2-221 app_spim_receive_with_ia接口

函数原型	uint16_t app_spim_receive_with_ia(app_spi_id_t id, uint8_t instruction, uint32_t address, uint8_t dummy_bytes, uint8_t * p_data, uint16_t data_length)
功能说明	SPI主设备带指令和地址的轮询方式（同步）数据读取接口，主要用于对Flash/EEPROM/Display设备的读取操作
输入参数	• id：SPI主设备 • instruction：读取数据的指令 • address：待读取数据的地址 • dummy_bytes：设置SPI主设备从发送完数据读取地址到开始读取的间隔时间，具体的间隔时间由从设备的指标来确定。取值范围：0 ～ 4，单位：Byte • p_data：需要接收的数据buffer • data_length：需要读取的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	超时时间默认1s，定义请见DEFAULT_POLLING_WAIT_TIMEOUT_MS

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.12.4.6 app_spim_transmit_with_ia

表 2-222 app_spim_transmit_with_ia接口

函数原型	uint16_t app_spim_transmit_with_ia(app_spi_id_t id, uint8_t instruction, uint32_t address, uint8_t * p_data, uint16_t data_length)
功能说明	SPI master携带指令和地址的轮询方式（同步）数据发送接口，主要用于对Flash/EEPROM/Display设备进行数据写入操作
输入参数	• id：SPI主设备 • instruction：数据写入操作指令 • address：数据写入从设备的地址 • p_data：待写入数据的buffer • data_length：需要发送/写入的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.7 app_spi_transmit_sync

表 2-223 app_spi_transmit_sync接口

函数原型	uint16_t app_spi_transmit_sync(app_spi_id_t id, uint8_t *p_data, uint16_t size, uint32_t timeout)
功能说明	SPI轮询方式（同步）发送数据
输入参数	• id：SPI主设备/从设备 • p_data：需要发送的数据buffer指针 • size：需要发送的数据长度，以Byte为单位 • timeout：发送超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.8 app_spi_transmit_async

表 2-224 app_spi_transmit_async接口

函数原型	uint16_t app_spi_transmit_async(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI中断方式（异步）发送数据
输入参数	• id：SPI主设备/从设备 • p_data：需要发送的数据buffer指针 • size：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.12.4.9 app_spi_transmit_receive_sync

表 2-225 app_spi_transmit_receive_sync接口

函数原型	uint16_t app_spi_transmit_receive_sync(app_spi_id_t id, uint8_t *p_tx_data, uint8_t *p_rx_data, uint32_t size, uint32_t timeout)
功能说明	SPI以轮询方式（同步）进行全双工数据发送和接收
输入参数	• id：SPI主设备/从设备 • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • size：双工发送/接收的数据长度，以Byte为单位 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.10 app_spi_transmit_receive_async

表 2-226 app_spi_transmit_receive_async接口

函数原型	uint16_t app_spi_transmit_receive_async(app_spi_id_t id, uint8_t *p_tx_data, uint8_t *p_rx_data, uint32_t size)
功能说明	SPI以中断方式（异步）进行双工数据发送和接收
输入参数	• id：SPI主设备/从设备 • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • size：双工发送/接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.11 app_spi_read_eeprom_sync

表 2-227 app_spi_read_eeprom_sync接口

函数原型	uint16_t app_spi_read_eeprom_sync(app_spi_id_t id, uint8_t *p_tx_data, uint8_t *p_rx_data, uint32_t tx_size, uint32_t rx_size, uint32_t timeout)
功能说明	SPI轮询方式（同步）读取EEPROM特定内存地址数据，在读取之前按照EEPROM的协议格式把要读取的起始地址通过tx_data发送出去
输入参数	• id：SPI主设备 • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • tx_size：需要发送的数据长度，以Byte为单位 • rx_size：需要接收的数据长度，以Byte为单位 • timeout：读取超时时间
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义

版权所有 © 2025 深圳市汇顶科技股份有限公司

备注	

2.12.4.12 app_spi_read_eeprom_async

表 2-228 app_spi_read_eeprom_async接口

函数原型	uint16_t app_spi_read_eeprom_async(app_spi_id_t id, uint8_t *p_tx_data, uint8_t *p_rx_data, uint32_t tx_size, uint32_t rx_size)
功能说明	SPI中断方式（异步）读取EEPROM特定内存地址数据，在读取之前按照EEPROM的协议格式把要读取的起始地址通过tx_data发送出去
输入参数	• id：SPI 主设备 • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • tx_size：需要发送的数据长度，以Byte为单位 • rx_size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.13 app_spi_write_memory_async

表 2-229 app_spi_write_memory_async接口

函数原型	uint16_t app_spi_write_memory_async(app_spi_id_t id, uint8_t *p_cmd_data, uint8_t *p_tx_data, uint32_t cmd_size, uint32_t tx_size)
功能说明	SPI中断方式（异步）写入数据到Memory，在写入之前先发送命令，主要用于读写存储设备，比如Flash等
输入参数	• id：SPI主设备 • p_cmd_data：命令数据的buffer • p_tx_data：待写入数据的buffer • cmd_size：命令的长度，以Byte为单位 • size：需要写入的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.14 app_spi_read_memory_async

表 2-230 app_spi_read_memory_async接口

函数原型	uint16_t app_spi_read_memory_async(app_spi_id_t id, uint8_t *p_cmd_data, uint8_t *p_rx_data, uint32_t cmd_size, uint32_t rx_size)
功能说明	SPI中断方式（异步）读取Memory数据，在读取之前先发送命令，主要用于读写存储设备，比如Flash等
输入参数	• id：SPI主设备 • p_cmd_data：命令数据的buffer指针 • p_rx_data：读取数据的buffer指针

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• cmd_size：命令的长度，以Byte为单位 • size：需要读取的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.15 app_spi_get_handle

表 2-231 app_spi_get_handle接口

函数原型	spi_handle_t *app_spi_get_handle(app_spi_id_t id)
功能说明	获取SPI句柄
输入参数	id：SPI模块ID（见2.12.3.1 app_spi_id_t枚举）
返回值	SPI句柄指针
备注	

2.12.4.16 app_spi_transmit_high_speed_sync

表 2-232 app_spi_transmit_high_speed_sync接口

函数原型	uint16_t app_spi_transmit_high_speed_sync(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI中断方式（同步）高速发送数据，此函数关闭全局中断，避免高优先级中断的干扰
输入参数	• id：SPI 主设备/从设备 • p_data：需要发送数据的buffer • size：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.17 app_spi_receive_high_speed_sync

表 2-233 app_spi_receive_high_speed_sync接口

函数原型	uint16_t app_spi_receive_high_speed_sync(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI中断方式（同步）高速接收数据，此函数关闭全局中断，避免高优先级中断的干扰
输入参数	• id：SPI 主设备/从设备 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.4.18 app_spi_abort

表 2-234 app_spi_abort接口

函数原型	uint16_t app_spi_abort(app_spi_id_t id)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	SPI中止传输（发送/接收）
输入参数	id：SPI主设备/从设备
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	需要等到APP_SPI_EVT_ABORT事件，才表示中止传输完成

2.12.5 DMA接口函数

表 2-235 DMA接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_spi_dma_init	Y	Y	Y	Y	Y
app_spi_dma_deinit	Y	Y	Y	Y	Y
app_spim_dma_transmit_with_ia	N	Y	Y	Y	Y
app_spim_dma_receive_with_ia	N	Y	Y	Y	Y
app_spi_dma_receive_async	Y	Y	Y	Y	Y
app_spi_dma_transmit_async	Y	Y	Y	Y	Y
app_spi_dma_transmit_receive_async	Y	Y	Y	Y	Y
app_spi_dma_read_eeprom_async	Y	Y	Y	Y	Y
app_spi_dma_write_memory_async	Y	N	N	N	N
app_spi_dma_read_memory_async	Y	N	N	N	N
app_spi_dma_receive_high_speed_sync	Y	N	N	N	N
app_spi_dma_transmit_high_speed_sync	Y	N	N	N	N

2.12.5.1 app_spi_dma_init

表 2-236 app_spi_dma_init接口

函数原型	uint16_t app_spi_dma_init(app_spi_params_t *p_params)
功能说明	初始化SPI DMA模式
输入参数	p_params：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.2 app_spi_dma_deinit

表 2-237 app_spi_dma_deinit接口

函数原型	uint16_t app_spi_dma_deinit(app_spi_id_t id)
功能说明	反初始化SPI DMA模式
输入参数	id：SPI主设备/从设备

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.3 app_spim_dma_receive_with_ia

表 2-238 app_spim_dma_receive_with_ia接口

函数原型	uint16_t app_spim_dma_receive_with_ia(app_spi_id_t id, uint8_t instruction, uint32_t address, uint8_t dummy_bytes, uint8_t * p_data, uint16_t data_length)
功能说明	SPI主设备带指令和地址的数据读取接口，主要用于对Flash/EEPROM/Display设备的读取操作（DMA模式）
输入参数	• id：SPI主设备 • instruction：读取数据的指令 • address：待读取数据的地址 • dummy_bytes：设置SPI主设备从发送完数据读取地址到开始读取的间隔时间，给从设备进行数据就绪的时间。取值范围：0 ～ 4，单位：Byte • p_data：需要读取的数据buffer • data_length：需要读取的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.4 app_spim_dma_transmit_with_ia

表 2-239 app_spim_dma_transmit_with_ia接口

函数原型	uint16_t app_spim_dma_transmit_with_ia(app_spi_id_t id, uint8_t instruction, uint32_t address, uint8_t * p_data, uint16_t data_length)
功能说明	SPI主设备带指令和地址的数据写入接口，主要用于对Flash/EEPROM/Display设备进行数据写入操作（DMA模式）
输入参数	• id：SPI主设备 • instruction：数据写入操作指令 • address：数据写入从设备的地址 • p_data：待写入数据的buffer • data_length：需要发送/写入的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.5 app_spi_dma_receive_async

表 2-240 app_spi_dma_receive_async接口

函数原型	uint16_t app_spi_dma_receive_async(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI异步DMA方式接收数据

版权所有 © 2025 深圳市汇顶科技股份有限公司

输入参数	• id：SPI主设备/从设备 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.6 app_spi_dma_transmit_async

表 2-241 app_spi_dma_transmit_async接口

函数原型	uint16_t app_spi_dma_transmit_async(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI异步DMA方式发送数据
输入参数	• id：SPI 主设备/从设备 • p_data：需要发送数据的buffer • size：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.7 app_spi_dma_transmit_receive_async

表 2-242 app_spi_dma_transmit_receive_async接口

函数原型	uint16_t app_spi_dma_transmit_receive_async(app_spi_id_t id, uint8_t *p_tx_data, uint8_t *p_rx_data, uint32_t size)
功能说明	SPI全双工异步收发数据（DMA方式）
输入参数	• id：SPI主设备/从设备 • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • size：双工发送/接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.8 app_spi_dma_read_eeprom_async

表 2-243 app_spi_dma_read_eeprom_async接口

函数原型	uint16_t app_spi_dma_read_eeprom_async(app_spi_id_t id, uint8_t *p_tx_data, uint8_t *p_rx_data, uint32_t tx_size, uint32_t rx_size)
功能说明	SPI以DMA方式（异步）读取EEPROM特定内存地址数据，在读取之前按照EEPROM的协议格式把要读取的起始地址通过tx_data发送出去
输入参数	• id：SPI主设备 • p_tx_data：发送数据的buffer • p_rx_data：接收数据的buffer • tx_size：需要发送的数据长度，以Byte为单位

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• rx_size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.9 app_spi_dma_write_memory_async

表 2-244 app_spi_dma_write_memory_async接口

函数原型	uint16_t app_spi_dma_write_memory_async(app_spi_id_t id, uint8_t *p_cmd_data, uint8_t *p_tx_data, uint32_t cmd_size, uint32_t tx_size)
功能说明	SPI以DMA方式（异步）向Memory写入数据，在写入之前先发送命令，主要用于读写存储设备，比如Flash等
输入参数	• id：SPI主设备 • p_cmd_data：命令数据的buffer • p_tx_data：待写入数据的buffer • cmd_size：命令的长度，以Byte为单位 • size：需要写入的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.10 app_spi_dma_read_memory_async

表 2-245 app_spi_dma_read_memory_async接口

函数原型	uint16_t app_spi_dma_read_memory_async(app_spi_id_t id, uint8_t *p_cmd_data, uint8_t *p_rx_data, uint32_t cmd_size, uint32_t rx_size)
功能说明	SPI异步DMA方式读取Memory数据，在读取之前先发送命令，主要用于读写存储设备，比如Flash等
输入参数	• id：SPI主设备 • p_cmd_data：命令数据的buffer • p_rx_data：读取数据的buffer • cmd_size：命令的长度，以Byte为单位 • size：需要读取的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.11 app_spi_dma_transmit_high_speed_sync

表 2-246 app_spi_dma_transmit_high_speed_sync接口

函数原型	uint16_t app_spi_dma_transmit_high_speed_sync(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI同步高速DMA方式发送数据，此函数关闭全局中断，避免高优先级中断的干扰
输入参数	• id：SPI主设备/从设备 • p_data：需要发送数据的buffer • size：需要发送的数据长度，以Byte为单位

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.5.12 app_spi_dma_receive_high_speed_sync

表 2-247 app_spi_dma_receive_high_speed_sync接口

函数原型	uint16_t app_spi_dma_receive_high_speed_sync(app_spi_id_t id, uint8_t *p_data, uint16_t size)
功能说明	SPI同步高速DMA方式接收数据，此函数关闭全局中断，避免高优先级中断的干扰
输入参数	• id：SPI主设备/从设备 • p_data：接收数据的buffer • size：需要接收的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.12.6 示例工程

2.12.6.1 SPI读写Flash

示例实现SPI作为主设备用DMA方式对Flash设备进行读写，工程路径：SDK_Folder\projects\peripheral\spi\app_spi。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.12.6.1.1 流程图

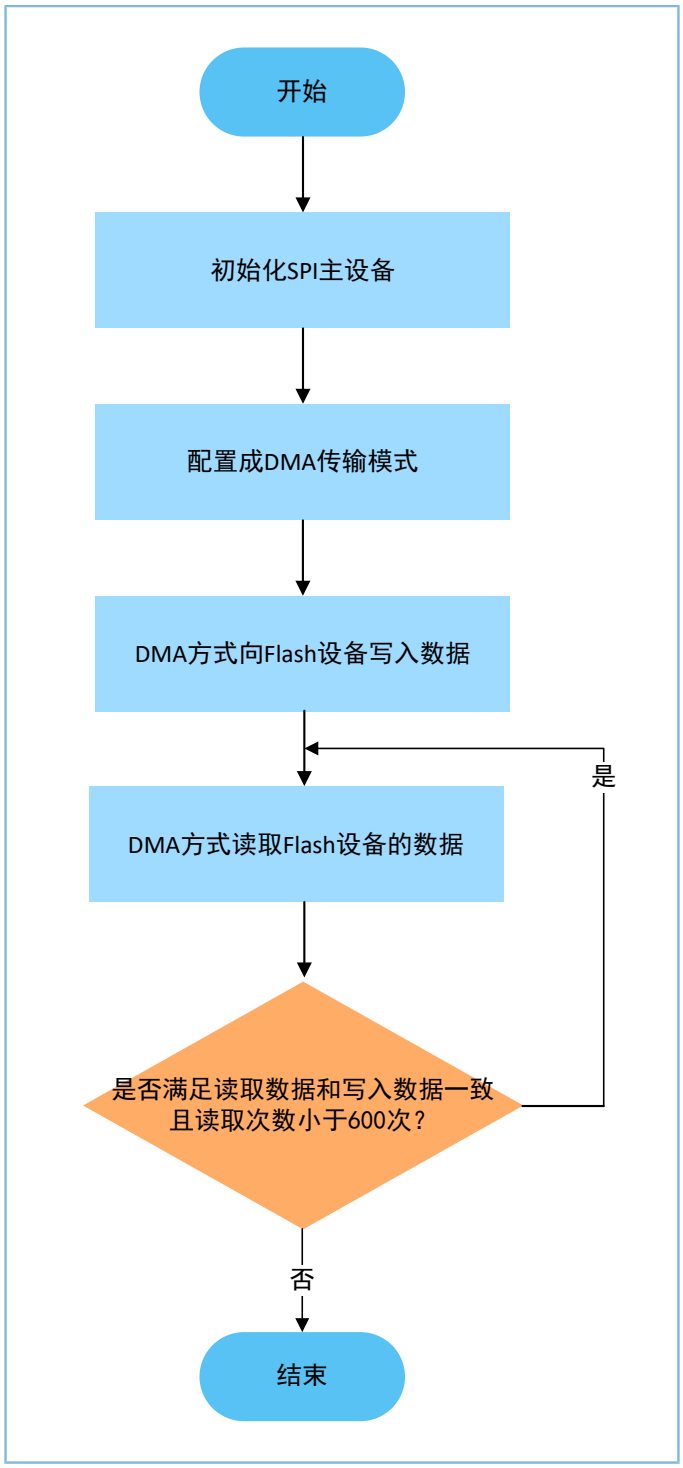

图 2-18 Flash读写示例流程图

2.12.6.1.2 文件调用关系

示例工程在APP_DRV层基础上实现*spi_flash.h*，面向对Flash的操作，*test_demo_cfg.h*基于*spi_flash.h*，实现示例设计。示例工程的文件调用关系如下图所示：

版权所有 © 2025 深圳市汇顶科技股份有限公司

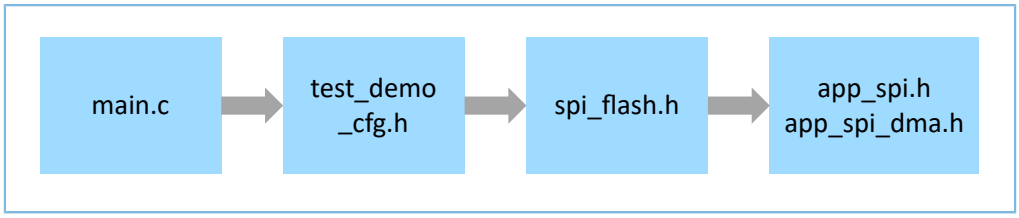

图 2-19 Flash读写示例文件调用流程

2.12.6.1.3 初始化参数

```
app_spi_params_t spi_params = {
    .id = APP_SPI_ID_MASTER,
    .pin_cfg = {
       .cs = {
           .type   = APP_SPIM_CS_IO_TYPE,
           .mux    = APP_SPIM_CS_PINMUX,
           .pin    = APP_SPIM_CS_PIN,
           .mode   = APP_IO_MODE_MUX,
           .pull   = APP_IO_PULLUP,
           .enable = APP_SPI_PIN_ENABLE,
       },
       .clk  = {
           .type   = APP_SPIM_CLK_IO_TYPE,
           .mux    = APP_SPIM_CLK_PINMUX,
           .pin    = APP_SPIM_CLK_PIN,
           .mode   = APP_IO_MODE_MUX,
           .pull   = APP_IO_PULLUP,
           .enable = APP_SPI_PIN_ENABLE,
       },
       .mosi = {
           .type   = APP_SPIM_MOSI_IO_TYPE,
           .mux    = APP_SPIM_MOSI_PINMUX,
           .pin    = APP_SPIM_MOSI_PIN,
           .mode   = APP_IO_MODE_MUX,
           .pull   = APP_IO_PULLUP,
           .enable = APP_SPI_PIN_ENABLE,
       },
       .miso = {
           .type   = APP_SPIM_MISO_IO_TYPE,
           .mux    = APP_SPIM_MISO_PINMUX,
           .pin    = APP_SPIM_MISO_PIN,
           .mode   = APP_IO_MODE_MUX,
           .pull   = APP_IO_PULLUP,
           .enable = APP_SPI_PIN_ENABLE,
       },
    },
    .dma_cfg = {
        .tx_dma_instance = DMA0,
        .rx_dma_instance = DMA0,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
        .tx_dma_channel  = DMA_Channel0,
        .rx_dma_channel  = DMA_Channel1,
#if (APP_DRIVER_CHIP_TYPE == APP_DRIVER_GR5526X)
        .wait_timeout_ms = SPI_WAIT_TIMEOUT_MS,
        .extend = 0,
#endif
    },
    .init = {
        .data_size      = SPI_DATASIZE_8BIT,
        .clock_polarity = SPI_POLARITY_LOW,
        .clock_phase    = SPI_PHASE_1EDGE,
        .baudrate_prescaler = SPI_CLOCK_PRESCALER,
        .ti_mode        = SPI_TIMODE_DISABLE,
        .slave_select   = SPI_SLAVE_SELECT_0,
#if (APP_DRIVER_CHIP_TYPE == APP_DRIVER_GR5526X)
        .rx_sample_delay = RX_SAMPLE_DELAY,
#endif
    },

    .is_soft_cs = SPI_SOFT_CS_MODE_DISABLE,
};
```

- id：SPI角色定义，APP_SPI_ID_MASTER主设备
- pin_cfg：SPI引脚定义，包含CS、CLK、MOSI、MISO引脚配置，配置复用模式，上下拉模式和引脚，具体见*board_SK.h*

  以GR5526为例，定义如下：

```
/*******SPI IO CONFIG**************************/
#define APP_SPIM_CS_PIN                     APP_IO_PIN_10
#define APP_SPIM_CLK_PIN                    APP_IO_PIN_5
#define APP_SPIM_MOSI_PIN                   APP_IO_PIN_6
#define APP_SPIM_MISO_PIN                   APP_IO_PIN_7
#define APP_SPIM_CS_IO_TYPE                 APP_IO_TYPE_GPIOB
#define APP_SPIM_CLK_IO_TYPE                APP_IO_TYPE_GPIOB
#define APP_SPIM_MOSI_IO_TYPE               APP_IO_TYPE_GPIOB
#define APP_SPIM_MISO_IO_TYPE               APP_IO_TYPE_GPIOB
#define APP_SPIM_CS_PINMUX                  APP_IO_MUX_1
#define APP_SPIM_CLK_PINMUX                 APP_IO_MUX_1
#define APP_SPIM_MOSI_PINMUX                APP_IO_MUX_1
#define APP_SPIM_MISO_PINMUX                APP_IO_MUX_1
```

- dma_cfg：DMA参数配置
  - dma_cfg.tx_dma_instance：SPI发送使用DMA0实例
  - dma_cfg.rx_dma_instance：SPI接收使用DMA0实例
  - dma_cfg.tx_dma_channel：SPI发送使用DMA0实例通道，选择通道0

版权所有 © 2025 深圳市汇顶科技股份有限公司

- dma_cfg.rx_dma_channel：SPI接收使用DMA0实例通道，选择通道1

- init：SPI传输参数
  - init.data_size：数据位宽SPI_DATASIZE_8BIT，8 bits
  - init.clock_polarity： 串行时钟有效极性SPI_POLARITY_LOW，CLK非活动状态时为低电平
  - init.clock_phase：位捕获的时钟活动边沿SPI_PHASE_1EDGE，在第一个时钟变化沿捕获数据
  - init.baudrate_prescaler：分频系数，SPI传输时钟等于系统时钟除以分频系数，定义如下：

```
#define SPI_CLOCK_PRESCALER             16u
```

  - init.ti_mode：定义是否使能TI模式，SPI_TIMODE_DISABLE不使能
  - init.slave_select：SPI_SLAVE_SELECT_0选择从设备0
  - init.rx_sample_delay：RX输入端口的采样延时，以系统外设时钟周期为单位
  - init.is_soft_cs：设置是否通过软件控制CS，SPI_SOFT_CS_MODE_DISABLE不使能软件控制CS

2.12.6.1.4 重要函数

- Flash初始化

```
spi_flash_init(APP_SPI_TYPE_DMA, FLASH_CLOCK_PRESCALER)
```

- 读取Flash的ID

```
flash_id = spi_flash_read_id();
```

- 擦除Flash数据

```
ret = spi_flash_erase(FLASH_PROGRAM_START_ADDR)
```

- 写入Flash数据

```
spi_flash_program(FLASH_PROGRAM_START_ADDR, write_buffer, FLASH_OPERATION_LENGTH)
```

- 读取Flash的数据

```
spi_flash_fast_read(FLASH_PROGRAM_START_ADDR, read_buffer, FLASH_OPERATION_LENGTH)
```

2.12.6.1.5 测试验证

1. 参考示例工程代码进行Flash连接。

2. 测试：

   连接开发板到PC端，打开串口调试助手，连接串口。打印如下，读取Flash ID成功，且写入Flash后，读取Flash设备数据和写入的一致。

```
Flash id is:0xb6017
[TEST Time:00:00] SPI flash read data verify success.
[11:14:34.214] [TEST Time:00:00] SPI flash read data verify success.
[11:14:34.741] [TEST Time:00:00] SPI flash read data verify success.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13 APP DSPI驱动

该驱动仅适用于GR5526和GR5x25。

2.13.1 宏定义

2.13.1.1 APP_DSPI_PIN_ENABLE

表 2-248 APP_DSPI_PIN_ENABLE宏定义

描述	值
DSPI引脚使能	1

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

2.13.1.2 APP_DSPI_PIN_DISABLE

表 2-249 APP_DSPI_PIN_DISABLE宏定义

描述	值
DSPI引脚不使能	0

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

2.13.2 结构体

2.13.2.1 app_dspi_pin_t

结构体定义如下：

表 2-250 app_dspi_pin_t结构体成员

成员	描述	取值
app_io_type_t type	设置IO类型	同2.2.1.2 app_io_evt_t中描述
app_io_mux_t mux	设置IO映射功能	参考2.2.1.1 app_io_init_t结构体，具体映射模式根据芯片Datasheet中的Pin Mux进行选择
uint32_t pin	设置IO引脚	参考2.2.1.1 app_io_init_t结构体，具体根据芯片Datasheet中DSPI资源进行选择
app_io_pull_t pull	设置IO上拉/下拉/浮空	该参数的取值可以是下列值中的任意一个： • APP_IO_NOPULL（浮空） • APP_IO_PULLUP（上拉） • APP_IO_PULLDOWN（下拉）
uin8_t enable	开启或关闭PIN	该参数的取值可以是下列值中的任意一个： • APP_DSPI_PIN_ENABLE（开启PIN） • APP_DSPI_PIN_DISABLE（关闭PIN）

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13.2.2 app_dspi_pin_cfg_t

结构体定义如下：

表 2-251 app_dspi_pin_cfg_t结构体成员

成员	描述	取值
app_dspi_pin_t cs	DSPI CS引脚配置	参考2.13.2.1 app_dspi_pin_t结构体
app_dspi_pin_t clk	DSPI CLOCK引脚配置	参考2.13.2.1 app_dspi_pin_t结构体
app_dspi_pin_t mosi	DSPI MOSI引脚配置	参考2.13.2.1 app_dspi_pin_t结构体
app_dspi_pin_t miso	DSPI MISO引脚配置	参考2.13.2.1 app_dspi_pin_t结构体
app_dspi_pin_t dcx	DSPI DCX引脚配置	参考2.13.2.1 app_dspi_pin_t结构体

详细信息请参考文件：`SDK_Folder\drivers\inc\app_dspi.h`。

2.13.2.3 dspi_init_t

结构体定义如下：

表 2-252 dspi_init_t结构体成员

成员	描述	取值
uint32_t data_size	DSPI数据宽度	该参数的取值可以是下列值中的任意一个： • DSPI_DATASIZE_04_BITS（4 bits） • DSPI_DATASIZE_05_BITS（5 bits） • DSPI_DATASIZE_06_BITS（6 bits） • DSPI_DATASIZE_07_BITS（7 bits） • DSPI_DATASIZE_08_BITS（8 bits） • DSPI_DATASIZE_09_BITS（9 bits） • DSPI_DATASIZE_10_BITS（10 bits） • DSPI_DATASIZE_11_BITS（11 bits） • DSPI_DATASIZE_12_BITS（12 bits） • DSPI_DATASIZE_13_BITS（13 bits） • DSPI_DATASIZE_14_BITS（14 bits） • DSPI_DATASIZE_15_BITS（15 bits） • DSPI_DATASIZE_16_BITS（16 bits） • DSPI_DATASIZE_17_BITS（17 bits） • DSPI_DATASIZE_18_BITS（18 bits） • DSPI_DATASIZE_19_BITS（19 bits） • DSPI_DATASIZE_20_BITS（20 bits） • DSPI_DATASIZE_21_BITS（21 bits） • DSPI_DATASIZE_22_BITS（22 bits） • DSPI_DATASIZE_23_BITS（23 bits） • DSPI_DATASIZE_24_BITS（24 bits） • DSPI_DATASIZE_25_BITS（25 bits） • DSPI_DATASIZE_26_BITS（26 bits）

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• DSPI_DATASIZE_27_BITS（27 bits） • DSPI_DATASIZE_28_BITS（28 bits） • DSPI_DATASIZE_29_BITS（29 bits） • DSPI_DATASIZE_30_BITS（30 bits） • DSPI_DATASIZE_31_BITS（31 bits） • DSPI_DATASIZE_32_BITS（32 bits）
uint32_t baud_rate	DSPI波特率预分频器值，该值将用于配置发送和接收SCK时钟	该参数的取值可以是下列值中的任意一个： • DSPI_BAUD_RATE_2P1PCLK（外设串行时钟频率/2） • DSPI_BAUD_RATE_4P1PCLK（外设串行时钟频率/4） • DSPI_BAUD_RATE_8P1PCLK（外设串行时钟频率/8） • DSPI_BAUD_RATE_16P1PCLK（外设串行时钟频率/16） • DSPI_BAUD_RATE_32P1PCLK（外设串行时钟频率/32） • DSPI_BAUD_RATE_64P1PCLK（外设串行时钟频率/64） • DSPI_BAUD_RATE_128P1PCLK（外设串行时钟频率/128） • DSPI_BAUD_RATE_256P1PCLK（外设串行时钟频率/256）
uint32_t dspi_mode	DSPI接口模式	该参数的取值可以是下列值中的任意一个： • DSPI_PROT_MODE_3W1L（3线1 lane） • DSPI_PROT_MODE_4W1L（4线1 lane） • DSPI_PROT_MODE_4W2L（4线2 lane）

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_dspi.h。

2.13.2.4 app_dspi_dma_cfg_t

结构体定义如下：

表 2-253 app_dspi_dma_cfg_t结构体成员

成员	描述	取值
dma_channel_t channel	DSPI DMA传输通道	同2.4.1.1 app_dma_params_t中通道描述

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

2.13.2.5 app_dspi_params_t

结构体定义如下：

表 2-254 app_dspi_params_t结构体成员

成员	描述	取值
app_dspi_pin_cfg_t pin_cfg	DSPI引脚功能设置	参考2.13.2.2 app_dspi_pin_cfg_t结构体
app_dspi_dma_cfg_t dma_cfg	DSPI传输方式选择	参考2.13.2.4 app_dspi_dma_cfg_t结构体
dspi_init_t init	DSPI初始化参数配置	参考2.13.2.3 dspi_init_t结构体
bool is_soft_cs	是否通过软件控制CS片选引脚	• true：软件控制片选引脚 • false：不通过软件控制片选引脚

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
dspi_env_t dspi_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

2.13.2.6 app_dspi_evt_t

结构体定义如下：

表 2-255 app_dspi_evt_t结构体成员

成员	描述	取值
app_dspi_evt_type_t type	串口事件类型	该参数的取值可以是下列值中的任意一个： • APP_DSPI_EVT_ERROR（DSPI 发送错误） • APP_DSPI_EVT_TX_CPLT（DSPI发送完成） • APP_DSPI_EVT_ABORT（DSPI中止传输）
union { uint32_t error_code; uint16_t size; }data;	• uint32_t error_code：发送事件返回的错误号 • uint16_t size：DSPI已发送的数据长度 说明： 若事件类型为传输错误，此处返回错误号； 若事件类型为发送完成，此处显示已传输的数据大小	DSPI错误号及对应含义如下： • HAL_DSPI_ERROR_NONE（无错误） • HAL_DSPI_ERROR_TIMEOUT（超时） • HAL_DSPI_ERROR_TRANSFER（传输错误） • HAL_DSPI_ERROR_DMA（DMA传输错误） • HAL_DSPI_ERROR_INVALID_PARAM（无效参数）

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

2.13.2.7 app_dspi_command_t

结构体定义如下：

表 2-256 app_dspi_command_t结构体成员

成员	描述	取值
uint32_t instruction	DSPI指令	需要发送的命令数据
uint32_t instruction_size	DSPI指令位宽	需要发送的命令数据位宽，一般取值为： • DSPI_INSTSIZE_08_BITS（8 bits） • DSPI_INSTSIZE_16_BITS（16 bits） • DSPI_INSTSIZE_32_BITS（32 bits）
uint32_t data_size	DSPI数据位宽	需要发送的数据位宽，一般取值为： • DSPI_INSTSIZE_08_BITS（8 bits） • DSPI_INSTSIZE_16_BITS（16 bits） • DSPI_INSTSIZE_32_BITS（32 bits）
uint32_t length	DSPI发送的数据长度，单位：字节	0x0 ～ 0xFFFFFFFF

版权所有 © 2025 深圳市汇顶科技股份有限公司

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_dspi.h。

2.13.3 枚举

2.13.3.1 app_dspi_evt_type_t

枚举定义如下：

表 2-257 app_dspi_evt_type_t枚举成员

成员	描述
APP_DSPI_EVT_ERROR = 0	DSPI发送错误事件
APP_DSPI_EVT_TX_CPLT = 1	DSPI发送完成事件
APP_DSPI_EVT_ABORT = 3	DSPI中止传输事件

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

2.13.3.2 app_dspi_state_t

枚举定义如下：

表 2-258 app_dspi_state_t枚举成员

成员	描述
APP_DSPI_INVALID = 0	DSPI不可用
APP_DSPI_ACTIVITY = 1	DSPI激活
APP_DSPI_SLEEP = 2	DSPI睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

2.13.3.3 app_dspi_dma_state_t

枚举定义如下：

表 2-259 app_dspi_dma_state_t枚举成员

成员	描述
APP_DSPI_DMA_INVALID	DSPI DMA不可用
APP_DSPI_DMA_ACTIVITY	DSPI DMA激活

详细信息请参考文件：SDK_Folder\drivers\inc\app_dspi.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13.4 接口函数

表 2-260 DSPI驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_dspi_init	N	Y	Y	N	N
app_dspi_deinit	N	Y	Y	N	N
app_dspi_command_transmit_sync	N	Y	Y	N	N
app_dspi_command_transmit_async	N	Y	Y	N	N
app_dspi_command_sync	N	Y	Y	N	N
app_dspi_command_async	N	Y	Y	N	N
app_dspi_transmit_sync	N	Y	Y	N	N
app_dspi_transmit_async	N	Y	Y	N	N
app_dspi_get_handle	N	Y	Y	N	N
app_dspi_config_mode	N	Y	Y	N	N
app_dspi_config_data_size	N	Y	Y	N	N
app_dspi_abort	N	Y	Y	N	N
app_dspi_abort_it	N	Y	Y	N	N

2.13.4.1 app_dspi_init

表 2-261 app_dspi_init接口

函数原型	uint16_t app_dspi_init(app_dspi_params_t *p_params, app_dspi_evt_handler_t evt_handler)
功能说明	初始化DSPI
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_dspi_evt_handler_t)(app_dspi_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理。

2.13.4.2 app_dspi_deinit

表 2-262 app_dspi_deinit接口

函数原型	uint16_t app_dspi_deinit(void)
功能说明	反初始化DSPI
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13.4.3 app_dspi_command_transmit_sync

表 2-263 app_dspi_command_transmit_sync接口

函数原型	uint16_t app_dspi_command_transmit_sync(app_dspi_command_t *p_cmd, uint8_t *p_data, uint32_t timeout)
功能说明	DSPI轮询方式（同步）发送数据，带控制命令封装
输入参数	• p_cmd：命令、数据参数的结构体指针 • p_data：需要发送的数据 • timeout：超时时间（单位：ms）。当传输数据时间超过该设定值，接口返回APP_DRV_ERR_TIMEOUT
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.4.4 app_dspi_command_transmit_async

表 2-264 app_dspi_command_transmit_async接口

函数原型	uint16_t app_dspi_command_transmit_async(app_dspi_command_t *p_cmd, uint8_t *p_data)
功能说明	DSPI中断方式（异步）发送数据，带控制命令封装
输入参数	• p_cmd：命令、数据参数的结构体指针 • p_data：需要发送的数据，以uint8_t类型数组给出
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.4.5 app_dspi_command_sync

表 2-265 app_dspi_command_sync接口

函数原型	uint16_t app_dspi_command_sync(app_dspi_command_t *p_cmd, uint32_t timeout)
功能说明	DSPI轮询方式（同步）发送命令
输入参数	• p_cmd：命令、数据参数的结构体指针 • timeout：超时时间（单位：ms），当接收数据时长超过该设定值，接口返回Timeout
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.4.6 app_dspi_command_async

表 2-266 app_dspi_command_async接口

函数原型	uint16_t app_dspi_command_async (app_dspi_command_t *p_cmd)
功能说明	DSPI中断方式（异步）发送命令
输入参数	p_cmd：命令、数据参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13.4.7 app_dspi_transmit_sync

表 2-267 app_dspi_transmit_sync接口

函数原型	uint16_t app_dspi_transmit_sync(uint8_t *p_data, uint32_t length, uint32_t timeout)
功能说明	DSPI轮询方式（同步）发送数据
输入参数	• p_data：需要发送的数据，以uint8_t类型数组给出 • length：需要发送的数据长度，以Byte为单位 • timeout：超时时间（单位：ms），当接收数据时长超过该设定值，接口返回Timeout
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.4.8 app_dspi_transmit_async

表 2-268 app_dspi_transmit_async接口

函数原型	uint16_t app_dspi_transmit_async(uint8_t *p_data, uint32_t length)
功能说明	DSPI中断方式（异步）发送数据
输入参数	• p_data：需要发送的数据，以uint8_t类型数组给出 • length：需要发送的数据长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.4.9 app_dspi_get_handle

表 2-269 app_dspi_get_handle接口

函数原型	dspi_handle_t *app_dspi_get_handle(void)
功能说明	获取当前DSPI句柄
返回值	dspi_handle_t：DSPI句柄结构体
备注	

2.13.4.10 app_dspi_config_mode

表 2-270 app_dspi_config_mode接口

函数原型	uint16_t app_dspi_config_mode(uint32_t mode)
功能说明	配置DSPI传输方式： • DSPI_PROT_MODE_3W1L • DSPI_PROT_MODE_4W1L • DSPI_PROT_MODE_4W2L
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13.4.11 app_dspi_config_data_size

表 2-271 app_dspi_config_data_size接口

函数原型	uint16_t app_dspi_config_data_size(uint32_t data_size)
功能说明	配置DSIP传输数据位宽
输入参数	data_size：需要传输的数据位宽，可取（DSPI_DATASIZE_04_BITS ～ DSPI_DATASIZE_32_BITS）参数中的一个
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.4.12 app_dspi_abort

表 2-272 app_dspi_abort接口

函数原型	uint16_t app_dspi_abort(void)
功能说明	DSPI中止传输（阻塞方式）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.4.13 app_dspi_abort_it

表 2-273 app_dspi_abort_it接口

函数原型	uint16_t app_dspi_abort_it(void)
功能说明	DSPI中止传输（非阻塞方式，可根据回调函数获取终止完成的状态）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.5 DMA接口函数

表 2-274 DMA接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_dspi_dma_init	N	Y	Y	N	N
app_dspi_dma_deinit	N	Y	Y	N	N
app_dspi_dma_command_transmit_async	N	Y	Y	N	N
app_dspi_dma_command_async	N	Y	Y	N	N
app_dspi_dma_transmit_async	N	Y	Y	N	N

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13.5.1 app_dspi_dma_init

表 2-275 app_dspi_dma_init接口

函数原型	uint16_t app_dspi_dma_init(app_dspi_params_t *p_params)
功能说明	初始化DSPI DMA模式
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.5.2 app_dspi_dma_deinit

表 2-276 app_dspi_dma_deinit接口

函数原型	uint16_t app_dspi_dma_deinit(void)
功能说明	反初始化DSPI DMA模式
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.5.3 app_dspi_dma_command_transmit_async

表 2-277 app_dspi_dma_command_transmit_async接口

函数原型	uint16_t app_dspi_dma_command_transmit_async(app_dspi_command_t *p_cmd, uint8_t *p_data)
功能说明	DSPI以DMA方式（异步）发送数据。先发送命令，后发送数据
输入参数	• p_cmd：存放发送命令的buffer • p_data：需要发送的数据，以uint8_t类型数组给出
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.5.4 app_dspi_dma_command_async

表 2-278 app_dspi_dma_command_async接口

函数原型	uint16_t app_dspi_dma_command_async(app_dspi_command_t *p_cmd)
功能说明	DSPI以DMA方式（异步）发送命令
输入参数	p_cmd：存放发送命令的buffer
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.13.5.5 app_dspi_dma_transmit_async

表 2-279 app_dspi_dma_transmit_async接口

函数原型	uint16_t app_dspi_dma_transmit_async(uint8_t *p_data, uint32_t length)
功能说明	DSPI以DMA方式（异步）发送数据
输入参数	• p_data：存放发送数据的buffer • length：待发送数据的长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.13.6 示例工程

2.13.6.1 DSPI刷屏测试

示例工程实现DSPI刷屏测试，定时刷新ST7789V屏幕，工程路径：SDK_Folder\projects\peripheral\dspi\app_dspi。该示例仅适用于GR5526、GR5525。

2.13.6.1.1 流程图

DSPI传输初始化参数给屏幕，实现屏幕初始化。最后将显示数据定时传输到屏幕，实现定时刷新屏幕。

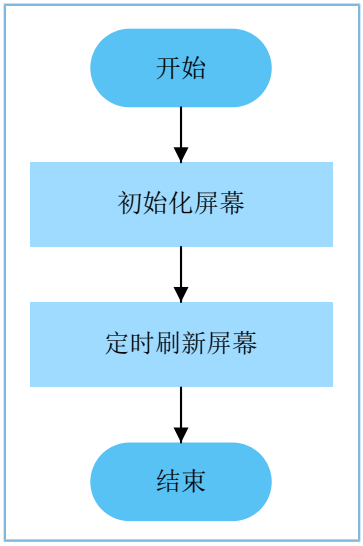

图 2-20 DSPI刷屏示例流程

2.13.6.1.2 初始化参数

```
app_dspi_params_t dspi_params = {
    .pin_cfg = {
        .cs = {
            .type    = APP_IO_TYPE_GPIOB,
            .mux     = APP_IO_MUX_2,
            .pin     = APP_IO_PIN_4,
            .pull    = APP_IO_NOPULL,
```

```
            .enable = APP_DSPI_PIN_ENABLE,
        },
        .clk = {
            .type   = APP_IO_TYPE_GPIOB,
            .mux    = APP_IO_MUX_2,
            .pin    = APP_IO_PIN_0,
            .pull   = APP_IO_NOPULL,
            .enable = APP_DSPI_PIN_ENABLE,
        },
        .mosi = {
            .type   = APP_IO_TYPE_GPIOB,
            .mux    = APP_IO_MUX_2,
            .pin    = APP_IO_PIN_1,
            .pull   = APP_IO_NOPULL,
            .enable = APP_DSPI_PIN_ENABLE,
        },
        .miso = {
            .type   = APP_IO_TYPE_GPIOB,
            .mux    = APP_IO_MUX_2,
            .pin    = APP_IO_PIN_2,
            .pull   = APP_IO_NOPULL,
            .enable = APP_DSPI_PIN_ENABLE,
        },
        .dcx = {
            .type   = APP_IO_TYPE_GPIOB,
            .mux    = APP_IO_MUX_2,
            .pin    = APP_IO_PIN_3,
            .pull   = APP_IO_NOPULL,
            .enable = APP_DSPI_PIN_ENABLE,
        },
    },
    .dma_cfg.channel = DMA_Channel0,
    .init = {
        .data_size = DSPI_DATASIZE_08_BITS,
        .baud_rate = DSPI_BAUD_RATE_2P1PCLK,
        . Init. baud_rate = DSPI_PROT_MODE_3W1L,
    },
};
```

- pin_cfg：DSPI引脚配置

  包含片选引脚CS、时钟引脚CLK、主出从入引脚MOSI、主入从出引脚MISO、控制引脚DCX，配置IO类型、PIN脚、上下拉模式和复用成DSPI引脚的复用模式。

- Init：DSPI传输参数配置

  - Init.data_size：数据传输位宽
  - Init.baud_rate：DSPI波特率预分频器值，选择2分频DSPI_BAUD_RATE_2P1PCLK，DSPI时钟等于串行时钟f_{PCLK}除以预分频器值

版权所有 © 2025 深圳市汇顶科技股份有限公司

- Init.Init.baud_rate：DSPI模式选用DSPI_PROT_MODE_3W1L，即3线1 lane

2.13.6.1.3 重要函数

- 初始化DSPI模块

```
ret = app_dspi_init(&params_t, app_dspi_evt_handler)
```

- 配置DMA传输方式

```
ret = app_dspi_dma_init(&params_t)
```

- 传输初始化参数

```
app_dspi_dma_command_transmit_async(&g_cmd, (uint8_t *)s_data_8)
```

- 传输控制命令

```
app_dspi_dma_command_async(&g_cmd)
```

- 初始化屏幕显示区域

```
app_dspi_display_coordinates(DSPI_ST7789V_X_START, DSPI_ST7789V_X_END,
                             DSPI_ST7789V_Y_START, DSPI_ST7789V_Y_END)
```

- 传输图像数据，刷新屏幕

```
app_dspi_display_write((uint8_t *)pixel, DSPI_POXEL_LEN * 2)
```

2.13.6.1.4 测试验证

- 硬件准备：LCD屏幕（型号：ST7789V）
- 测试步骤：
  1. 开发板连接LCD屏幕
  2. 下载运行程序，并打开串口调试助手查看打印日志
  3. 查看屏幕正常刷新显示，测试成功。

2.14 APP ADC驱动

2.14.1 结构体

2.14.1.1 app_adc_params_t

结构体定义如下：

表 2-280 app_adc_params_t结构体成员

成员	描述	取值
app_adc_pin_cfg_t pin_cfg	配置ADC模块输入通道	参考2.14.1.2 app_adc_pin_cfg_t结构体
app_adc_dma_cfg_t dma_cfg	ADC DMA参数配置	参考2.14.1.3 app_adc_dma_cfg_t结构体

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
adc_init_t init	ADC需要的初始化配置	参考2.14.1.5 adc_init_t结构体
adc_env_t adc_env	仅用于记录初始化设备的句柄、相关参数等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_adc.h。

2.14.1.2 app_adc_pin_cfg_t

结构体定义如下：

表 2-281 app_adc_pin_cfg_t结构体成员

成员	描述	取值
app_adc_pin_t channel_p	通道P引脚	参考2.14.1.4 app_adc_pin_t结构体
app_adc_pin_t channel_n	通道N引脚	参考2.14.1.4 app_adc_pin_t结构体
app_adc_pin_t extern_ref	外部参考源引脚	参考2.14.1.4 app_adc_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_adc.h。

2.14.1.3 app_adc_dma_cfg_t

结构体定义如下：

表 2-282 app_adc_dma_cfg_t结构体成员

成员	描述	取值
dma_regs_t * dma_instance	ADC模块ID	参考2.14.1.1 app_adc_params_t中DMA支持的外设
dma_channel_t dma_channel	ADC配置DMA通道	参考2.14.1.1 app_adc_params_t中通道描述

详细信息请参考文件：SDK_Folder\drivers\inc\app_adc.h。

2.14.1.4 app_adc_pin_t

结构体定义如下：

表 2-283 app_adc_pin_t结构体成员

成员	描述	取值
app_io_type_t type	IO类型	APP_IO_TYPE_MSIO
app_io_mux_t mux	设置IO映射功能	APP_IO_MUX
uint32_t pin	IO引脚	GR551x： • APP_IO_PIN0～ APP_IO_PIN4 GR5526/GR5x25/GR533x/GR5405：

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• APP_IO_PIN0～ APP_IO_PIN7

详细信息请参考文件：SDK_Folder\drivers\inc\app_adc.h。

2.14.1.5 adc_init_t

结构体定义如下：

表 2-284 adc_init_t结构体成员

成员	描述	取值
uint32_t channel_p	配置通道P映射	GR551x取值选项： • ADC_INPUT_SRC_IO0 • ADC_INPUT_SRC_IO1 • ADC_INPUT_SRC_IO2 • ADC_INPUT_SRC_IO3 • ADC_INPUT_SRC_IO4 • ADC_INPUT_SRC_TMP • ADC_INPUT_SRC_BAT • ADC_INPUT_SRC_REF GR5526/GR5x25/GR533x/GR5405取值选项： • ADC_INPUT_SRC_IO0 • ADC_INPUT_SRC_IO1 • ADC_INPUT_SRC_IO2 • ADC_INPUT_SRC_IO3 • ADC_INPUT_SRC_IO4 • ADC_INPUT_SRC_IO5 • ADC_INPUT_SRC_IO6 • ADC_INPUT_SRC_IO7 • ADC_INPUT_SRC_REF **说明：** 与通道P配置结构体2.14.1.4 app_adc_pin_t的pin引脚对应，例如：通道P配置为MSIO0，则该参数为ADC_INPUT_SRC_IO0。
uint32_t channel_n	配置通道N映射	GR551x取值选项： • ADC_INPUT_SRC_IO0 • ADC_INPUT_SRC_IO1 • ADC_INPUT_SRC_IO2 • ADC_INPUT_SRC_IO3 • ADC_INPUT_SRC_IO4 • ADC_INPUT_SRC_TMP • ADC_INPUT_SRC_BAT • ADC_INPUT_SRC_REF GR5526/GR5x25/GR533x/GR5405取值选项：

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• ADC_INPUT_SRC_IO0 • ADC_INPUT_SRC_IO1 • ADC_INPUT_SRC_IO2 • ADC_INPUT_SRC_IO3 • ADC_INPUT_SRC_IO4 • ADC_INPUT_SRC_IO5 • ADC_INPUT_SRC_IO6 • ADC_INPUT_SRC_IO7 • ADC_INPUT_SRC_TMP • ADC_INPUT_SRC_BAT • ADC_INPUT_SRC_REF **说明：** ADC_INPUT_SRC_BAT表示测量芯片的供电电压，ADC_INPUT_SRC_TMP表示测量芯片内部温度，除此之外该参数与通道N配置结构体2.14.1.4 app_adc_pin_t的pin引脚对应，例如：通道N配置为MSIO0，则该参数为ADC_INPUT_SRC_IO0。
uint32_t input_mode	输入模式设置	该参数的取值可以是下列值中的任意一个： • ADC_INPUT_SINGLE（单端采样） • ADC_INPUT_DIFFERENTIAL（差分采样）
uint32_t ref_source	参考源设置	该参数的取值可以是下列值中的任意一个： • ADC_REF_SRC_BUF_INT（内部参考源） • ADC_REF_SRC_IO0 • ADC_REF_SRC_IO1 • ADC_REF_SRC_IO2 • ADC_REF_SRC_IO3
uint32_t ref_value	内部参考源参考电压设置（外部参考，该参数无意义）	该参数的取值可以是下列值中的任意一个： • ADC_REF_VALUE_0P8（0.85 V） • ADC_REF_VALUE_1P2（1.28V） • ADC_REF_VALUE_1P6（1.6 V） • ADC_REF_VALUE_2P0（2.0 V，GR551x/GR533x/GR5405不适用）
uint32_t clock	ADC时钟设置	该参数的取值可以是下列值中的任意一个： GR5526/GR5x25/GR533x/GR5405： • ADC_CLK_16M • ADC_CLK_8M • ADC_CLK_4M • ADC_CLK_2M • ADC_CLK_1M • ADC_CLK_16K • ADC_CLK_8K

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• ADC_CLK_4K • ADC_CLK_NONE（关闭时钟） GR551x： • ADC_CLK_16M • ADC_CLK_1P6M • ADC_CLK_8M • ADC_CLK_4M • ADC_CLK_2M • ADC_CLK_1M

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_ll_adc.h
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_ll_adc.h
- GR5405：SDK_Folder\hal_drv\include\ll_adc.h

2.14.2 枚举

2.14.2.1 app_adc_evt_t

枚举定义如下：

表 2-285 app_adc_evt_t枚举成员

成员	描述	取值
app_adc_evt_type_t type	ADC事件结构体	APP_ADC_EVT_CONV_CPLT转换完成

详细信息请参考文件：SDK_Folder\drivers\inc\app_adc.h。

2.14.2.2 app_adc_sample_node_t

说明:

该枚举仅适用于GR551x。

枚举定义如下：

表 2-286 app_adc_samle_node_t枚举成员

成员	描述	取值
uint32_t channel	ADC采样通道	该参数的取值可以是下列值中的任意一个： • ADC_INPUT_SRC_IO0 • ADC_INPUT_SRC_IO1 • ADC_INPUT_SRC_IO2 • ADC_INPUT_SRC_IO3 • ADC_INPUT_SRC_TMP

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• ADC_INPUT_SRC_BAT • ADC_INPUT_SRC_REF
uint16_t *p_buf	采样得到的原始数据，由用户分配空间。	无
uint32_t len	采样的原始数据个数，由用户指定个数。	无
struct link_node *next	用于多通道采样的输入参数，采完一个通道后，自动采样下一个节点对应的通道。	无

详细信息请参考文件：SDK_Folder\drivers\inc\app_adc.h。

2.14.2.3 app_adc_evt_type_t

枚举定义如下：

表 2-287 app_adc_evt_type_t枚举成员

成员	描述
APP_ADC_EVT_CONV_CPLT	ADC转换完成

2.14.2.4 app_adc_dma_state_t

枚举定义如下：

表 2-288 app_adc_dma_state_t枚举成员

成员	描述
APP_ADC_DMA_INVALID	ADC DMA模式不可用
APP_ADC_DMA_ACTIVITY	ADC DMA模式激活

2.14.2.5 app_adc_state_t

枚举定义如下：

表 2-289 app_adc_state_t枚举成员

成员	描述
APP_ADC_INVALID = 0	ADC不可用
APP_ADC_ACTIVITY	ADC激活
APP_ADC_SLEEP	ADC睡眠

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.14.3 接口函数

表 2-290 ADC驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_adc_init	Y	Y	Y	Y	Y
app_adc_deinit	Y	Y	Y	Y	Y
app_adc_conversion_sync	Y	Y	Y	Y	Y
app_adc_conversion_async	Y	Y	Y	Y	Y
app_adc_voltage_intern	Y	Y	Y	Y	Y
app_adc_voltage_extern	Y	Y	Y	Y	Y
app_adc_temperature_conv	N	Y	Y	Y	Y
app_adc_vbat_conv	N	Y	Y	Y	Y
app_adc_get_handle	Y	Y	Y	Y	Y
app_adc_multi_channel_conversion_async	Y	N	N	N	N

2.14.3.1 app_adc_init

表 2-291 app_adc_init接口

函数原型	uint16_t app_adc_init(app_adc_params_t *p_params, app_adc_evt_handler_t evt_handler)
功能说明	初始化ADC
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_adc_evt_handler_t)(app_adc_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理。

2.14.3.2 app_adc_deinit

表 2-292 app_adc_deinit接口

函数原型	uint16_t app_adc_deinit(void)
功能说明	反初始化ADC
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.14.3.3 app_adc_conversion_sync

表 2-293 app_adc_conversion_sync接口

函数原型	uint16_t app_adc_conversion_sync(uint16_t *p_data, uint32_t length, uint32_t timeout)
功能说明	开启ADC同步转换
输入参数	• p_data：ADC数据的buffer • length：采样数据长度 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.14.3.4 app_adc_conversion_async

表 2-294 app_adc_conversion_async接口

函数原型	uint16_t app_adc_conversion_async(uint16_t *p_data, uint32_t length)
功能说明	开启ADC异步转换
输入参数	• p_data：ADC数据的buffer • length：采样数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.14.3.5 app_adc_voltage_intern

表 2-295 app_adc_voltage_intern接口

函数原型	uint16_t app_adc_voltage_intern(uint16_t *inbuf, double *outbuf, uint32_t buflen)
功能说明	将使用内部参考源的ADC数值转换为电压值
输入参数	• inbuf：ADC数据的buffer • outbuf：电压值的buffer • buflen：转换的数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	参考源采用内部参考时调用该函数

2.14.3.6 app_adc_voltage_extern

表 2-296 app_adc_voltage_extern接口

函数原型	uint16_t app_adc_voltage_extern(double ref, uint16_t *inbuf, double *outbuf, uint32_t buflen)
功能说明	将使用外部参考源的ADC数值转化为电压值
输入参数	• ref：外部参考电压的大小 • inbuf：ADC数据的buffer • outbuf：电压值的buffer • buflen：转换的数据长度

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	参考源采用外部参考时使用

2.14.3.7 app_adc_temperature_conv

表 2-297 app_adc_temperature_conv接口

函数原型	uint16_t app_adc_temperature_conv(uint16_t *inbuf, double *outbuf, uint32_t buflen)
功能说明	ADC数值转换为内部芯片温度
输入参数	• inbuf：ADC数据的buffer • outbuf：内部芯片温度数据的buffer • buflen：转换的数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.14.3.8 app_adc_vbat_conv

表 2-298 app_adc_vbat_conv接口

函数原型	uint16_t app_adc_vbat_conv(uint16_t *inbuf, double *outbuf, double *outbuf, uint32_t buflen)
功能说明	ADC数值转换为电池电压
输入参数	• inbuf：ADC数据的buffer • outbuf：电池电压数据的buffer • buflen：转换的数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.14.3.9 app_adc_get_handle

表 2-299 app_adc_get_handle接口

函数原型	adc_handle_t *app_adc_get_handle(void)
功能说明	获取ADC句柄
输入参数	
返回值	adc句柄，详见 • GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_ll_adc.h • GR533x：SDK_Folder\drivers\inc\hal\gr533x_ll_adc.h • GR5405：SDK_Folder\hal_drv\include\ll_adc.h
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.14.3.10 app_adc_multi_channel_conversion_async

表 2-300 app_adc_multi_channel_conversion_async接口

函数原型	uint16_t app_adc_multi_channel_conversion_async(app_adc_samle_node_t *p_begin_node, uint32_t total_nodes)
功能说明	开启ADC多通道异步转换
输入参数	• p_begin_node：多通道采样的通道链表指针 • total_nodes：采样通道数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.14.4 DMA接口函数

表 2-301 接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_adc_dma_init	Y	Y	Y	Y	Y
app_adc_dma_deinit	Y	Y	Y	Y	Y
app_adc_dma_conversion_async	Y	Y	Y	Y	Y

2.14.4.1 app_adc_dma_init

表 2-302 app_adc_dma_init接口

函数原型	uint16_t app_adc_dma_init(app_adc_params_t *p_params)
功能说明	初始化DMA方式传输的ADC模块
输入参数	p_params：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.14.4.2 app_adc_dma_deinit

表 2-303 app_adc_dma_deinit接口

函数原型	uint16_t app_adc_dma_deinit(void)
功能说明	反初始化DMA方式传输的ADC模块
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.14.4.3 app_adc_dma_conversion_async

表 2-304 app_adc_dma_conversion_async接口

函数原型	uint16_t app_adc_dma_conversion_async(uint16_t *p_data, uint32_t length)
功能说明	开启ADC异步DMA转换
输入参数	• p_data：ADC数值的buffer • length：采样数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.14.5 示例工程

2.14.5.1 ADC实现DMA方式对输入电压测量

ADC有两种采样模式：单端采样和差分采样。

- 单端采样时，用户需将有效信号连接至MSIO其中的一个引脚，单端输入信号幅值范围在0 ～ 2*Vref，且不大于VBAT。
- 差分采样时，用户需输入差分信号至MSIO中的两个引脚，差分输入信号幅值范围在-2*Vref ～ +2*Vref，共模值大于0.8 V。

示例工程实现ADC单端DMA采样，工程路径：SDK_Folder\projects\peripheral\adc\adc。

2.14.5.1.1 流程图

示例工程初始化ADC模块，采样单端采样，仅通道N为有效输入信号，配置通道N为MSIO1；用户将有效信号接入MSIO1引脚，开启ADC采样后将采集MSIO1有效信号，最后将有效信号的值转换为内部电压值并打印。

版权所有 © 2025 深圳市汇顶科技股份有限公司

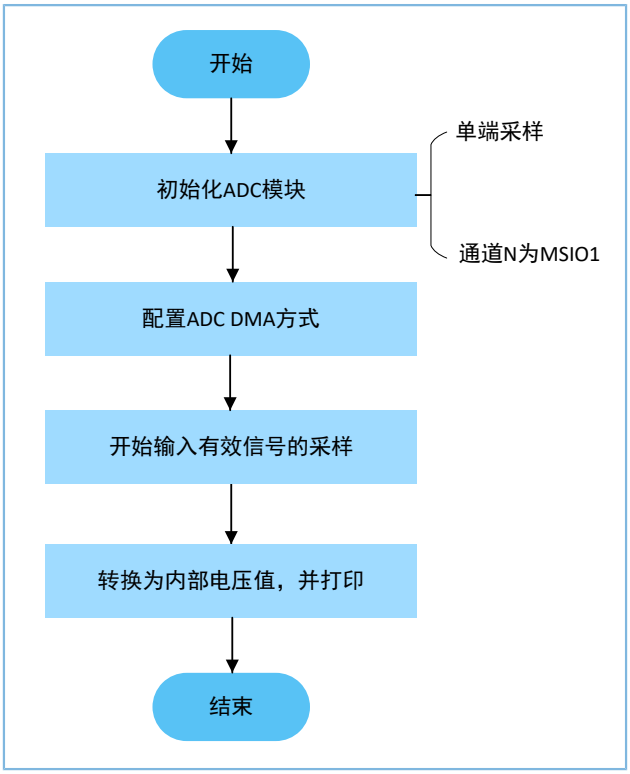

图 2-21 ADC输入电压测量流程

2.14.5.1.2 初始化参数

```
app_adc_params_t adc_params = {
    .pin_cfg = {
        .channel_p = {
            .type = APP_IO_TYPE_MSIO,
            .mux  = APP_IO_MUX_7,
            .pin  = APP_ADC_P_INPUT_PIN,
        },
        .channel_n = {
            .type = APP_IO_TYPE_MSIO,
            .mux  = APP_IO_MUX_7,
            .pin  = APP_ADC_N_INPUT_PIN,
        },
    },
    .dma_cfg = {
        .dma_instance = DMA0,
        .dma_channel  = DMA_Channel0,
    },
    .init = {
        .channel_p  = ADC_INPUT_SRC_IO0,
        .channel_n  = ADC_INPUT_SRC_IO1,
        .input_mode = ADC_INPUT_SINGLE,
        .ref_source = ADC_REF_SRC_BUF_INT,
        .ref_value  = ADC_REF_VALUE_1P6,
        .clock      = ADC_CLK_1M,
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
    },
};
```

- pin_cfg：引脚配置
  - pin_cfg.channel_p：ADC通道p引脚配置
  - pin_cfg.channel_n：ADC通道n引脚配置
- dma_cfg：DMA参数配置
  - dma_cfg.dma_instance：DMA实例使用DMA0
  - dma_cfg.dma_channel：DMA实例使用DMA0通道0
  - init.channel_p：配置通道P映射，选择ADC_INPUT_SRC_IO0（和通道P引脚配置对应，MSIO0对应ADC_INPUT_SRC_IO0）
  - init.channel_n：配置通道N映射，选择ADC_INPUT_SRC_IO1（和通道N引脚配置对应，MSIO1对应ADC_INPUT_SRC_IO1）
- init：ADC采集参数配置
  - init.input_mode：采样模式选择ADC_INPUT_SINGLE（单端采样）
  - init.ref_source：参考源选择ADC_REF_SRC_BUF_INT（选择内部参考源）
  - init.ref_value：内部参考源参考电压：ADC_REF_VALUE_1P6（选择1.6 V，输入量程为0 V～3.2 V）
  - init.clock：ADC时钟设置：ADC_CLK_1M（1 MHz）

2.14.5.1.3 重要函数

- 初始化ADC

```
app_adc_init(&adc_params, app_adc_evt_handler)
```

- 配置DMA方式

```
app_adc_dma_init(&adc_params)
```

- 开启ADC DMA采集

```
app_adc_dma_conversion_async(conversion, TEST_CONV_LENGTH)
```

  - conversion：采集的ADC数值集合
  - TEST_CONV_LENGTH：采集ADC数据的个数

- 转换成内部电压

```
app_adc_voltage_intern(conversion, voltage, TEST_CONV_LENGTH)
```

  - voltage：转换后电压值集合
  - TEST_CONV_LENGTH：数据个数

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.14.5.1.4 测试验证

外部输入电压到MSIO1，连接开发板到PC端，打开串口助手，连接串口，查看串口日志，观察串口打印，ADC 测量的电压等于输入电压，表明ADC采集电压准确。

电源输出0.5 V电压，将电源输出接到MSIO1，连接开发板到PC端，串口打印如下，测量电压为0.5 V左右。

```
Start single sampling.
DMA conversion is done.
Conversion value:
0.503V  0.506V  0.498V  0.490V  0.469V  0.498V  0.503V  0.501V  0.506V  0.502V  0.505V
 0.495V  0.501V  0.497V  0.500V  0.495V  0.496V  0.492V  0.484V  0.504V  0.490V  0.501V
 0.489V  0.493V  0.495V  0.499V  0.509V  0.498V  0.502V  0.501V  0.497V  0.494V  0.500V
 0.510V  0.500V  0.485V  0.506V  0.499V  0.493V  0.502V  0.498V  0.499V  0.497V  0.503V
 0.496V  0.502V  0.503V  0.498V  0.496V  0.503V  0.501V  0.500V  0.495V  0.504V  0.498V
 0.503V  0.497V  0.494V  0.506V  0.500V  0.493V  0.501V  0.490V  0.500V  0.498V  0.493V
 0.501V  0.486V  0.495V  0.495V  0.493V  0.506V  0.501V  0.491V  0.494V  0.494V  0.496V
 0.498V  0.498V  0.492V  0.499V  0.492V  0.498V  0.495V  0.495V  0.497V  0.496V  0.508V
 0.506V  0.522V  0.506V  0.500V  0.501V  0.501V  0.495V  0.500V  0.500V  0.502V  0.493V
 0.499V  0.497V  0.492V  0.497V  0.503V  0.498V  0.492V  0.492V  0.495V  0.492V  0.492V
 0.489V  0.495V  0.498V  0.492V  0.492V  0.493V  0.489V  0.491V  0.496V  0.489V  0.500V
 0.501V  0.501V  0.506V  0.505V  0.500V  0.494V  0.505V
```

2.14.5.2 ADC电池电压的测量

工程路径：SDK_Folder\projects\peripheral\adc\adc。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.14.5.2.1 流程图

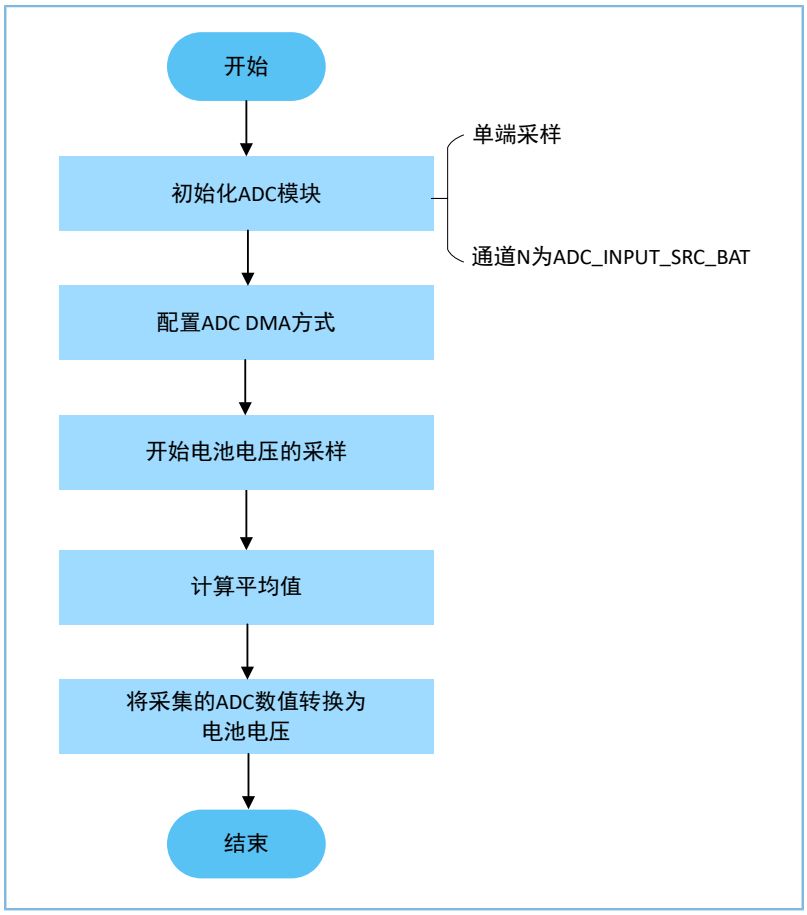

图 2-22 ADC电池电压测量流程图

ADC能实现电池电压的测量，通道应该选择ADC_INPUT_SRC_BAT，开启采样求取采样平均值，将平均采样值转换成电池电压。

2.14.5.2.2 初始化参数

实现对电池电压的采集，主要修改如下参数，其它参数参考2.14.5.1 ADC实现DMA方式对输入电压测量。

```
adc_params_t.init.channel_n = ADC_INPUT_SRC_BAT;
adc_params_t.init.ref_value = ADC_REF_VALUE_0P8;
```

- init.channel_n：ADC_INPUT_SRC_BAT（通道为BAT，测量的信号来自内部的分压电路）
- init.ref_value：ADC_REF_VALUE_0P8（0.85 V）

可检测的VBAT范围请参考对应芯片系列Datasheet，当选择ADC的通道为BAT时，测量的信号来自内部的分压电路（确保输入在0 ～ 1.6 V），所以参考电压应选择0.85 V，以提升测量精度。

2.14.5.2.3 重要函数

采集值转换成电池电压：

```
app_adc_vbat_conv((uint16_t *)&aver, &vbat, 1)
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.14.5.2.4 测试验证

连接开发板到PC端，打开串口助手，连接串口，查看串口日志，串口将打印电池电压值。

```
Start Vbattery sampling.
DMA conversion is done.
Average Vbattery cocdes = 4000
Vbattery= 3.307 V
```

2.14.5.3 ADC实现DMA方式对芯片内部温度测量

工程路径：SDK_Folder\projects\peripheral\adc\adc。

2.14.5.3.1 流程图

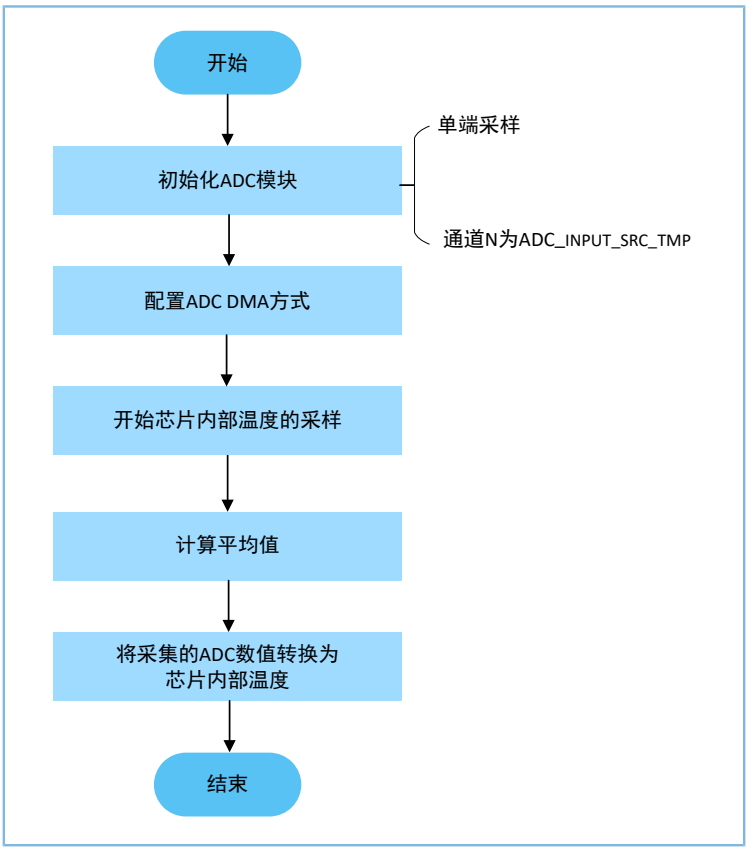

图 2-23 ADC芯片内部温度测量流程

ADC能实现芯片内部温度的测量，通道应选择ADC_INPUT_SRC_TMP，开启采样求取采样平均值，将平均采样值转换成芯片内部温度。

2.14.5.3.2 初始化参数

实现对芯片内部温度的采集，主要修改如下参数，其他参数参考2.14.5.1 ADC实现DMA方式对输入电压测量。

```
adc_params_t.init.channel_n = ADC_INPUT_SRC_TMP;
adc_params_t.init.ref_value = ADC_REF_VALUE_0P8;
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.channel_n：ADC_INPUT_SRC_TMP（通道为TMP，测量的信号来自内部的分压电路）
- init.ref_value：ADC_REF_VALUE_0P8（0.85 V）

2.14.5.3.3 重要函数

采样值转换成芯片内部温度：

```
app_adc_temperature_conv((uint16_t *)&aver, &temp, 1)
```

2.14.5.3.4 测试验证

连接开发板到PC端，打开串口助手，连接串口，查看串口日志，串口将打印芯片内部温度。

```
Start temperature sampling.
DMA conversion is done.
Average Temperature cocdes = 1594
Temperature value= 26.6 C
```

2.15 APP COMP驱动

2.15.1 结构体

2.15.1.1 app_comp_params_t

结构体定义如下：

表 2-305 app_comp_params_t结构体成员

成员	描述	取值
app_comp_pin_cfg_t pin_cfg	对COMP模拟输入引脚input和模拟参考源引脚Vref进行配置	参考2.15.1.2 app_comp_pin_t结构体
comp_init_t init	COMP需要的初始化配置	参考2.15.1.3 comp_init_t结构体
comp_env_t comp_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_comp.h。

2.15.1.2 app_comp_pin_t

结构体定义如下：

表 2-306 app_comp_pin_t结构体成员

成员	描述	取值
app_io_type_t type	IO类型	可配置类型： APP_IO_TYPE_MSIO
app_io_mux_t mux	设置IO映射功能	APP_IO_MUX

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
uint32_t pin	IO引脚	• GR551x取值选项： APP_IO_PIN0～ APP_IO_PIN4 • GR5526/GR5x25/GR533x/GR5405取值选项： APP_IO_PIN0～ APP_IO_PIN7

详细信息请参考文件：SDK_Folder\drivers\inc\app_comp.h。

2.15.1.3 comp_init_t

结构体定义如下：

表 2-307 comp_init_t结构体成员

成员	描述	取值
uint32_t input_source	COMP输入引脚	GR551x取值选项： • COMP_INPUT_SRC_IO0 • COMP_INPUT_SRC_IO1 • COMP_INPUT_SRC_IO2 • COMP_INPUT_SRC_IO3 • COMP_INPUT_SRC_IO4 对应引脚为MSIO0 ～ MSIO4 GR5526/GR5x25/GR533x/GR5405取值选项： • COMP_INPUT_SRC_IO0 • COMP_INPUT_SRC_IO1 • COMP_INPUT_SRC_IO2 • COMP_INPUT_SRC_IO3 • COMP_INPUT_SRC_IO4 • COMP_INPUT_SRC_IO5 • COMP_INPUT_SRC_IO6 • COMP_INPUT_SRC_IO7 • COMP_INPUT_SRC_VBAT • COMP_INPUT_SRC_VREF 对应引脚为MSIO0 ～ MSIO7
uint32_t ref_source	COMP参考源	GR551x取值选项： • COMP_REF_SRC_IO0 • COMP_REF_SRC_IO1 • COMP_REF_SRC_IO2 • COMP_REF_SRC_IO3 • COMP_REF_SRC_IO4 • COMP_REF_SRC_VBAT（电池电压） • COMP_REF_SRC_VREF（芯片内部参考源） GR5526/GR5x25/GR533x/GR5405取值选项： • COMP_REF_SRC_IO0 • COMP_REF_SRC_IO1

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• COMP_REF_SRC_IO2 • COMP_REF_SRC_IO3 • COMP_REF_SRC_IO4 • COMP_REF_SRC_IO5 • COMP_REF_SRC_IO6 • COMP_REF_SRC_IO7 • COMP_REF_SRC_VBAT（电池） • COMP_REF_SRC_VREF（内部参考）
uint32_t ref_value	COMP参考输入电压值	GR551x取值选项： • 外部参考源档位：无 • 电池参考源档位：0 ～ 7 • 内部参考源档位：0 ～ 63 GR5526/GR5x25/GR533x/GR5405取值选项： • 外部参考源档位：无 • 电池参考源档位：0 ～ 7 • 内部参考源档位：0 ～ 255 **说明：** 参考源为内部参考时，实际参考电压：GR551x为30 mV*档位，GR5526/GR5x25/GR533x/GR5405为7.45 mV*档位；参考源为电池时，实际参考电压为VBAT*[(档位+1)/10]
uint32_t hyst	迟滞（GR551x不适用）	该参数的取值可以是下列值中的任意一个，用或运算进行多个选择： • COMP_HYST_POSITIVE（开启正向迟滞，当输入电压大于Vref + Vhyst_positive时，触发上升沿中断），具体取值参考芯片Datasheet • COMP_HYST_NEGATIVE（开启逆向迟滞，在输入电压小于Vref – Vhyst_negative时，触发下降沿中断），具体取值参考芯片Datasheet **说明：** 迟滞和开启逆向迟滞总是开启，用户无需关心此参数输入
uint32_t edge	中断触发方式（GR551x默认只有上升沿触发）	该参数的取值可以是下列值中的任意一个： • COMP_WAKEUP_EDGE_FALLING（输入电压低于参考源电压时触发中断） • COMP_WAKEUP_EDGE_RISING（输入电压高于参考源电压时触发中断） • COMP_WAKEUP_EDGE_BOTH（输入电压低于或高于参考源电压时触发中断）
uint32_t res_deg	矫正值（GR551x不适用）	暂不支持，默认为0

详细信息请参考文件：

- GR551x/GR5526/GR5x25：`SDK_Folder\drivers\inc\hal\gr55xx_ll_comp.h`

版权所有 © 2025 深圳市汇顶科技股份有限公司

- GR533x：SDK_Folder\drivers\inc\hal\gr533x_ll_comp.h
- GR5405：SDK_Folder\hal_drv\include\ll_comp.h

2.15.1.4 app_comp_pin_cfg_t

结构体定义如下：

表 2-308 app_comp_pin_cfg_t结构体成员

成员	描述	取值
app_comp_pin_t input	COMP输入引脚配置	参考2.15.1.2 app_comp_pin_t结构体
app_comp_pin_t vref	COMP参考引脚配置	参考2.15.1.2 app_comp_pin_t结构体，如果参考源是内部参考电压或者电池电压，此参数可为NULL

详细信息请参考文件：SDK_Folder\drivers\inc\app_comp.h。

2.15.2 枚举

2.15.2.1 app_comp_evt_t

表 2-309 app_comp_evt_t枚举成员

成员	描述
APP_COMP_EVT_RISING = 0	COMP上升沿触发中断事件
APP_COMP_EVT_FALLING = 1	COMP下降沿触发中断事件
APP_COMP_EVT_DONE = 0	COMP上升沿触发中断事件（仅适用于GR551x），仅为了兼容GR551x SDK V1.7.0及之前的版本
APP_COMP_EVT_ERROR	比较器错误事件

2.15.3 接口函数

表 2-310 COMP驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_comp_init	Y	Y	Y	Y	Y
app_comp_deinit	Y	Y	Y	Y	Y
app_comp_start	Y	Y	Y	Y	Y
app_comp_stop	Y	Y	Y	Y	Y
app_comp_get_handle	Y	Y	Y	Y	Y

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.15.3.1 app_comp_init

表 2-311 app_comp_init接口

函数原型	uint16_t app_comp_init(app_comp_params_t *p_params, app_comp_evt_handler_t evt_handler)
功能说明	初始化COMP
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_comp_evt_handler_t)(app_comp_evt_t *p_evt) HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理

2.15.3.2 app_comp_deinit

表 2-312 app_comp_deinit接口

函数原型	uint16_t app_comp_deinit(void)
功能说明	反初始化COMP
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.15.3.3 app_comp_start

表 2-313 app_comp_start接口

函数原型	uint16_t app_comp_start (void)
功能说明	开启比较器
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.15.3.4 app_comp_stop

表 2-314 app_comp_stop接口

函数原型	uint16_t app_comp_stop(void)
功能说明	停止比较器
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.15.3.5 app_comp_get_handle

表 2-315 app_comp_get_handle接口

函数原型	comp_handle_t *app_comp_get_handle(void)
功能说明	获取COMP句柄
输入参数	
返回值	COMP句柄，详见 • GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_comp.h • GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_comp.h • GR5405：SDK_Folder\hal_drv\include\hal_comp.h
备注	

2.15.4 示例工程

示例工程代码帮助开发者快速理解并使用COMP驱动。

2.15.4.1 COMP实现内部参考电压与MSIO引脚电压值的比较

示例工程实现输入源和参考源电压值的比较，工程路径：SDK_Folder\projects\peripheral\comp。

2.15.4.1.1 流程图

COMP比较器实现输入源和参考源电压值的比较，输入源电压值为MSIO引脚外接电压值、参考源电压值可以是MSIO引脚外接电压值、电池电压和内部参考电压，当输出源电压大于参考源电压，中断产生。

示例设置COMP的模拟输入引脚MSIO0、选择内部参考电压，开启比较器，串口打印中断日志。

版权所有 © 2025 深圳市汇顶科技股份有限公司

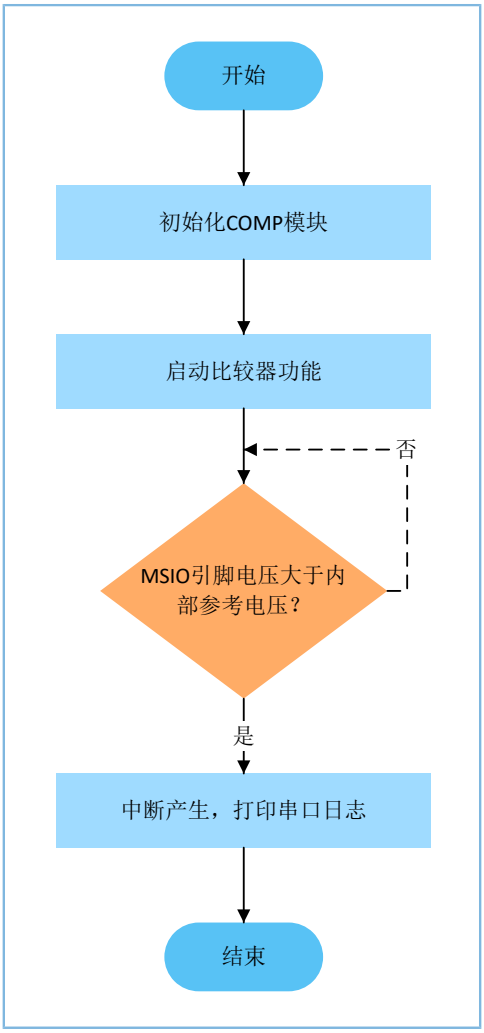

图 2-24 COMP实现电压比较流程图

2.15.4.1.2 初始化参数

```
app_comp_params_t params = {
    .pin_cfg = {
        .input = {
            .type = APP_IO_TYPE_MSIO,
            .mux  = APP_COMP_INPUT_PIN_MUX,
            .pin  = APP_COMP_INPUT_PIN,
        },
        .vref = {
            .type = APP_IO_TYPE_MSIO,
            .mux  = APP_COMP_VREF_PIN_MUX,
            .pin  = APP_COMP_VREF_PIN,
        },
    },
    .init = {
        .input_source = COMP_INPUT_SRC_IO0,
        .ref_source   = COMP_REF_SRC_VREF,
#if (APP_DRIVER_CHIP_TYPE == APP_DRIVER_GR551X)
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
        .ref_value    = 30,
#else
        .ref_value    = 120,
#endif
    },};
void comp_interrupt(void)
{
#if (APP_DRIVER_CHIP_TYPE != APP_DRIVER_GR551X)
    params.init.edge = COMP_WAKEUP_EDGE_BOTH;
#endif
    app_comp_init(&params, app_comp_event_handler);
    app_comp_start();
}
```

- pin_cfg：引脚配置
  - pin_cfg.input：比较器输入引脚配置
  - pin_cfg.vref：比较器参考源引脚配置
- init：COMP参数配置
  - init.input_source：输入源COMP_INPUT_SRC_IO0（选择MSIO0）
  - init.ref_source：参考源COMP_REF_SRC_VREF（选择内部参考电压）
  - init.ref_value：内部参考源档位，GR551x为30，其他芯片为120（GR551x参考电压：30*30 mV；其他：120*7.45 mV）
  - init.edge：触发方式LL_COMP_WAKEUP_EDGE_RISING（上升沿触发）

2.15.4.1.3 重要函数

- 初始化COMP

```
app_comp_init(&params, app_comp_event_handler)
```

- 开启比较器

```
app_comp_start();
```

2.15.4.1.4 测试验证

连接开发板到PC端，打开串口助手，连接串口，查看串口日志，MSIO0输入大于参考电压的电压，查看中断是否触发。

以GR5526为例，参考电压约等于0.894 V，即（120*7.45 ）mV，开启了正向和逆向迟滞，上升沿触发时，只有输入电压大于0.92 V会触发中断，即（0.894 + 0.026）mV。再次触发需要输入电压下降到小于0.868 V（0.894 － 0.026）之后，再上升到大于0.92 V才能继续触发上升沿中断，下降沿触发类似。用电源输出1.0 V的电压，接入MSIO0，打印日志如下：

```
[19:23:48.668]
This example demo end.
[19:26:14.001] Comp is rising triggered.
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
[19:42:47.356]
```

2.16 APP PWM驱动

2.16.1 宏定义

2.16.1.1 APP_PWM_PIN_ENABLE

表 2-316 APP_PWM_PIN_ENABLE宏定义

描述	值
开启PWM引脚功能	1

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.1.2 APP_PWM_PIN_DISABLE

表 2-317 APP_PWM_PIN_DISABLE宏定义

描述	值
关闭PWM引脚功能	0

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.2 结构体

2.16.2.1 app_pwm_pin_t

结构体定义如下：

表 2-318 app_pwm_pin_t结构体成员

成员	描述	取值
app_io_type_t type	设置IO类型	同2.2.1.2 app_io_evt_t中描述
app_io_mux_t mux	设置IO映射功能	参考2.2.1.1 app_io_init_t结构体，具体映射模式参考芯片Datasheet中的Pin Mux
uint32_t pin	设置IO引脚	参考2.2.1.1 app_io_init_t结构体，具体根据芯片Datasheet中PWM资源进行选择
app_io_pull_t pull	设置IO上拉/下拉/浮空	同2.2.1.1 app_io_init_t结构体中描述
uin8_t enable	开启或关闭PIN	该参数的取值可以是下列值中的任意一个： • APP_PWM_PIN_ENABLE（开启PIN） • APP_PWM_PIN_DISABLE（关闭PIN）

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.16.2.2 app_pwm_pin_cfg_t

结构体定义如下：

表 2-319 app_pwm_pin_cfg_t结构体成员

成员	描述	取值
app_pwm_pin_t channel_a	配置通道A的IO引脚	参考结构体2.16.2.1 app_pwm_pin_t
app_pwm_pin_t channel_b	配置通道B的IO引脚	参考结构体2.16.2.1 app_pwm_pin_t
app_pwm_pin_t channel_c	配置通道C的IO引脚	参考结构体2.16.2.1 app_pwm_pin_t

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.2.3 app_pwm_channel_init_t

结构体定义如下：

表 2-320 app_pwm_channel_init_t结构体成员

成员	描述	取值
uint8_t duty	配置PWM输出占空比，占空比从0% ～ 100%可配	0 ～ 100之间的任意整数
uint8_t drive_polarity	配置PWM输出极性	该参数的取值可以是下列值中的任意一个： • PWM_DRIVEPOLARITY_NEGATIVE（输出低电平为激活态） • PWM_DRIVEPOLARITY_POSITIVE（输出高电平为激活态）
uint32_t fstoplvl	Flicker模式下的停止时电平配置，GR551x无该参数	该参数的取值可以是下列值中的任意一个： • PWM_STOP_LVL_LOW（PWM停止后IO保持低电平） • PWM_STOP_LVL_HIGH（PWM停止后IO保持高电平）

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.2.4 app_pwm_params_t

结构体定义如下：

表 2-321 app_pwm_params_t结构体成员

成员	描述	取值
app_pwm_id_t id	指定PWM模块ID	该参数的取值可以是下列值中的任意一个： • APP_PWM_ID_0（PWM模块0） • APP_PWM_ID_1（PWM模块1）
app_pwm_pin_cfg_t pin_cfg	配置PWM各通道IO属性	参考结构体2.16.2.2 app_pwm_pin_cfg_t
app_pwm_active_channel_t active_channel	配置PWM激活的通道	该参数的取值可以是下列值中的任意一个： • APP_PWM_ACTIVE_CHANNEL_A（PWM通道A激活） • APP_PWM_ACTIVE_CHANNEL_B（PWM通道B激活） • APP_PWM_ACTIVE_CHANNEL_C（PWM通道C激活） • APP_PWM_ACTIVE_CHANNEL_ALL（PWM所有通道激活）

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• APP_PWM_ACTIVE_CHANNEL_CLEARED（PWM所有通道不激活）
pwm_init_t init	配置PWM初始化参数	参考结构体2.16.2.5 pwm_init_t
pwm_env_t pwm_env	仅用于记录初始化设备的句柄、相关参数等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.2.5 pwm_init_t

结构体定义如下：

表 2-322 pwm_init_t结构体成员（适用于GR551x、GR5526和GR5x25）

成员	描述	取值
uint32_t mode	PWM模式	该参数的取值可以是下列值中的任意一个： • PWM_MODE_FLICKER（固定占空比模式） • PWM_MODE_BREATH（呼吸模式）
uint32_t align	PWM脉冲对齐方式	该参数的取值可以是下列值中的任意一个： • PWM_ALIGNED_EDGE（三个通道的波形以PWM周期起始边沿对齐） • PWM_ALIGNED_CENTER（三个通道的波形以PWM周期中心点对齐）
uint32_t freq	PWM频率	0 ～ SystemFreq/2（SystemFreq为系统时钟频率）
uint32_t bperiod	PWM在呼吸模式下的呼吸周期，单位ms	0 ～ 0xFFFFFFFF/SystemFreq*1000（SystemFreq为系统时钟频率） **注意：** PWM占空比从0%变化到100%所需的周期数应至少大于2。
uint32_t hperiod	PWM在呼吸模式下的保持时间，单位ms	0 ～ 0xFFFFFF/SystemFreq*1000（SystemFreq为系统时钟频率）
uint32_t bstoplvl	PWM在呼吸模式下，停止时电平配置（GR551x无该参数）	该参数的取值可以是下列值中的任意一个： • PWM_STOP_LVL_LOW（PWM停止时为低电平） • PWM_STOP_LVL_HIGH（PWM停止时为高电平）
pwm_channel_init_t channel_a	通道A的配置参数	参考2.16.2.6 pwm_channel_init_t结构体
pwm_channel_init_t channel_b	通道B的配置参数	参考2.16.2.6 pwm_channel_init_t结构体
pwm_channel_init_t channel_c	通道C的配置参数	参考2.16.2.6 pwm_channel_init_t结构体

pwm_init_t结构体成员（适用于GR551x、GR5526和GR5x25）的详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_pwm.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-323 GR533x/GR5405 pwm_init_t结构体成员

成员	描述	取值
uint32_t mode	模式	该参数的取值可以是下列值中的任意一个： • PWM_MODE_FLICKER（固定占空比模式） • PWM_MODE_BREATH（呼吸模式） • PWM_CODING_MODE （编码模式）
uint32_t prd_cycles	周期循环个数	循环输出脉冲数，为0时持续输出
pwm_none_coding_init_t none_coding_mode_cfg	非编码下配置	参考代码中的pwm_none_coding_init_t
pwm_coding_init_t coding_mode_cfg	编码配置	参考代码中的pwm_coding_init_t

pwm_init_t结构体成员的详细信息，请参考文件：

- GR533x：`SDK_Folder\drivers\inc\hal\gr533x_hal_pwm.h`
- GR5405：`SDK_Folder\hal_drv\include\hal_pwm.h`

2.16.2.6 pwm_channel_init_t

结构体定义如下：

表 2-324 pwm_channel_init_t结构体成员

成员	描述	取值
uint8_t duty	PWM占空比	0 ～ 100
uint8_t drive_polarity	PWM输出模式下的驱动极性	该参数的取值可以是下列值中的任意一个： • PWM_DRIVEPOLARITY_NEGATIVE（负极性） • PWM_DRIVEPOLARITY_POSITIVE（正极性）
uint32_t fstoplvl	PWM在Flicker模式下，停止时的电平配置（GR551x无该参数）	该参数的取值可以是下列值中的任意一个： • PWM_STOP_LVL_LOW（PWM停止时为低电平） • PWM_STOP_LVL_HIGH（PWM停止时为高电平）

详细信息请参考文件：

- GR551x/GR5526/GR5x25：`SDK_Folder\drivers\inc\hal\gr55xx_hal_pwm.h`
- GR533x：`SDK_Folder\drivers\inc\hal\gr533x_hal_pwm.h`
- GR5405：`SDK_Folder\hal_drv\include\hal_pwm.h`

2.16.3 枚举

2.16.3.1 app_pwm_id_t

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-325 app_pwm_id_t枚举成员

成员	描述
APP_PWM_ID_0 = 0	PWM模块0
APP_PWM_ID_1 = 1	PWM模块1
APP_PWM_ID_MAX	PWM模块总数为2，用作参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.3.2 app_pwm_active_channel_t

枚举定义如下：

表 2-326 app_pwm_active_channel_t枚举成员

成员	描述
APP_PWM_ACTIVE_CHANNEL_A = 0x01	PWM激活通道A
APP_PWM_ACTIVE_CHANNEL_B = 0x02	PWM激活通道B
APP_PWM_ACTIVE_CHANNEL_C = 0x04	PWM激活通道C
APP_PWM_ACTIVE_CHANNEL_ALL = 0x07	PWM激活所有通道
APP_PWM_ACTIVE_CHANNEL_CLEARED = 0x00	PWM不激活所有通道

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.3.3 app_pwm_state_t

枚举定义如下：

表 2-327 app_pwm_state_t枚举成员

成员	描述
APP_PWM_INVALID = 0	PWM不可用
APP_PWM_ACTIVITY = 1	PWM运行中
APP_PWM_SLEEP= 2	PWM睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.3.4 app_pwm_module_state_t

枚举定义如下：

表 2-328 app_pwm_module_state_t枚举成员

成员	描述
APP_PWM_STOP = 0	PWM停止运行
APP_PWM_START = 1	PWM开始运行

版权所有 © 2025 深圳市汇顶科技股份有限公司

详细信息请参考文件：SDK_Folder\drivers\inc\app_pwm.h。

2.16.4 接口函数

表 2-329 PWM驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_pwm_init	Y	Y	Y	Y	Y
app_pwm_deinit	Y	Y	Y	Y	Y
app_pwm_start	Y	Y	Y	Y	Y
app_pwm_stop	Y	Y	Y	Y	Y
app_pwm_update_freq	Y	Y	Y	Y	Y
app_pwm_resume	N	N	N	Y	Y
app_pwm_pause	N	N	N	Y	Y
app_pwm_config_channel	Y	Y	Y	Y	Y
app_pwm_inactive_channel	N	Y	Y	Y	Y
app_pwm_get_handle	Y	Y	Y	Y	Y
app_pwm_set_coding_data_in_one_channel	N	N	N	Y	Y
app_pwm_set_coding_data_in_three_channels	N	N	N	Y	Y
app_pwm_start_coding_in_one_channel	N	N	N	Y	Y
app_pwm_start_coding_in_three_channels	N	N	N	Y	Y

2.16.4.1 app_pwm_init

表 2-330 app_pwm_init接口

函数原型	uint16_t app_pwm_init(app_pwm_params *p_params)
功能说明	根据初始化参数初始化PWM模块
输入参数	p_params：初始化参数的结构体指针，把结构体里的初始化变量传入该函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.16.4.2 app_pwm_deinit

表 2-331 app_pwm_deinit接口

函数原型	uint16_t app_pwm_deinit(app_pwm_id_t id)
功能说明	反初始化PWM模块
输入参数	id：指定需要反初始化的PWM模块（0或1）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义

版权所有 © 2025 深圳市汇顶科技股份有限公司

备注	

2.16.4.3 app_pwm_start

表 2-332 app_pwm_start接口

函数原型	uint16_t app_pwm_start(app_pwm_id_t id)
功能说明	PWM开始运行
输入参数	id：指定需要运行的PWM模块（0或1）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.16.4.4 app_pwm_stop

表 2-333 app_pwm_stop接口

函数原型	uint16_t app_pwm_stop(app_pwm_id_t id)
功能说明	PWM停止运行
输入参数	id：指定需要停止的PWM模块（0或1）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.16.4.5 app_pwm_update_freq

表 2-334 app_pwm_update_freq接口

函数原型	uint16_t app_pwm_update_freq(app_pwm_id_t id, uint32_t freq)
功能说明	更新PWM频率
输入参数	• id：指定需要更新频率的PWM模块（0或1） • freq：新的PWM频率
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	在PWM运行和停止时均可更新频率

2.16.4.6 app_pwm_resume

表 2-335 app_pwm_resume接口

函数原型	uint16_t app_pwm_resume(app_pwm_id_t id)
功能说明	恢复PWM信号输出
输入参数	id：指定需要配置的PWM模块（0或1）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.16.4.7 app_pwm_pause

表 2-336 app_pwm_pause接口

函数原型	uint16_t app_pwm_pause (app_pwm_id_t id)
功能说明	暂停PWM信号输出
输入参数	id：指定需要配置的PWM模块（0或1）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.16.4.8 app_pwm_config_channel

表 2-337 app_pwm_config_channel接口

函数原型	uint16_t app_pwm_config_channel(app_pwm_id_t id, app_pwm_active_channel_t channel, app_pwm_channel_init_t *p_config)
功能说明	配置PWM通道参数，包括占空比、极性和停止电平
输入参数	• id：指定需要配置的PWM模块（0或1） • channel：需要配置的通道选择 • p_config：需要配置的参数，包括占空比、极性和停止电平
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	在PWM运行和停止时均可调用该接口

2.16.4.9 app_pwm_inactive_channel

表 2-338 app_pwm_inactive_channel接口

函数原型	uint16_t app_pwm_inactive_channel(app_pwm_id_t id, app_pwm_active_channel_t channel)
功能说明	停止指定的通道输出
输入参数	• id：指定需要配置的PWM模块（0或1） • channel：需要失能的通道选择
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	在PWM运行和停止时均可调用该接口

2.16.4.10 app_pwm_get_handle

表 2-339 app_pwm_get_handle接口

函数原型	pwm_handle_t *app_pwm_get_handle(app_pwm_id_t id)
功能说明	获取当前PWM句柄
输入参数	id：指定需要获取的PWM模块（0或1）
返回值	pwm_handle_t：PWM句柄结构体
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.16.4.11 app_pwm_set_coding_data_in_one_channel

表 2-340 app_pwm_set_coding_data_in_one_channel接口

函数原型	uint16_t app_pwm_set_coding_data_in_one_channel(app_pwm_id_t id, uint32_t coding_data)
功能说明	配置单通道模式下PWM模块的编码数据
输入参数	• id：指定需要获取的PWM模块（0或1） • conding_data：一个32位的编码数据，每一位代表PWM0_A或PWM1_A的状态
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅单通道模式下可以使用该接口

2.16.4.12 app_pwm_set_coding_data_in_three_channels

表 2-341 app_pwm_set_coding_data_in_three_channels接口

函数原型	uint16_t app_pwm_set_coding_data_in_three_channels(app_pwm_id_t id, uint32_t coding_data0, uint32_t coding_data1, uint32_t coding_data2)
功能说明	配置三通道模式下PWM模块的编码数据
输入参数	• id：指定需要获取的PWM模块（0或1） • conding_data0：一个32位的编码数据，每一位代表PWM0_A或PWM1_A的状态 • conding_data1：一个32位的编码数据，每一位代表PWM0_B或PWM1_B的状态 • conding_data2：一个32位的编码数据，每一位代表PWM0_C或PWM1_C的状态
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅三通道模式下可以使用该接口

2.16.4.13 app_pwm_start_coding_in_one_channel

表 2-342 app_pwm_start_coding_in_one_channel接口

函数原型	uint16_t app_pwm_start_coding_in_one_channel(app_pwm_id_t id, uint32_t coding_data)
功能说明	在中断模式下开始生成一个通道的编码波形
输入参数	• id：指定需要获取的PWM模块（0或1） • conding_data：一个32位的编码数据，每一位代表PWM0_A或PWM1_A的状态
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅单通道模式下可以使用该接口

2.16.4.14 app_pwm_start_coding_in_three_channels

表 2-343 app_pwm_start_coding_in_three_channels接口

函数原型	uint16_t app_pwm_start_coding_in_three_channels(app_pwm_id_t id, uint32_t coding_data0, uint32_t coding_data1, uint32_t coding_data2)
功能说明	在中断模式下开始生成三个通道的编码波形
输入参数	• id：指定需要获取的PWM模块（0或1）

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• conding_data0：一个32位的编码数据，每一位代表PWM0_A或PWM1_A的状态 • conding_data1：一个32位的编码数据，每一位代表PWM0_B或PWM1_B的状态 • conding_data2：一个32位的编码数据，每一位代表PWM0_C或PWM1_C的状态
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅三通道模式下可以使用该接口

2.16.5 DMA接口函数

表 2-344 DMA接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_pwm_dma_init	N	N	N	Y	Y
app_pwm_dma_deinit	N	N	N	Y	Y
app_pwm_start_coding_with_dma	N	N	N	Y	Y

2.16.5.1 app_pwm_dma_init

表 2-345 app_pwm_dma_init接口

函数原型	uint16_t app_pwm_dma_init(app_pwm_params_t *p_params)
功能说明	初始化PWM DMA方式传输
输入参数	p_params：初始化参数的结构体指针，将结构体中的初始化变量传入该函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.16.5.2 app_pwm_dma_deinit

表 2-346 app_pwm_dma_deinit接口

函数原型	uint16_t app_pwm_dma_deinit(app_pwm_id_t id)
功能说明	反初始化PWM DMA方式传输
输入参数	id：指定需要获取的PWM模块（0或1）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.16.5.3 app_pwm_start_coding_with_dma

表 2-347 app_pwm_start_coding_with_dma接口

函数原型	uint16_t app_pwm_start_coding_with_dma(app_pwm_id_t id, uint32_t *p_data, uint16_t size)
功能说明	在DMA模式下开始生成单个/三个通道的编码波形
输入参数	• id：指定需要获取的PWM模块（0或1）

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• p_data：待搬运的编码数据。每搬运完连续的3个uin32_t数据后，会自动丢弃第4个uint32_t数据 • size：待搬运的编码数据的长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	• 初始化coding channel配置为单通道时，该接口生成PWMA的编码波形 • 初始化coding channel配置为三通道时，该接口生成PWMA、PWMB、PWMC的编码波形

2.16.6 示例工程

2.16.6.1 PWM Flicker

示例工程实现PWM的固定占空比模式，并改变PWM的频率、占空比等参数，可以通过逻辑分析仪进行波形观察，工程路径：SDK_Folder\projects\peripheral\pwm\app_pwm。

2.16.6.1.1 初始化参数

```
app_pwm_params_t pwm_params = {
    .id = APP_PWM0_MODULE,
    .pin_cfg = {
        .channel_a = {
            .type   = APP_PWM0_GPIO_TYPE,
            .mux    = APP_PWM0_GPIO_MUX,
            .pin    = APP_PWM0_CHANNEL_A,
            .pull   = APP_IO_NOPULL,
            .enable = APP_PWM_PIN_ENABLE,
        },
        .channel_b = {
            .type   = APP_PWM0_GPIO_TYPE,
            .mux    = APP_PWM0_GPIO_MUX,
            .pin    = APP_PWM0_CHANNEL_B,
            .pull   = APP_IO_NOPULL,
            .enable = APP_PWM_PIN_ENABLE,
        },
        .channel_c = {
            .type   = APP_PWM0_GPIO_TYPE,
            .mux    = APP_PWM0_GPIO_MUX,
            .pin    = APP_PWM0_CHANNEL_C,
            .pull   = APP_IO_NOPULL,
            .enable = APP_PWM_PIN_ENABLE,
        },
    },
    .active_channel = APP_PWM_ACTIVE_CHANNEL_ALL,
    .init = {
        .mode     = PWM_MODE_FLICKER,
        .align    = PWM_ALIGNED_EDGE,
        .freq     = 100,
        .bperiod  = 500,
        .hperiod  = 200,
#if (APP_DRIVER_CHIP_TYPE != APP_DRIVER_GR551X)
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
        .bstoplvl = PWM_STOP_LVL_LOW,
#endif
        .channel_a = {
            .duty = 30,
            .drive_polarity = PWM_DRIVEPOLARITY_POSITIVE,
#if (APP_DRIVER_CHIP_TYPE != APP_DRIVER_GR551X)
            .fstoplvl = PWM_STOP_LVL_LOW,
#endif
        },
        .channel_b = {
            .duty = 50,
            .drive_polarity = PWM_DRIVEPOLARITY_NEGATIVE,
#if (APP_DRIVER_CHIP_TYPE != APP_DRIVER_GR551X)
            .fstoplvl = PWM_STOP_LVL_HIGH,
#endif
        },
        .channel_c = {
            .duty = 80,
            .drive_polarity = PWM_DRIVEPOLARITY_POSITIVE,
#if (APP_DRIVER_CHIP_TYPE != APP_DRIVER_GR551X)
            .fstoplvl = PWM_STOP_LVL_LOW,
#endif
        },
    },
};
```

- id：定义PWM ID

  选择APP_PWM0_MODULE，即PWM0

- pin_cfg：配置通道A/B/C引脚
  - pin_cfg.channel_a：PWM通道A引脚配置
  - pin_cfg.channel_b：PWM通道B引脚配置
  - pin_cfg.channel_c：PWM通道C引脚配置
- active_channel：选择激活的通道

  选择APP_PWM_ACTIVE_CHANNEL_ALL，激活所有通道

- init：PWM参数配置
  - init.mode：采用PWM_MODE_FLICKER，即选择Flicker模式
  - init.align：PWM脉冲对齐方式PWM_ALIGNED_EDGE
  - init.freq：PWM频率10 MHz
  - init.bperiod：PWM在呼吸模式下的呼吸周期为500 ms（Flicker模式下，该参数无效）
  - init.hperiod：PWM在呼吸模式下的保持时间为200 ms（Flicker模式下，该参数无效）

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.bstoplvl：呼吸模式下停止电平：PWM_STOP_LVL_LOW表示停止电平为低电平，PWM_STOP_LVL_HIGH表示停止电平为高电平
- init.channel_a.duty：PWM通道A占空比
- init.channel_a.drive_polarity：PWM输出模式下的驱动极性，PWM_DRIVEPOLARITY_POSITIVE表示正极性，起始电平为低电平；PWM_DRIVEPOLARITY_NEGATIVE表示负极性，起始电平为高电平
- init.channel_a.fstoplvl：Flicker模式下停止电平：PWM_STOP_LVL_LOW表示停止电平为低电平，PWM_STOP_LVL_HIGH表示停止电平为高电平

2.16.6.1.2 重要函数

- 初始化PWM

```
ret = app_pwm_init(&pwm_params)
```

- 开始PWM输出

```
app_pwm_start(APP_PWM0_MODULE)
```

- 更新PWM频率

```
app_pwm_update_freq(APP_PWM0_MODULE, 1000)
```

- 修改通道参数

```
app_pwm_config_channel(APP_PWM0_MODULE, APP_PWM_ACTIVE_CHANNEL_B, &channel_cfg)
```

- 停止指定的通道输出

```
app_pwm_inactive_channel(APP_PWM0_MODULE, APP_PWM_ACTIVE_CHANNEL_A)
```

- 停止PWM输出

```
app_pwm_stop(APP_PWM0_MODULE)
```

2.16.6.1.3 测试验证

用逻辑分析仪观察MSIO0、MSIO1和MSIO2的波形变化，以GR5526为例，下面是观察的波形。

- MSIO0：开始输出波形占空比为30%，频率为100 Hz，然后停止输出，最后设置成频率为1000 Hz，占空比为60%的波形。
- MSIO1：开始输出波形占空比为50%，频率为100 Hz，然后进行了一次极性反转，接着输出频率为1000 Hz，占空比为20%的波形，最后设置成频率为1000 Hz，占空比为60%的波形。
- MSIO2：开始输出波形占空比为80%，频率为100 Hz，接着改变频率为1000 Hz，最后设置成频率为1000 Hz，占空比为60%的波形。

版权所有 © 2025 深圳市汇顶科技股份有限公司

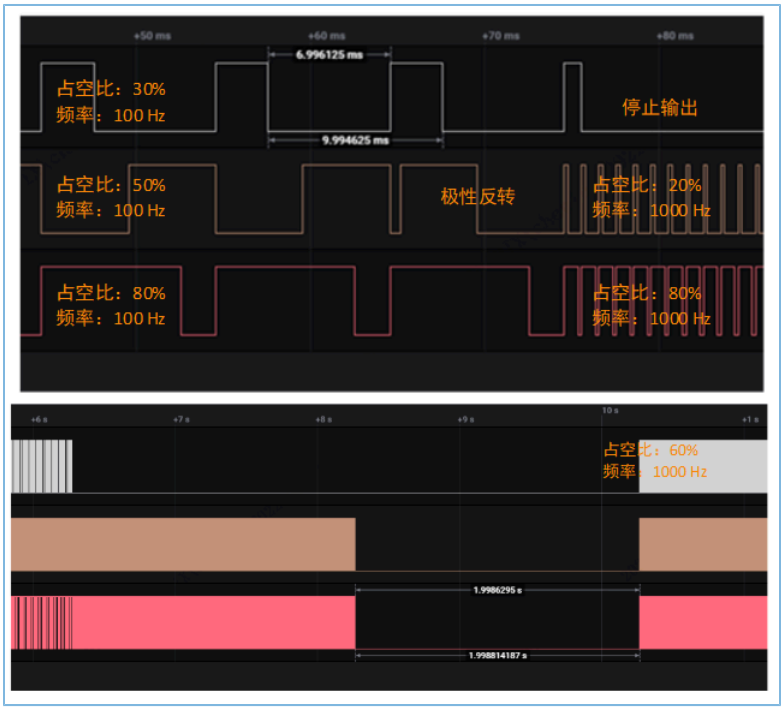

图 2-25 PWM Flicker示例工程输出波形图

2.17 APP RNG驱动

2.17.1 结构体

2.17.1.1 app_rng_params_t

结构体定义如下：

表 2-348 app_rng_params_t结构体成员

成员	描述	取值
app_rng_type_t use_type	设置RNG运行模式	该参数的取值可以是下列值中的任意一个： • APP_RNG_TYPE_INTERRUPT（中断模式） • APP_RNG_TYPE_POLLING（轮询模式） • APP_RNG_TYPE_MAX（参数检查时使用）
rng_init_t init	设置RNG初始化参数	参考2.17.1.3 rng_init_t结构体
rng_env_t rng_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_rng.h。

2.17.1.2 app_rng_evt_t

结构体定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-349 app_rng_evt_t结构体成员

成员	描述	取值
app_rng_evt_type_t type	事件的类型	该参数的取值可以是下列值中的任意一个： • APP_RNG_EVT_DONE（随机数生成成功） • APP_RNG_EVT_ERROR（随机数生成出错）
uint32_t random_data	生成的随机数	该参数为中断运行模式下生成的随机数

详细信息请参考文件：SDK_Folder\drivers\inc\app_rng.h。

2.17.1.3 rng_init_t

结构体定义如下：

表 2-350 rng_init_t结构体成员

成员	描述	取值
uint32_t seed_mode	产生随机数提供种子的方式	该参数的取值可以是下列值中的任意一个： • RNG_SEED_USER（采用用户提供的种子） • RNG_SEED_FR0_S0（环形振荡器S0提供的种子）
uint32_t lfsr_mode	LFSR模式	该参数的取值可以是下列值中的任意一个： • RNG_LFSR_MODE_59BIT（LFSR-59bit） • RNG_LFSR_MODE_128BIT（LFSR-128bit）
uint32_t out_mode	输出方式	该参数的取值可以是下列值中的任意一个： • RNG_OUTPUT_FR0_S0（环形振荡器输出） • RNG_OUTPUT_CYCLIC_PARITY（RNG、LFSR循环采样加奇偶校验输出） • RNG_OUTPUT_CYCLIC（RNG、LFSR循环采样输出） • RNG_OUTPUT_LFSR_RNG（LFSR、RNG异或输出） • RNG_OUTPUT_LFSR（LFSR输出）
uint32_t post_mode	后处理模式	该参数的取值可以是下列值中的任意一个： • RNG_POST_PRO_NOT（无处理） • RNG_POST_PRO_SKIPPING（跳位处理） • RNG_POST_PRO_COUNTING（位计数处理） • RNG_POST_PRO_NEUMANN（冯诺依曼处理）

详细信息请参考文件：

- GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_rng.h
- GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_rng.h
- GR5405：SDK_Folder\hal_drv\include\hal_rng.h

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.17.2 枚举

2.17.2.1 app_rng_type_t

枚举定义如下：

表 2-351 app_rng_type_t枚举成员

成员	描述
APP_RNG_TYPE_INTERRUPT	中断运行模式
APP_RNG_TYPE_POLLING	轮询运行模式
APP_RNG_TYPE_MAX	参数检查使用，不能作为入参

详细信息请参考文件：SDK_Folder\drivers\inc\app_rng.h。

2.17.2.2 app_rng_evt_type_t

枚举定义如下：

表 2-352 app_rng_evt_type_t枚举成员

成员	描述
APP_RNG_EVT_DONE	随机数生成完成事件标志
APP_RNG_EVT_ERROR	随机数生成出错事件标志

详细信息请参考文件：SDK_Folder\drivers\inc\app_rng.h。

2.17.2.3 app_rng_state_t

枚举定义如下：

表 2-353 app_rng_state_t枚举成员

成员	描述
APP_RNG_INVALID = 0	RNG不可用
APP_RNG_ACTIVITY = 1	RNG激活
APP_RNG_SLEEP = 2	RNG睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_rng.h。

2.17.3 接口函数

表 2-354 RNG驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_rng_init	Y	Y	Y	Y	Y
app_rng_deinit	Y	Y	Y	Y	Y

版权所有 © 2025 深圳市汇顶科技股份有限公司

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_rng_gen_sync	Y	Y	Y	Y	Y
app_rng_gen_async	Y	Y	Y	Y	Y
app_rng_get_handle	Y	Y	Y	Y	Y

2.17.3.1 app_rng_init

表 2-355 app_rng_init接口

函数原型	uint16_t app_rng_init(app_rng_params_t *p_params, app_rng_evt_handler_t evt_handler);
功能说明	根据初始化参数初始化RNG模块
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_rng_evt_handler_t)(app_rng_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理。

2.17.3.2 app_rng_deinit

表 2-356 app_rng_deinit接口

函数原型	uint16_t app_rng_deinit(void);
功能说明	反初始化RNG模块
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.17.3.3 app_rng_gen_sync

表 2-357 app_rng_gen_sync接口

函数原型	uint16_t app_rng_gen_sync(uint16_t *p_seed, uint32_t *p_random32bit);
功能说明	同步（轮询）模式产生随机数
输入参数	• p_seed：用户指定种子 • p_random32bit：生成的32-bit随机数指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	p_seed仅在配置为RNG_SEED_USER才生效，若采用59-bit模式则需要提供64-bit种子，若采用128-bit模式则需要提供128-bit种子；当配置为RNG_SEED_FR0_S0时，p_seed可传入NULL

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.17.3.4 app_rng_gen_async

表 2-358 app_rng_gen_async接口

函数原型	uint16_t app_rng_gen_async(uint16_t *p_seed);
功能说明	异步（中断）模式产生随机数
输入参数	p_seed：用户指定种子
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	p_seed仅在配置为RNG_SEED_USER才生效，若采用59-bit模式则需要提供64-bit种子，若采用128-bit模式则需要提供128-bit种子；当配置为RNG_SEED_FR0_S0时，p_seed可传入NULL

2.17.3.5 app_rng_get_handle

表 2-359 app_rng_get_handle接口

函数原型	rng_handle_t *app_rng_get_handle(void);
功能说明	获取RNG句柄
输入参数	
返回值	RNG句柄，详见： • GR551x/GR5526/GR5x25：SDK_Folder\drivers\inc\hal\gr55xx_hal_rng.h • GR533x：SDK_Folder\drivers\inc\hal\gr533x_hal_rng.h • GR5405：SDK_Folder\hal_drv\include\hal_rng.h
备注	

2.17.4 示例工程

2.17.4.1 RNG

采用用户提供的种子和环形振荡器产生的种子生成随机数，工程路径：SDK_Folder\projects\peripheral\rng\app_rng。

2.17.4.1.1 流程图

示例工程采用了两种种子，且通过中断和轮询两种方式生成随机数。下面只以轮询方式进行说明，中断方式查看示例工程。

版权所有 © 2025 深圳市汇顶科技股份有限公司

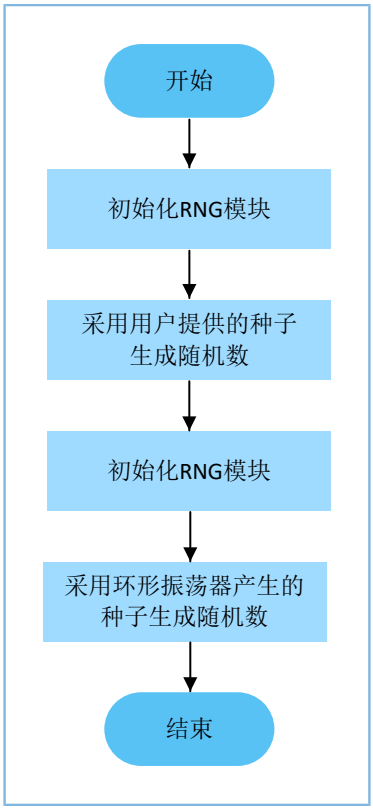

图 2-26 RNG生成随机数流程

2.17.4.1.2 使用用户提供的种子生成随机数

```
app_rng_params_t rng_polling_params = {
    .use_type = APP_RNG_TYPE_POLLING,
    .init = {
        .seed_mode = RNG_SEED_USER,
        .lfsr_mode = RNG_LFSR_MODE_59BIT,
        .out_mode  = RNG_OUTPUT_LFSR,
        .post_mode = RNG_POST_PRO_NOT,
    },
};
```

- use_type：传输方式选择APP_RNG_TYPE_POLLING（轮询）
- init.seed_mode：种子源选择RNG_SEED_USER（用户提供）
- init.lfsr_mode：LFSR模式选择RNG_LFSR_MODE_59BIT
- init.out_mode：输出模式选择RNG_OUTPUT_LFSR（LFSR直接输出）
- init.post_mode：后处理选择RNG_POST_PRO_NOT（不进行后处理）

2.17.4.1.3 使用环形振荡器产生的种子生成随机数

```
rng_polling_params.init.seed_mode = RNG_SEED_FR0_S0;
rng_polling_params.init.out_mode = RNG_OUTPUT_FR0_S0;
```

- init.seed_mode：种子源选择RNG_SEED_FR0_S0（环形振荡器S0提供的种子）

版权所有 © 2025 深圳市汇顶科技股份有限公司

- init.out_mode：输出模式选择RNG_OUTPUT_FR0_S0（环形振荡器直接输出）

2.17.4.1.4 重要函数

- 初始化RNG

```
app_rng_init (&rng_polling_params, NULL)
```

- 同步生成随机数

```
app_rng_gen_sync(g_random_seed, &random_number)
```

  - g_random_seed：随机种子（使用环形振荡器产生的种子，此参数可传入NULL）。
  - random_number：生成的随机数。

- 异步生成随机数

```
app_rng_gen_async(g_random_seed)
```

  g_random_seed：随机种子（使用环形振荡器产生的种子，此参数可传入NULL）。

2.17.4.1.5 测试验证

连接开发板到PC端，打开串口，串口将打印生成的随机数。

2.18 APP ISO7816驱动

2.18.1 宏定义

2.18.1.1 APP_ISO7816_ACTION_NONE

表 2-360 APP_ISO7816_ACTION_NONE宏定义

描述	值
空值，不执行动作	HAL_ISO7816_ACTION_NONE

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.2 APP_ISO7816_ACTION_OFF

表 2-361 APP_ISO7816_ACTION_OFF宏定义

描述	值
关闭	HAL_ISO7816_ACTION_OFF

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.1.3 APP_ISO7816_ACTION_STOPCLK

表 2-362 APP_ISO7816_ACTION_STOPCLK宏定义

描述	值
停止时钟	HAL_ISO7816_ACTION_STOPCLK

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.4 APP_ISO7816_ACTION_ON

表 2-363 APP_ISO7816_ACTION_ON宏定义

描述	值
上电并接受ATR	HAL_ISO7816_ACTION_ON

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.5 APP_ISO7816_ACTION_WARMRST

表 2-364 APP_ISO7816_ACTION_WARMRST宏定义

描述	值
热复位	HAL_ISO7816_ACTION_WARMRST

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.6 APP_ISO7816_ACTION_RX

表 2-365 APP_ISO7816_ACTION_RX宏定义

描述	值
接收数据	HAL_ISO7816_ACTION_RX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.7 APP_ISO7816_ACTION_TX

表 2-366 APP_ISO7816_ACTION_TX宏定义

描述	值
发送数据	HAL_ISO7816_ACTION_TX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.1.8 APP_ISO7816_ACTION_TXRX

表 2-367 APP_ISO7816_ACTION_TXRX宏定义

描述	值
发送数据然后接收数据	HAL_ISO7816_ACTION_TXRX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.9 APP_ISO7816_INTR_PRESENCE

表 2-368 APP_ISO7816_INTR_PRESENCE宏定义

描述	值
卡插入中断	HAL_ISO7816_INTR_PRESENCE

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.10 APP_ISO7816_INTR_STATE_ERR

表 2-369 APP_ISO7816_INTR_STATE_ERR宏定义

描述	值
状态错误中断	HAL_ISO7816_INTR_STATE_ERR

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.11 APP_ISO7816_INTR_DMA_ERR

表 2-370 APP_ISO7816_INTR_DMA_ERR宏定义

描述	值
DMA错误中断	HAL_ISO7816_INTR_DMA_ERR

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.12 APP_ISO7816_INTR_RETRY_ERR

表 2-371 APP_ISO7816_INTR_RETRY_ERR宏定义

描述	值
重复多次操作依然失败	HAL_ISO7816_INTR_RETRY_ERR

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.1.13 APP_ISO7816_INTR_RX_ERR

表 2-372 APP_ISO7816_INTR_RX_ERR宏定义

描述	值
接收异常中断	HAL_ISO7816_INTR_RX_ERR

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.14 APP_ISO7816_INTR_DONE

表 2-373 APP_ISO7816_INTR_DONE宏定义

描述	值
动作完成	HAL_ISO7816_INTR_DONE

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.15 APP_ISO7816_CARD_ABSENT

表 2-374 APP_ISO7816_CARD_ABSENT宏定义

描述	值
卡未插入	HAL_ISO7816_CARD_ABSENT

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.16 APP_ISO7816_CARD_PRESENT

表 2-375 APP_ISO7816_CARD_PRESENT宏定义

描述	值
卡已插入	HAL_ISO7816_CARD_PRESENT

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.17 APP_ISO7816_IO_STATE_OFF

表 2-376 APP_ISO7816_IO_STATE_OFF宏定义

描述	值
IO处于关闭状态	HAL_ISO7816_IO_STATE_OFF

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.1.18 APP_ISO7816_IO_STATE_IDLE

表 2-377 APP_ISO7816_IO_STATE_IDLE宏定义

描述	值
IO处于空闲状态	HAL_ISO7816_IO_STATE_IDLE

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.19 APP_ISO7816_IO_STATE_RX_WAIT

表 2-378 APP_ISO7816_IO_STATE_RX_WAIT宏定义

描述	值
IO处于等待接收状态	HAL_ISO7816_IO_STATE_RX_WAIT

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.20 APP_ISO7816_IO_STATE_RX

表 2-379 APP_ISO7816_IO_STATE_RX宏定义

描述	值
IO处于正在接收状态	HAL_ISO7816_IO_STATE_RX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.21 APP_ISO7816_IO_STATE_TX

表 2-380 APP_ISO7816_IO_STATE_TX宏定义

描述	值
IO处于正在发送状态	HAL_ISO7816_IO_STATE_TX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.22 APP_ISO7816_IO_STATE_TX_GUARD

表 2-381 APP_ISO7816_IO_STATE_TX_GUARD宏定义

描述	值
IO处于发送保护状态	HAL_ISO7816_IO_STATE_TX_GUARD

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.1.23 APP_ISO7816_PWR_STATE_OFF

表 2-382 APP_ISO7816_PWR_STATE_OFF宏定义

描述	值
电源为关闭状态	HAL_ISO7816_PWR_STATE_OFF

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.24 APP_ISO7816_PWR_STATE_PWRUP_VCC

表 2-383 APP_ISO7816_PWR_STATE_PWRUP_VCC宏定义

描述	值
电源为VCC上电状态	HAL_ISO7816_PWR_STATE_PWRUP_VCC

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.25 APP_ISO7816_PWR_STATE_PWRUP_RST

表 2-384 APP_ISO7816_PWR_STATE_PWRUP_RST宏定义

描述	值
电源为上电复位状态	HAL_ISO7816_PWR_STATE_PWRUP_RST

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.26 APP_ISO7816_PWR_STATE_PWRDN_RST

表 2-385 APP_ISO7816_PWR_STATE_PWRDN_RST宏定义

描述	值
电源为下电复位状态	HAL_ISO7816_PWR_STATE_PWRDN_RST

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.27 APP_ISO7816_PWR_STATE_PWRDN_VCC

表 2-386 APP_ISO7816_PWR_STATE_PWRDN_VCCR宏定义

描述	值
电源为VCC下电状态	HAL_ISO7816_PWR_STATE_PWRDN_VCC

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.1.28 APP_ISO7816_PWR_STATE_STOP_PRE

表 2-387 APP_ISO7816_PWR_STATE_STOP_PRE宏定义

描述	值
电源为准备停止时钟状态	HAL_ISO7816_PWR_STATE_STOP_PRE

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.29 APP_ISO7816_PWR_STATE_STOP

表 2-388 APP_ISO7816_PWR_STATE_STOP宏定义

描述	值
电源为时钟停止状态	HAL_ISO7816_PWR_STATE_STOP

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.30 APP_ISO7816_PWR_STATE_STOP_POST

表 2-389 APP_ISO7816_PWR_STATE_STOP_POST宏定义

描述	值
电源为退出时钟停止状态	HAL_ISO7816_PWR_STATE_STOP_POST

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.31 APP_ISO7816_PWR_STATE_IDLE

表 2-390 APP_ISO7816_PWR_STATE_IDLE宏定义

描述	值
电源为空闲态	HAL_ISO7816_PWR_STATE_IDLE

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.32 APP_ISO7816_PWR_STATE_RX_TS0

表 2-391 APP_ISO7816_PWR_STATE_RX_TS0宏定义

描述	值
电源为接收TS0状态	HAL_ISO7816_PWR_STATE_RX_TS0

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.1.33 APP_ISO7816_PWR_STATE_RX_TS1

表 2-392 APP_ISO7816_PWR_STATE_RX_TS1宏定义

描述	值
电源为接收TS1状态	HAL_ISO7816_PWR_STATE_RX_TS1

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.34 APP_ISO7816_PWR_STATE_RX

表 2-393 APP_ISO7816_PWR_STATE_RX宏定义

描述	值
电源为接收状态	HAL_ISO7816_PWR_STATE_RX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.35 APP_ISO7816_PWR_STATE_TX

表 2-394 APP_ISO7816_PWR_STATE_TX宏定义

描述	值
电源为发送状态	HAL_ISO7816_PWR_STATE_TX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.1.36 APP_ISO7816_PWR_STATE_TX_RX

表 2-395 APP_ISO7816_PWR_STATE_TX_RX宏定义

描述	值
电源为发送并接收状态	HAL_ISO7816_PWR_STATE_TX_RX

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.2 结构体

2.18.2.1 app_iso7816_params_t

结构体定义如下：

表 2-396 app_iso7816_params_t结构体成员

成员	描述	取值
app_iso7816_mode_t use_mode	ISO7816操作模式	该参数的取值可以是下列值中的任意一个： • APP_ISO7816_TYPE_INTERRUPT（中断） • APP_ISO7816_TYPE_POLLING（轮询）
app_iso7816_pin_cfg_t pin_cfg	ISO7816引脚配置	参考2.18.2.2 app_iso7816_pin_cfg_t结构体

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
iso7816_init_t init	ISO7816通信参数	参考2.18.2.4 iso7816_init_t结构体
iso7816_env_t iso7816_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.2.2 app_iso7816_pin_cfg_t

结构体定义如下：

表 2-397 app_iso7816_pin_cfg_t结构体成员

成员	描述	取值
app_iso7816_pin_t clk	ISO7816时钟引脚	参考2.18.2.3 app_iso7816_pin_t结构体
app_iso7816_pin_t rst	ISO7816 reset引脚	参考2.18.2.3 app_iso7816_pin_t结构体
app_iso7816_pin_t io	ISO7816 IO引脚	参考2.18.2.3 app_iso7816_pin_t结构体
app_iso7816_pin_t presence	ISO7816 状态引脚	参考2.18.2.3 app_iso7816_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.2.3 app_iso7816_pin_t

结构体定义如下：

表 2-398 app_iso7816_pin_t结构体成员

成员	描述	取值
app_io_type_t type	IO类型	同2.2.1.2 app_io_evt_t定义
app_io_mux_t mux	设置IO映射功能	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
uint32_t pin	选择IO引脚	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
app_io_pull_t pull	设置IO上拉/下拉/浮空	同2.2.1.1 app_io_init_t结构体中的描述

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.2.4 iso7816_init_t

结构体定义如下：

表 2-399 iso7816_init_t结构体成员

成员	描述	取值
uint32_t clk_div	分频系数	ISO7816输出时钟为(system clock)/(clk_div+1)

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
uint32_t wait_time	等待回应时间，单位：ETU.	0 ～ 0x3FFFF
uint16_t guard_time	两个连续字符之间的间隔时间，单位：ETU	0 ～ 0x3FF
uint8_t detect_coding	指定在ATR接收期间是否自动检测编码约定。	• 0：不使能 • 1：使能

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.2.5 app_iso7816_evt_t

结构体定义如下：

表 2-400 app_iso7816_evt_t结构体成员

成员	描述	取值
app_iso7816_evt_type_t type	事件类型	该参数的取值可以是下列值中的任意一个： • APP_ISO7816_EVT_ERROR（发生错误） • APP_ISO7816_EVT_ABORT（放弃之前操作） • APP_ISO7816_EVT_PRESENCE（卡插入或拔出） • APP_ISO7816_EVT_ATR_CPLT（ATR接收完成） • APP_ISO7816_EVT_TX_CPLT（发送完成） • APP_ISO7816_EVT_RX_CPLT（接收完成） • APP_ISO7816_EVT_TX_RX_CPLT（发送并接收完成）
union { uint32_t error_code; uint16_t size; }data;	uint32_t error_code：ISO7816事件返回的错误号 uint16_t size：ISO7816已接收/发送的数据长度 **说明：** 若事件类型为传输错误，此处返回错误号；若事件类型为接收/发送完成，此处显示已接收/发送的数据长度	错误号及对应含义如下： • HAL_ISO7816_ERROR_NONE（无错误） • HAL_ISO7816_ERROR_TIMEOUT（超时） • HAL_ISO7816_ERROR_TRANSFER（发送错误） • HAL_ISO7816_ERROR_INVALID_PARAM（无效参数）

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.3 枚举

2.18.3.1 app_iso7816_mode_t

枚举定义如下：

表 2-401 app_iso7816_mode_t枚举成员

成员	描述
APP_ISO7816_TYPE_INTERRUPT = 0	中断模式
APP_ISO7816_TYPE_POLLING = 1	轮询模式
APP_ISO7816_TYPE_MAX = 2	ISO7816模式总个数，用于参数检测

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.3.2 app_iso7816_state_t

枚举定义如下：

表 2-402 app_iso7816_state_t枚举成员

成员	描述
APP_ISO7816_INVALID = 0	ISO7816不可用
APP_ISO7816_ACTIVITY= 1	ISO7816运行中
APP_ISO7816_SLEEP = 2	ISO7816睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.3.3 app_iso7816_evt_type_t

枚举定义如下：

表 2-403 app_iso7816_evt_type_t枚举成员

成员	描述
APP_ISO7816_EVT_ERROR	发生错误
APP_ISO7816_EVT_ABORT	放弃之前操作
APP_ISO7816_EVT_PRESENCE	卡插入或拔出
APP_ISO7816_EVT_ATR_CPLT	ATR接收完成
APP_ISO7816_EVT_TX_CPLT	发送完成
APP_ISO7816_EVT_RX_CPLT	接收完成
APP_ISO7816_EVT_TX_RX_CPLT	发送并接收完成

详细信息请参考文件：SDK_Folder\drivers\inc\app_iso7816.h。

2.18.4 接口函数

表 2-404 ISO7816驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_iso7816_init	Y	Y	N	N	N

版权所有 © 2025 深圳市汇顶科技股份有限公司

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_iso7816_deinit	Y	Y	N	N	N
app_iso7816_receive_sync	Y	Y	N	N	N
app_iso7816_receive_async	Y	Y	N	N	N
app_iso7816_transmit_sync	Y	Y	N	N	N
app_iso7816_transmit_async	Y	Y	N	N	N
app_iso7816_transmit_receive_sync	Y	Y	N	N	N
app_iso7816_transmit_receive_async	Y	Y	N	N	N
app_iso7816_set_action	Y	Y	N	N	N
app_iso7816_get_power_states	Y	Y	N	N	N
app_iso7816_set_etudiv	Y	Y	N	N	N
app_iso7816_get_handle	Y	Y	N	N	N

2.18.4.1 app_iso7816_init

表 2-405 app_iso7816_init接口

函数原型	uint16_t app_iso7816_init(app_iso7816_params_t *p_params, app_iso7816_evt_handler_t evt_handler)
功能说明	初始化ISO7816
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_iso7816_evt_handler_t)(app_iso7816_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理。

2.18.4.2 app_iso7816_deinit

表 2-406 app_iso7816_deinit接口

函数原型	uint16_t app_iso7816_deinit(void)
功能说明	反初始化ISO7816
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.3 app_iso7816_receive_sync

表 2-407 app_iso7816_receive_sync接口

函数原型	uint16_t app_iso7816_receive_sync(uint16_t size, uint32_t timeout)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	ISO7816轮询方式（同步）接收数据
输入参数	• size：数据长度 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.4 app_iso7816_receive_async

表 2-408 app_iso7816_receive_async接口

函数原型	uint16_t app_iso7816_receive_async(uint16_t size)
功能说明	ISO7816中断方式（异步）接收数据
输入参数	size：数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.5 app_iso7816_transmit_sync

表 2-409 app_iso7816_transmit_sync接口

函数原型	uint16_t app_iso7816_transmit_sync(uint16_t size, uint32_t timeout)
功能说明	ISO7816轮询方式（同步）发送数据
输入参数	• size：数据长度 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.6 app_iso7816_transmit_async

表 2-410 app_iso7816_transmit_async接口

函数原型	uint16_t app_iso7816_transmit_async(uint16_t size)
功能说明	ISO7816中断方式（异步）发送数据
输入参数	size：数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.7 app_iso7816_transmit_receive_sync

表 2-411 app_iso7816_transmit_receive_sync接口

函数原型	uint16_t app_iso7816_transmit_receive_sync(uint16_t tx_size, uint16_t rx_size, uint32_t timeout)
功能说明	ISO7816轮询方式（同步）先发送后接收数据

版权所有 © 2025 深圳市汇顶科技股份有限公司

输入参数	• tx_size：发送数据长度 • rx_size：接收数据长度 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.8 app_iso7816_transmit_receive_async

表 2-412 app_iso7816_transmit_receive_async接口

函数原型	uint16_t app_iso7816_transmit_receive_async(uint16_t tx_size, uint16_t rx_size)
功能说明	ISO7816中断方式（异步）先发送后接收数据
输入参数	• tx_size：发送数据长度 • rx_size：接收数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.9 app_iso7816_set_action

表 2-413 app_iso7816_set_action接口

函数原型	uint16_t app_iso7816_set_action(uint32_t action)
功能说明	请求ISO7816执行下一个操作
输入参数	action：操作类型 选择以下参数的一个： • APP_ISO7816_ACTION_NONE • APP_ISO7816_ACTION_OFF • APP_ISO7816_ACTION_STOPCLK • APP_ISO7816_ACTION_ON • APP_ISO7816_ACTION_WARMRST • APP_ISO7816_ACTION_RX • APP_ISO7816_ACTION_TX • APP_ISO7816_ACTION_TXRX
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.10 app_iso7816_get_power_states

表 2-414 app_iso7816_get_power_states接口

函数原型	uint32_t app_iso7816_get_power_states(void)
功能说明	得到ISO7816电源状态
输入参数	

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	返回下面参数的一个： • APP_ISO7816_PWR_STATE_OFF • APP_ISO7816_PWR_STATE_PWRUP_VCC • APP_ISO7816_PWR_STATE_PWRUP_RST • APP_ISO7816_PWR_STATE_PWRDN_RST • APP_ISO7816_PWR_STATE_PWRDN_VCC • APP_ISO7816_PWR_STATE_STOP_PRE • APP_ISO7816_PWR_STATE_STOP • APP_ISO7816_PWR_STATE_STOP_POST • APP_ISO7816_PWR_STATE_IDLE • APP_ISO7816_PWR_STATE_RX_TS0 • APP_ISO7816_PWR_STATE_RX_TS1 • APP_ISO7816_PWR_STATE_RX • APP_ISO7816_PWR_STATE_TX • APP_ISO7816_PWR_STATE_TX_RX
备注	

2.18.4.11 app_iso7816_set_etudiv

表 2-415 app_iso7816_set_etudiv接口

函数原型	uint16_t app_iso7816_set_etudiv(uint32_t divide)
功能说明	对时钟进行分频
输入参数	divide：ETU分频系数（将CLK脚输出时钟除以该值+1，等于ETU长度）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.18.4.12 app_iso7816_get_handle

表 2-416 app_iso7816_get_handle接口

函数原型	iso7816_handle_t *app_iso7816_get_handle(void)
功能说明	获取ISO7816句柄
输入参数	
返回值	ISO7816句柄：详见SDK_Folder\drivers\inc\hal\gr55xx_hal_iso7816.h结构体
备注	

2.18.5 示例工程

2.18.5.1 ISO7816

ISO7816模块实现ISO 7816标准，本示例实现ISO7816模块读取SIM卡信息，工程路径：SDK_Folder\projects\peripheral\iso7816\app_iso7816。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.5.1.1 流程图

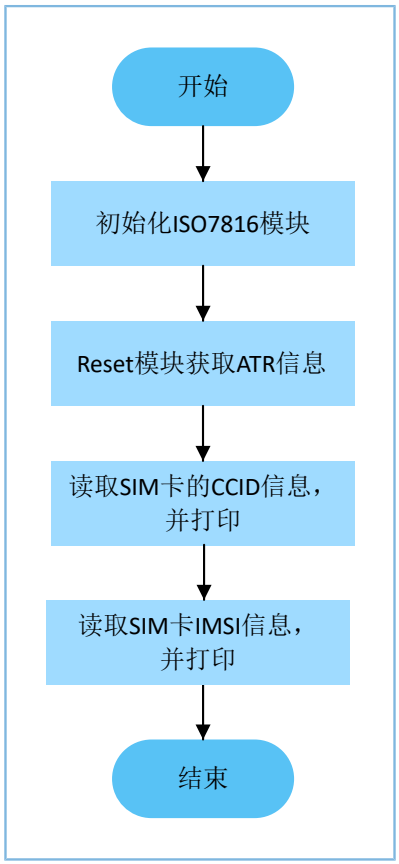

图 2-27 ISO7816读取SIM卡信息流程

2.18.5.1.2 初始化参数

```
app_iso7816_params_t params = {

    .use_mode = APP_ISO7816_TYPE_POLLING,
    .pin_cfg = DEFAULT_IO_CONFIG,
    .init = {
    .wait_time = 0x6B,
    .guard_time = 0x0A,
    .detect_coding = ENABLE,
    },
};
```

- use_mode：传输模式配置，选择APP_ISO7816_TYPE_POLLING，轮询模式
- pin_cfg：引脚配置，包含时钟引脚CLK、复位引脚Reset、数据输入输出引脚I/O、UICC Presence引脚
- init：通信参数配置，包括：
  - init.wait_time：等待回应时间，0x6B ETU
  - init.guard_time：两个连续字符之间的间隔时间，0x0A ETU
  - init.detect_coding：指定在ATR接收期间自动检测编码约定

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.18.5.1.3 重要函数

- 初始化ISO7816

```
app_iso7816_init(&params, app_iso7816_evt_handler)
```

- 获取ISO7816句柄

```
g_sim_card_handle = *app_iso7816_get_handle()
```

- Reset ISO7816模块，获取ATR信息

```
sim_card_get_ATR(&g_sim_card_handle)
```

- 协议类型选择（对ATR的响应）程序和配置通信速度

```
sim_card_PTS(&g_sim_card_handle)
```

- 选择Master的根目录

```
sim_card_select_MF(&g_sim_card_handle)
```

- 读取ICCID信息

```
sim_card_read_ICCID(&g_sim_card_handle)
```

- 读取IMSI信息

```
sim_card_read_IMSI(&g_sim_card_handle)
```

2.18.5.1.4 测试验证

外接SIM卡到开发板，连接开发板到PC端，打开串口助手，连接串口，查看串口日志，串口将打印ICCID和IMSI信息。每个SIM卡ICCID和IMSI信息存在差异，需要根据实际的SIM对比。

```
APP_ISO7816_EVT_PRESENCE

[16:43:43.815] APP_ISO7816_EVT_ATR_CPLT
3b 9d 95 80 1f c7 80 31 e0 73 fe 21 13 65 15 09 63 86 83 a1 00 00 00 00 00 00 00 00 00 00 00 00 00

[16:43:43.878]
ICCID: 89 86 00 82 19 18 85 44 35 42
[16:43:43.894]
IMSI: 80 94 60 02 47 54 35 40 32 APP_ISO7816_EVT_PRESENCE

This example demo end.
```

图 2-28 串口打印SIM卡信息

2.19 APP PDM驱动

2.19.1 结构体

2.19.1.1 app_pdm_params_t

结构体定义如下：

表 2-417 app_pdm_params_t结构体成员

成员	描述	取值
app_pdm_pin_cfg_t pin_cfg	引脚配置	参考2.19.1.2 app_pdm_pin_cfg_t结构体
app_pdm_dma_t dma_cfg	DMA配置	参考2.19.1.4 app_pdm_dma_t结构体
pdm_init_t init	初始化参数结构体	参考2.19.1.5 pdm_init_t结构体
pdm_env_t pdm_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

2.19.1.2 app_pdm_pin_cfg_t

结构体定义如下：

表 2-418 app_pdm_pin_cfg_t结构体成员

成员	描述	取值
app_pdm_pin_t clk	时钟引脚	参考2.19.1.3 app_pdm_pin_t结构体
app_pdm_pin_t data	数据引脚	参考2.19.1.3 app_pdm_pin_t结构体

2.19.1.3 app_pdm_pin_t

结构体定义如下：

表 2-419 app_pdm_pin_t结构体成员

成员	描述	取值
app_io_type_t type	IO类型	同2.2.1.2 app_io_evt_t定义
app_io_mux_t mux	设置IO映射功能	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
uint32_t pin	选择IO引脚	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考芯片Datasheet中的Pin Mux
app_io_pull_t pull	设置IO上拉/下拉/浮空	同2.2.1.1 app_io_init_t结构体中的描述

2.19.1.4 app_pdm_dma_t

结构体定义如下：

表 2-420 app_pdm_dma_t结构体成员

成员	描述	取值
dma_regs_t *dma_instance	DMA实例	DMA1
dma_channel_t dma_channel	DMA通道	同2.4.1.1 app_dma_params_t中通道描述

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.19.1.5 pdm_init_t

结构体定义如下：

表 2-421 pdm_init_t结构体成员

成员	描述	取值
uint32_t mode	采样模式	该参数的取值可以是下列值中的任意一个： • PDM_MODE_LEFT（左声道采样） • PDM_MODE_RIGHT（右声道采样） • PDM_MODE_STEREO（立体声采样）
uint32_t gain_l	左声道增益值，低14位有效，无符号数	0x0 ～ 0x3FFF 功率增益转换公式：Gain = 20*log(gain_l/1024)(dB)
uint32_t gain_r	右声道增益值，低14位有效，无符号数	0x0 ～ 0x3FFF 功率增益转换公式：Gain = 20*log(gain_r/1024)(dB)
uint32_t sample_rate	采样率	该参数的取值可以是下列值中的任意一个： • PDM_SAMPLE_RATE_15_625K（15.625 kHz） • PDM_SAMPLE_RATE_16K（16 kHz） • PDM_SAMPLE_RATE_8K（8 kHz）

详细信息请参考文件：`SDK_Folder\drivers\inc\hal\gr55xx_ll_pdm.h`。

2.19.1.6 app_pdm_evt_t

结构体定义如下：

表 2-422 app_pdm_evt_t结构体成员

成员	描述	取值
app_pdm_evt_type_t type	PDM事件	该参数的取值可以是下列值中的任意一个： • APP_PDM_EVT_DMA_ERROR = 0（DMA传输错误） • APP_PDM_EVT_DMA_TFR = 1（DMA传输完成） • APP_PDM_EVT_DMA_BLK = 2（DMA Block传输完成） • APP_PDM_EVT_LEFT_OVERFLOW = 3（PDM左通道溢出） • APP_PDM_EVT_RIGHT_OVERFLOW = 4（PDM右通道溢出）
uint32_t error_code	错误码	该参数的取值可以是下列值中的任意一个： • HAL_PDM_ERROR_NONE（无错误） • HAL_PDM_ERROR_TIMEOUT（超时） • HAL_PDM_ERROR_TRANSFER（发送错误） • HAL_PDM_ERROR_DMA（DMA传输错误） • HAL_PDM_ERROR_INVALID_PARAM（无效参数）

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.19.2 枚举

2.19.2.1 app_pdm_state_t

表 2-423 app_pdm_state_t枚举成员

成员	描述
APP_PDM_INACTIVITY = 0	PDM不可用
APP_PDM_ACTIVITY = 1	PDM运行中
APP_PDM_SLEEP = 2	PDM睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_pdm.h。

2.19.2.2 app_pdm_evt_type_t

表 2-424 app_pdm_evt_type_t枚举成员

成员	描述
APP_PDM_EVT_DMA_ERROR = 0	DMA传输错误
APP_PDM_EVT_DMA_TFR = 1	DMA传输完成
APP_PDM_EVT_DMA_BLK = 2	DMA Block传输完成
APP_PDM_EVT_LEFT_OVERFLOW = 3	PDM左通道溢出
APP_PDM_EVT_RIGHT_OVERFLOW = 4	PDM右通道溢出

2.19.3 接口函数

表 2-425 PDM驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_pdm_init	N	Y	Y	N	N
app_pdm_deinit	N	Y	Y	N	N
app_pdm_left_start_dma	N	Y	Y	N	N
app_pdm_right_start_dma	N	Y	Y	N	N
app_pdm_stereo_start_dma	N	Y	Y	N	N
app_pdm_left_start_dma_sg_llp	N	Y	Y	N	N
app_pdm_right_start_dma_sg_llp	N	Y	Y	N	N
app_pdm_stereo_start_dma_sg_llp	N	Y	Y	N	N
app_pdm_get_handle	N	Y	Y	N	N

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.19.3.1 app_pdm_init

表 2-426 app_pdm_init接口

函数原型	uint16_t app_pdm_init(app_pdm_params_t *p_params, app_pdm_evt_handler_t evt_handler)
功能说明	初始化PDM
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_pdm_evt_handler_t)(app_pdm_evt_t *type)； HAL层已经实现中断标志位的清理，用户不用关心中断标志的清理

2.19.3.2 app_pdm_deinit

表 2-427 app_pdm_deinit接口

函数原型	uint16_t app_pdm_deinit(void)
功能说明	反初始化PDM
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.19.3.3 app_pdm_left_start_dma

表 2-428 app_pdm_left_start_dma接口

函数原型	uint16_t app_pdm_left_start_dma(uint16_t *p_data, uint32_t length)
功能说明	PDM以DMA方式（异步）接收左声道数据
输入参数	• p_data：数据缓存指针 • length：数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.19.3.4 app_pdm_right_start_dma

表 2-429 app_pdm_right_start_dma接口

函数原型	uint16_t app_pdm_right_start_dma(uint16_t *p_data, uint32_t length)
功能说明	PDM以DMA方式（异步）接收右声道数据
输入参数	• p_data：数据缓存指针 • length：数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.19.3.5 app_pdm_stereo_start_dma

表 2-430 app_pdm_stereo_start_dma接口

函数原型	uint16_t app_pdm_stereo_start_dma(uint32_t *p_data, uint32_t length)
功能说明	PDM以DMA方式（异步）接收双声道数据
输入参数	• p_data：数据缓存指针 • length：数据长度
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.19.3.6 app_pdm_left_start_dma_sg_llp

表 2-431 app_pdm_left_start_dma_sg_llp接口

函数原型	uint16_t app_pdm_left_start_dma_sg_llp(uint16_t *p_data, uint32_t length, dma_sg_llp_config_t *sg_llp_config)
功能说明	PDM以DMA方式（异步）接收左声道数据，带分散聚合与传输链表配置
输入参数	• p_data：数据缓存指针 • length：数据长度 • sg_llp_config：分散聚合与传输链表
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.19.3.7 app_pdm_right_start_dma_sg_llp

表 2-432 app_pdm_right_start_dma_sg_llp接口

函数原型	uint16_t app_pdm_right_start_dma_sg_llp(uint16_t *p_data, uint32_t length, dma_sg_llp_config_t *sg_llp_config)
功能说明	PDM以DMA方式（异步）接收右声道数据，带分散聚合与传输链表配置
输入参数	• p_data：数据缓存指针 • length：数据长度 • sg_llp_config：分散聚合与传输链表
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.19.3.8 app_pdm_stereo_start_dma_sg_llp

表 2-433 app_pdm_stereo_start_dma_sg_llp接口

函数原型	uint16_t app_pdm_stereo_start_dma_sg_llp(uint32_t *p_data, uint32_t length, dma_sg_llp_config_t *sg_llp_config)
功能说明	PDM以DMA方式（异步）接收双声道数据，带分散聚合与传输链表配置

版权所有 © 2025 深圳市汇顶科技股份有限公司

输入参数	• p_data：数据缓存指针 • length：数据长度 • sg_llp_config：分散聚合与传输链表
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.19.3.9 app_pdm_get_handle

表 2-434 app_pdm_get_handle接口

函数原型	pdm_handle_t *app_pdm_get_handle(void)
功能说明	获取PDM句柄
输入参数	
返回值	PDM句柄：详见SDK_Folder\drivers\inc\hal\gr55xx_hal_pdm.h结构体
备注	

2.19.4 示例工程

同I2S，请参考2.11.5 示例工程。

2.20 APP GPU驱动

该驱动仅适用于GR5526。

2.20.1 宏定义

2.20.1.1 GRAPHICS_GPU_BASEADDR

表 2-435 GRAPHICS_GPU_BASEADDR宏定义

描述	值
定义GPU在Memory Map中的基地址	0xA3FF0000（此值应固定不变）

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_gpu.h。

2.20.1.2 GGPU_IRQ_EVT_CMD_LIST_END

表 2-436 GGPU_IRQ_EVT_CMD_LIST_END宏定义

描述	值
GPU命令列表结束中断回调标志	0x01

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_gpu.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.20.1.3 GGPU_IRQ_EVT_DRAW_CMD_END

表 2-437 GGPU_IRQ_EVT_DRAW_CMD_END宏定义

描述	值
GPU绘制命令结束中断回调标志	0x02

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_gpu.h。

2.20.1.4 GGPU_IRQ_EVT_IRQ_ID_CLEAR

表 2-438 GGPU_IRQ_EVT_IRQ_ID_CLEAR宏定义

描述	值
GPU中断ID清除中断回调标志	0x03

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_gpu.h。

2.20.1.5 GGPU_IRQ_EVT_OTHER

表 2-439 GGPU_IRQ_EVT_OTHER宏定义

描述	值
GPU其他中断回调标志	0x04

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_gpu.h。

2.20.2 枚举

2.20.2.1 graphics_gpu_power_state_e

枚举定义如下：

表 2-440 graphics_gpu_power_state_e枚举成员

成员	描述
GPU_POWER_STATE_SLEEP = 0	GPU电源模式为睡眠
GPU_POWER_STATE_ACTIVE = 1	GPU电源模式为活动

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_gpu.h。

2.20.3 接口函数

表 2-441 GPU驱动接口函数

API	芯片	
	GR551x/GR5x25/GR533x/GR5405	GR5526
graphics_gpu_init	N	Y
app_graphics_gpu_set_power_state	N	Y

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.20.3.1 graphics_gpu_init

表 2-442 graphics_gpu_init接口

函数原型	uint16_t graphics_gpu_init(graphics_gpu_irq_event_notify evt_cb)
功能说明	初始化GPU
输入参数	evt_cb：用户自己实现的中断回调函数（若没有中断回调函数，此参数可以为NULL）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.20.3.2 app_graphics_gpu_set_power_state

表 2-443 app_graphics_gpu_set_power_state接口

函数原型	void app_graphics_gpu_set_power_state(graphics_gpu_power_state_e state)
功能说明	配置GPU的电源模式
输入参数	state：GPU的电源模式，详见2.20.2.1 graphics_gpu_power_state_e
返回值	
备注	

2.20.4 示例工程

GPU的示例工程代码请参阅《GR5526 GPU开发者指南》。

2.21 APP OSPI驱动

该驱动仅适用于GR5526。

2.21.1 宏定义

2.21.1.1 OSPI_PSRAM_DEVICE_ID

表 2-444 OSPI_PSRAM_DEVICE_ID宏定义

描述	值
PSRAM ID	0x13

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.1.2 OSPI_PSRAM_MIN_XIP_ADDRESS

表 2-445 OSPI_PSRAM_MIN_XIP_ADDRESS宏定义

描述	值
XIP基地址	OSPI0_XIP_BASE（0x30080000UL）

版权所有 © 2025 深圳市汇顶科技股份有限公司

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.1.3 OSPI_PSRAM_MAX_XIP_ADDRESS

表 2-446 OSPI_PSRAM_MAX_XIP_ADDRESS宏定义

描述	值
64 Mbit空间，最大XIP地址	(OSPI0_XIP_BASE + 0x7FFFFF)

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.1.4 OSPI_PSRAM_BYTE_SIZE

表 2-447 OSPI_PSRAM_BYTE_SIZE宏定义

描述	值
PSRAM空间大小	0x800000

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.2 结构体

2.21.2.1 app_graphics_ospi_params_t

结构体定义如下：

表 2-448 app_graphics_ospi_params_t结构体成员

成员	描述	取值
app_ospi_clock_freq_e ospi_freq	OSPI时钟频率	该参数的取值可以是下列值中的任意一个： • OSPI_CLOCK_FREQ_48MHz = 0x00（48 MHz） • OSPI_CLOCK_FREQ_32MHz = 0x01（32 MHz） • OSPI_CLOCK_FREQ_24MHz = 0x02（24 MHz） • OSPI_CLOCK_FREQ_16MHz = 0x03 （16 MHz）
app_ospi_psram_drv_strength_e drv_strength	OSPI驱动强度， 驱动强度越大功耗越高	该参数的取值可以是下列值中的任意一个： • OSPI_PSRAM_DRV_STR_FULL = 0x00（25 Ohm） • OSPI_PSRAM_DRV_STR_HALF = 0x01（50 Ohm） • OSPI_PSRAM_DRV_STR_ONE_FORTH = 0x02（100 Ohm） • OSPI_PSRAM_DRV_STR_ONE_EIGHTH = 0x03（200 Ohm）
app_ospi_psram_rd_latency_e rd_lc	读访问latency	该参数的取值可以是下列值中的任意一个： • OSPI_PSRAM_RD_LATENCY_3 = 0x00 • OSPI_PSRAM_RD_LATENCY_4 = 0x01 • OSPI_PSRAM_RD_LATENCY_5 = 0x02 • OSPI_PSRAM_RD_LATENCY_6 = 0x03 • OSPI_PSRAM_RD_LATENCY_7 = 0x04
app_ospi_psram_wr_latency_e wr_lc	写访问latency	该参数的取值可以是下列值中的任意一个： • OSPI_PSRAM_WR_LATENCY_3 = 0x00

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• OSPI_PSRAM_WR_LATENCY_4 = 0x04 • OSPI_PSRAM_WR_LATENCY_5 = 0x02 • OSPI_PSRAM_WR_LATENCY_6 = 0x06 • OSPI_PSRAM_WR_LATENCY_7 = 0x01
uint32_t phy_delay	DQS PHY延时档位	该参数的取值可以是下列值中的任意一个： • OSPI_PHY_DELAY_TAP_2 = 0x01 • OSPI_PHY_DELAY_TAP_3 = 0x02（只能在频率小于等于32 MHz下使用） • OSPI_PHY_DELAY_TAP_4 = 0x03（只能在频率小于等于32 MHz下使用）
uint8_t is_read_prefetch	是否开启预取功能	• true：使能 • false：不使能

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.3 枚举

2.21.3.1 app_ospi_clock_freq_e

枚举定义如下：

表 2-449 app_ospi_clock_freq_e枚举成员

成员	描述
OSPI_CLOCK_FREQ_48MHz = 0x00	OSPI时钟频率48 MHz
OSPI_CLOCK_FREQ_32MHz = 0x01	OSPI时钟频率32 MHz
OSPI_CLOCK_FREQ_24MHz = 0x02	OSPI时钟频率24 MHz
OSPI_CLOCK_FREQ_16MHz = 0x03	OSPI时钟频率16 MHz

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.3.2 app_ospi_phy_delay_e

枚举定义如下：

表 2-450 app_ospi_phy_delay_e枚举成员

成员	描述
OSPI_PHY_DELAY_TAP_2 = 0x01	DQS Phy延时档位2
OSPI_PHY_DELAY_TAP_3 = 0x02	DQS Phy延时档位3（只能在频率小于等于32 MHz下使用）
OSPI_PHY_DELAY_TAP_4 = 0x03	DQS Phy延时档位4（只能在频率小于等于32 MHz下使用）

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.3.3 app_ospi_psram_drv_strength_e

枚举定义如下：

表 2-451 app_ospi_psram_drv_strength_e枚举成员

成员	描述
OSPI_PSRAM_DRV_STR_FULL = 0x00	PSRAM采用最强驱动
OSPI_PSRAM_DRV_STR_HALF = 0x01	PSRAM采用1/2强度驱动
OSPI_PSRAM_DRV_STR_ONE_FORTH = 0x02	PSRAM采用1/4强度驱动
OSPI_PSRAM_DRV_STR_ONE_EIGHTH = 0x03	PSRAM采用1/8强度驱动

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.3.4 app_ospi_psram_rd_latency_e

枚举定义如下：

表 2-452 app_ospi_psram_rd_latency_e枚举成员

成员	描述
OSPI_PSRAM_RD_LATENCY_3 = 0x00	配置读latency为3
OSPI_PSRAM_RD_LATENCY_4 = 0x01	配置读latency为4
OSPI_PSRAM_RD_LATENCY_5 = 0x02	配置读latency为5
OSPI_PSRAM_RD_LATENCY_6 = 0x03	配置读latency为6
OSPI_PSRAM_RD_LATENCY_7 = 0x04	配置读latency为7

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.3.5 app_ospi_psram_wr_latency_e

枚举定义如下：

表 2-453 app_ospi_psram_wr_latency_e枚举成员

成员	描述
OSPI_PSRAM_WR_LATENCY_3 = 0x00	配置写latency为3
OSPI_PSRAM_WR_LATENCY_4 = 0x04	配置写latency为4
OSPI_PSRAM_WR_LATENCY_5 = 0x02	配置写latency为5
OSPI_PSRAM_WR_LATENCY_6 = 0x06	配置写latency为6
OSPI_PSRAM_WR_LATENCY_7 = 0x01	配置写latency为7

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.3.6 app_ospi_access_mode_e

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-454 app_ospi_access_mode_e枚举成员

成员	描述
OSPI_ACCESS_MEMORY = 0	访问PSRAM的内存
OSPI_ACCESS_REGISTER	访问PSRAM的控制寄存器
OSPI_ACCESS_UNSET	未使用

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.3.7 app_ospi_work_state_e

枚举定义如下：

表 2-455 app_ospi_work_state_e枚举成员

成员	描述
OSPI_STATE_DEEP_SLEEP = 0	深度睡眠，除寄存器数据，所有内存数据擦除
OSPI_STATE_HALF_SLEEP = 1	浅度睡眠，所有数据保留
OSPI_STATE_ACTIVE = 2	激活状态，不进入睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

2.21.3.8 app_ospi_pasr_e

枚举定义如下：

表 2-456 app_ospi_pasr_e枚举成员

成员	描述
OSPI_PASR_FULL = 0x00	全部刷新
OSPI_PASR_BOTTOM_4MB = 0x01	刷新范围为低4 MB空间
OSPI_PASR_BOTTOM_2MB = 0x02	刷新范围为低2 MB空间
OSPI_PASR_BOTTOM_1MB = 0x03	刷新范围为低1 MB空间
OSPI_PASR_NONE = 0x04	关闭刷新
OSPI_PASR_TOP_4MB = 0x05	刷新范围为高4 MB空间
OSPI_PASR_TOP_2MB = 0x06	刷新范围为高2 MB空间
OSPI_PASR_TOP_1MB = 0x07	刷新范围为高1 MB空间
OSPI_PASR_MAX	刷新模式个数，仅用于参数检查

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_ospi.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.4 接口函数

表 2-457 OSPI驱动接口函数

API	芯片	
	GR551x/GR5x25/GR533x/GR5405	GR5526
app_graphics_ospi_init	N	Y
app_graphics_ospi_deinit	N	Y
app_graphics_ospi_reset	N	Y
app_graphics_ospi_register_psram_reload_func	N	Y
app_graphics_ospi_set_sleep_state	N	Y
app_graphics_ospi_get_sleep_state	N	Y
app_graphics_ospi_get_base_address	N	Y
app_graphics_ospi_set_pasr	N	Y
app_graphics_is_ospi_address	N	Y
app_graphics_ospi_set_read_prefetch	N	Y
app_graphics_ospi_get_read_prefetch	N	Y
app_graphics_ospi_get_byte_size	N	Y
app_graphics_ospi_pasr_update	N	Y
app_graphics_ospi_set_power_state	N	Y

2.21.4.1 app_graphics_ospi_init

表 2-458 app_graphics_ospi_init接口

函数原型	uint16_t app_graphics_ospi_init(app_graphics_ospi_params_t * p_params)
功能说明	初始化OSPI
输入参数	p_params：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.21.4.2 app_graphics_ospi_deinit

表 2-459 app_graphics_ospi_deinit接口

函数原型	void app_graphics_ospi_deinit(void)
功能说明	反初始化OSPI
输入参数	
返回值	
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.4.3 app_graphics_ospi_reset

表 2-460 app_graphics_ospi_reset接口

函数原型	void app_graphics_ospi_reset(void)
功能说明	软件复位PSRAM
输入参数	
返回值	
备注	复位后，PSRAM控制参数恢复成出厂状态，但数据不变化

2.21.4.4 app_graphics_ospi_register_psram_reload_func

表 2-461 app_graphics_ospi_register_psram_reload_func接口

函数原型	void app_graphics_ospi_register_psram_reload_func(psram_reload_func_t psram_reload_func)
功能说明	设置重载函数的指针
输入参数	psram_reload_func：进行资源重载的函数指针
返回值	
备注	PSRAM深度睡眠被唤醒后，会执行指针指向的重载函数（包括PSRAM内存资源重载等操作）

2.21.4.5 app_graphics_ospi_set_sleep_state

表 2-462 app_graphics_ospi_set_sleep_state接口

函数原型	void app_graphics_ospi_set_sleep_state(app_ospi_work_state_e state)
功能说明	设置系统休眠时OSPI PSRAM的工作状态
输入参数	state：系统休眠时OSPI PSRAM的工作状态
返回值	
备注	

2.21.4.6 app_graphics_ospi_get_sleep_state

表 2-463 app_graphics_ospi_get_sleep_state接口

函数原型	app_ospi_work_state_e app_graphics_ospi_get_sleep_state(void)
功能说明	获取系统休眠时OSPI PSRAM的工作状态
输入参数	
返回值	系统休眠时OSPI PSRAM的工作状态
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.4.7 app_graphics_ospi_get_base_address

表 2-464 app_graphics_ospi_get_base_address接口

函数原型	uint32_t app_graphics_ospi_get_base_address(void)
功能说明	获取OSPI PSRAM基地址
输入参数	
返回值	PSRAM基地址
备注	

2.21.4.8 app_graphics_ospi_set_pasr

表 2-465 app_graphics_ospi_set_pasr接口

函数原型	void app_graphics_ospi_set_pasr(app_ospi_pasr_e area)
功能说明	开始自刷新PSRAM指定区域
输入参数	参考2.21.3.8 app_ospi_pasr_e
返回值	
备注	进行刷新的空间会进行数据保持，不刷新区域不会进行数据保持，按需刷新以节省功耗

2.21.4.9 app_graphics_ospi_pasr_update

表 2-466 app_graphics_ospi_pasr_update接口

函数原型	void app_graphics_ospi_pasr_update(uint32_t psram_addr)
功能说明	通过使用地址配置PSRAM的自刷新区域
输入参数	PSRAM的最高使用地址（绝对地址）
返回值	
备注	该函数最终调用app_graphics_ospi_set_pasr(app_ospi_pasr_e area)实现自刷新

2.21.4.10 app_graphics_ospi_set_power_state

表 2-467 app_graphics_ospi_set_power_state接口

函数原型	void app_graphics_ospi_set_power_state(app_ospi_work_state_e state)
功能说明	设置OSPI工作状态
输入参数	参考2.21.3.7 app_ospi_work_state_e
返回值	
备注	• 在上一次设置为HALF_SLEEP或DEEP_SLEEP模式后，再次调用传输接口前，都需要调用此函数将OSPI工作状态设置为ACTIVE。 • OSPI的休眠唤醒跟随系统的休眠唤醒，由SDK统一管控。用户无需调用本接口再次设置OSPI的工作状态，只需调用void app_graphics_ospi_set_sleep_state(app_ospi_work_state_e state)接口指定系统休眠时的OSPI工作状态即可。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.4.11 app_graphics_ospi_get_byte_size

表 2-468 app_graphics_ospi_get_byte_size接口

函数原型	uint32_t app_graphics_ospi_get_byte_size(void)
功能说明	获取OSPI PSRAM空间大小
输入参数	
返回值	PSRAM空间大小，以byte为单位
备注	

2.21.4.12 app_graphics_is_ospi_address

表 2-469 app_graphics_is_ospi_address接口

函数原型	__STATIC_INLINE bool app_graphics_is_ospi_address(uint32_t address)
功能说明	检查地址是否在OSPI的PSRAM上
输入参数	address：需检查的地址
返回值	• true：地址在PSRAM上 • false：地址不在PSRAM上
备注	

2.21.4.13 app_graphics_ospi_set_read_prefetch

表 2-470 app_graphics_ospi_set_read_prefetch接口

函数原型	__STATIC_INLINE void app_graphics_ospi_set_read_prefetch(bool enable)
功能说明	使能/不使能预读取PSRAM
输入参数	• true：使能 • false：不使能
返回值	
备注	

2.21.4.14 app_graphics_ospi_get_read_prefetch

表 2-471 app_graphics_ospi_get_read_prefetch接口

函数原型	__STATIC_INLINE bool app_graphics_ospi_get_read_prefetch(void)
功能说明	判断OSPI是否激活预读取配置
输入参数	
返回值	• true：激活 • false：未激活
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.5 示例工程

2.21.5.1 读写PSRAM

说明:

该示例工程仅适用于GR5526。

示例工程主要实现对PSRAM内存的读写，工程路径：`SDK_Folder\projects\peripheral\ospi\app_ospi_random_access`。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.5.1.1 流程图

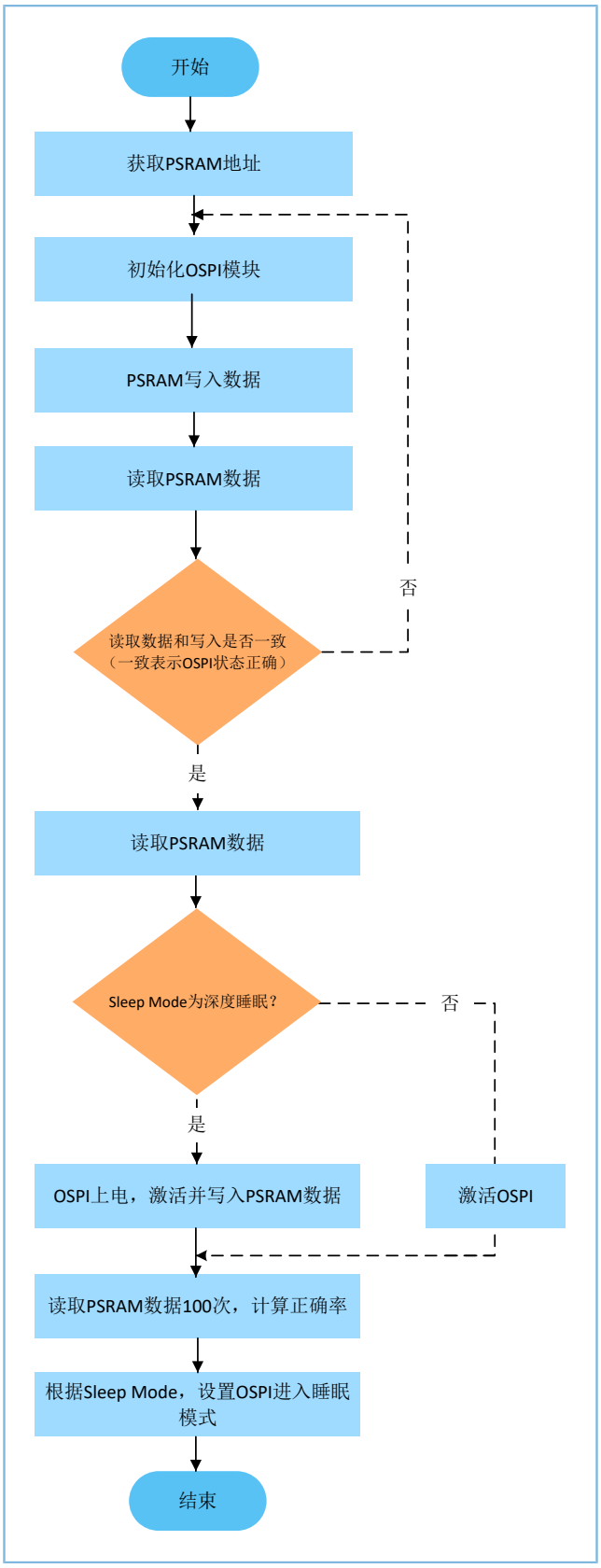

图 2-29 PSRAM内存读写流程

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.5.1.2 初始化参数

```
init.ospi_freq = O_PSRAM_TEST_CLOCK_FREQ ;
init.drv_strength = O_PSRAM_TEST_DRV_STRNGTH ;
init.rd_lc = O_PSRAM_TEST_RD_LATENCY ;
init.wr_lc = O_PSRAM_TEST_WR_LATENCY ;
init.phy_delay = O_PSRAM_TEST_PHY_DELAY ;
init.is_read_prefetch = O_PSRAM_TEST_PREFETCH ;
app_graphics_ospi_init(&init);
```

- init.ospi_freq：时钟频率O_PSRAM_TEST_CLOCK_FREQ，定义为OSPI_CLOCK_FREQ_48MHz，代表OSPI时钟频率48 MHz
- init.drv_strength：OSPI驱动强度O_PSRAM_TEST_DRV_STRNGTH，定义为OSPI_PSRAM_DRV_STR_ONE_FORTH，代表OSPI驱动强度为100 ohm
- init.rd_lc：读访问latency选择O_PSRAM_TEST_RD_LATENCY，定义为OSPI_PSRAM_RD_LATENCY_3，代表读访问latency3
- init.wr_lc：写访问latency选择O_PSRAM_TEST_WR_LATENCY，定义为OSPI_PSRAM_WR_LATENCY_3，代表写访问latency3
- init.phy_delay：DQS Phy延时选择O_PSRAM_TEST_PHY_DELAY，定义为OSPI_PHY_DELAY_TAP_2，代表DQS Phy延时档位2
- init.is_read_prefetch：是否开启预取功能，选择O_PSRAM_TEST_PREFETCH，定义为0，代表不开启预取功能

2.21.5.1.3 重要函数

- 获取PSRAM基地址

```
app_graphics_ospi_get_base_address()
```

- 初始化OSPI

```
ospi_init()
```

- 写PSRAM

```
cpu_write(base_addr, (char*)s_pattern_data, beat_length)
```

- 读PSRAM

```
cpu_read(base_addr, (char*)x_data, beat_length)
```

- 设置OSPI模块工作状态为激活态

```
app_graphics_ospi_set_power_state(OSPI_STATE_ACTIVE)
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.21.5.1.4 测试验证

示例工程实现不同数据传输模式和不同工作状态下，PSRAM的读写，具体测试可以根据需要进行选择。默认选择CPU+DEEP_SLEEP，连接开发板到PC端，打开串口调试助手，打印读PSRAM的错误率。

```
+++ Prepare Write-Data OK !
 +++ HalfSleep CPU Access Test, Failed/Total: 0/100
X
eee
 +++ HalfSleep CPU Access Test, Failed/Total: 0/200
X
eee
 +++ HalfSleep CPU Access Test, Failed/Total: 0/300
X
eee
 +++ HalfSleep CPU Access Test, Failed/Total: 0/400
X
eee
 +++ HalfSleep CPU Access Test, Failed/Total: 0/500
    app_graphics_ospi_init(&init);
```

2.22 APP DC驱动

该驱动仅适用于GR5526。

2.22.1 宏定义

2.22.1.1 GRAPHICS_DC_LAYER_0

表 2-472 GRAPHICS_DC_LAYER_0宏定义

描述	值
图层0	0u

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.1.2 GRAPHICS_DC_LAYER_1

表 2-473 GRAPHICS_DC_LAYER_1宏定义

描述	值
图层1	1u

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.22.1.3 GDC_IRQ_EVT_FRAME_TRANSMITION_END

表 2-474 GDC_IRQ_EVT_FRAME_TRANSMITION_END宏定义

描述	值
帧传输完成	0x01

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.1.4 GDC_IRQ_EVT_CMD_TRANSMITION_END

表 2-475 GDC_IRQ_EVT_CMD_TRANSMITION_END宏定义

描述	值
命令传输完成	0x02

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.1.5 GRAPHICS_DC_BASEADDR

表 2-476 GRAPHICS_DC_BASEADDR宏定义

描述	值
DC寄存器基地址	0xA3FF4000

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.2 结构体

2.22.2.1 app_graphics_dc_params_t

结构体定义如下：

表 2-477 app_graphics_dc_params_t结构体成员

成员	描述	取值
graphics_dc_mspi_e mspi_mode	DC SPI工作模式	该参数的取值可以是下列值中的任意一个： • GDC_MODE_SPI = 0 • GDC_MODE_DSPI • GDC_MODE_QSPI
graphics_dc_clock_freq_e clock_freq	DC时钟频率	该参数的取值可以是下列值中的任意一个： • GDC_CLOCK_FREQ_48MHz = 0x00 • GDC_CLOCK_FREQ_24MHz = 0x03 • GDC_CLOCK_FREQ_12MHz = 0x05 • GDC_CLOCK_FREQ_6MHz = 0x09 • GDC_CLOCK_FREQ_3MHz = 0x11
graphics_dc_clock_mode_e clock_mode	DC时钟模式	该参数的取值可以是下列值中的任意一个： • GDC_CLOCK_MODE_0 = 0x00

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• GDC_CLOCK_MODE_1 = 0x01 • GDC_CLOCK_MODE_2 = 0x02 • GDC_CLOCK_MODE_3 = 0x03
graphics_dc_tcsu_cycle_e tcsu_cycle	DC Tcsu延时	该参数的取值可以是下列值中的任意一个： • GDC_TCSU_CYCLE_0 = 0x00 • GDC_TCSU_CYCLE_1 = 0x01 • GDC_TCSU_CYCLE_2 = 0x02 • GDC_TCSU_CYCLE_3 = 0x03 • GDC_TCSU_CYCLE_4 = 0x04
graphics_dc_layer_mode_e layer_mode	DC渲染图层	该参数的取值可以是下列值中的任意一个： • GDC_ONE_LAYER_MODE = 0x00 • GDC_TWO_LAYER_MODE = 0x01
graphics_dc_mipi_format_e mipicfg_format	图像数据编码格式	该参数的取值可以是下列值中的任意一个： • GDC_MIPICFG_SPI_RGB565_OPT0 =MIPICFG_1RGB565_OPT0 • GDC_MIPICFG_SPI_RGB888_OPT0= MIPICFG_1RGB888_OPT0 • GDC_MIPICFG_DSPI_RGB565_OPT0 = MIPICFG_2RGB565_OPT0 • GDC_MIPICFG_DSPI_RGB888_OPT0= MIPICFG_2RGB888_OPT1 • GDC_MIPICFG_DSPI_RGB888_OPT1= MIPICFG_2RGB888_OPT0 • GDC_MIPICFG_QSPI_RGB565_OPT0= MIPICFG_4RGB565_OPT0 • GDC_MIPICFG_QSPI_RGB888_OPT0= MIPICFG_4RGB888_OPT0
uint16_t resolution_x	水平分辨率，单位：Pixel	
uint16_t resolution_y	垂直分辨率，单位：Pixel	
app_graphics_dc_pins_t pins_cfg	引脚配置	参考结构体2.22.2.2 app_graphics_dc_pins_t

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.2.2 app_graphics_dc_pins_t

结构体定义如下：

表 2-478 app_graphics_dc_pins_t结构体成员

成员	描述	取值
app_graphics_dc_pin_t csn	片选引脚	参考2.22.2.3 app_graphics_dc_pin_t结构体
app_graphics_dc_pin_t clk	CLK引脚	参考2.22.2.3 app_graphics_dc_pin_t结构体
app_graphics_dc_pin_t io0	IO0	参考2.22.2.3 app_graphics_dc_pin_t结构体
app_graphics_dc_pin_t io1	IO1	参考2.22.2.3 app_graphics_dc_pin_t结构体

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
app_graphics_dc_pin_t io2	IO2	参考2.22.2.3 app_graphics_dc_pin_t结构体
app_graphics_dc_pin_t io3	IO3	参考2.22.2.3 app_graphics_dc_pin_t结构体
app_graphics_dc_pin_t dcx	数据/命令控制脚	参考2.22.2.3 app_graphics_dc_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.2.3 app_graphics_dc_pin_t

结构体定义如下：

表 2-479 app_graphics_dc_pin_t结构体成员

成员	描述	取值
app_io_pull_t pull	设置IO上拉/下拉/浮空	同2.2.1.1 app_io_init_t结构体中的描述
uint8_t enable	引脚使能/不使能	• true：使能 • false：不使能

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.2.4 app_graphics_dc_framelayer_t

结构体定义如下：

表 2-480 app_graphics_dc_framelayer_t结构体成员

成员	描述	取值
void * frame_baseaddr	Frame Buffer地址	SRAM地址、Flash地址、PSRAM地址等；用户装载图片数据的内存地址，可以为Flash、PSRAM或SRAM，根据实际情况决定
uint32_t resolution_x	水平分辨率	根据实际刷屏分辨率决定
uint32_t resolution_y	垂直分辨率	根据实际刷屏分辨率决定
int32_t row_stride	行步长，单位字节	resolution_x乘以每个像素的字节数，不确定时可以传-1，驱动会辅助计算
int32_t start_x	开始渲染x坐标	一般从0开始
int32_t start_y	开始渲染y坐标	一般从0开始
uint32_t size_x	水平渲染范围	resolution_x
uint32_t size_y	垂直渲染范围	resolution_y
uint8_t alpha	透明度控制（Alpha）	默认0xff
uint8_t blendmode	混合模式	默认HAL_GDC_BL_SRC
graphics_dc_data_format_e data_format	图像数据编码格式	该参数的取值可以是下列值中的任意一个： • GDC_DATA_FORMAT_RGB565 = HAL_GDC_RGB565 • GDC_DATA_FORMAT_RGBA8888 = HAL_GDC_RGBA8888 • GDC_DATA_FORMAT_ABGR8888 = HAL_GDC_ABGR8888 • GDC_DATA_FORMAT_ARGB8888 = HAL_GDC_ARGB8888

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• GDC_DATA_FORMAT_BGRA8888 = HAL_GDC_BGRA8888 • GDC_DATA_FORMAT_TSC4 = HAL_GDC_TSC4 • GDC_DATA_FORMAT_TSC6 = HAL_GDC_TSC6 • GDC_DATA_FORMAT_TSC6A = HAL_GDC_TSC6A

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.2.5 app_graphics_dc_cmd_t

结构体定义如下：

表 2-481 app_graphics_dc_cmd_t结构体成员

成员	描述	取值
uint8_t command	Timing解析命令	根据外接屏幕配置
uint32_t address	Timing值地址	根据外接屏幕配置
app_graphics_dc_frame_address_width_e address_width	地址宽度	该参数的取值可以是下列值中的任意一个： • GDC_FRAME_ADDRESS_WIDTH_NONE = 0xFF • GDC_FRAME_ADDRESS_WIDTH_08BIT = MIPI_CMD08 • GDC_FRAME_ADDRESS_WIDTH_16BIT = MIPI_CMD16 • GDC_FRAME_ADDRESS_WIDTH_24BIT = MIPI_CMD24
app_graphics_dc_frame_timing_e frame_timing	帧时序	该参数的取值可以是下列值中的任意一个： • GDC_SPI_FRAME_TIMING_0 = 0x00 • GDC_DSPI_FRAME_TIMING_0 • GDC_QSPI_FRAME_TIMING_0 • GDC_QSPI_FRAME_TIMING_1

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.3 枚举

2.22.3.1 graphics_dc_mspi_e

枚举定义如下：

表 2-482 graphics_dc_mspi_e枚举成员

成员	描述
GDC_MODE_SPI = 0	使用MIPI Type-C SPI模式
GDC_MODE_DSPI	使用MIPI Type-C Dual SPI模式
GDC_MODE_QSPI	使用MIPI Type-C Quad SPI模式

2.22.3.2 graphics_dc_clock_freq_e

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-483 graphics_dc_clock_freq_e枚举成员

成员	描述
GDC_CLOCK_FREQ_48MHz = 0x00	DC时钟频率48 MHZ
GDC_CLOCK_FREQ_24MHz = 0x03	DC时钟频率24 MHZ
GDC_CLOCK_FREQ_12MHz = 0x05	DC时钟频率12 MHZ
GDC_CLOCK_FREQ_6MHz = 0x09	DC时钟频率6 MHZ
GDC_CLOCK_FREQ_3MHz = 0x11	DC时钟频率3 MHZ

2.22.3.3 graphics_dc_clock_mode_e

枚举定义如下：

表 2-484 graphics_dc_clock_mode_e枚举成员

成员	描述
GDC_CLOCK_MODE_0 = 0x00	时钟模式0，空闲状态下的时钟极性是逻辑低电平，数据在上升沿采样
GDC_CLOCK_MODE_1 = 0x01	时钟模式1，空闲状态下的时钟极性是逻辑低电平，数据在下降沿采样
GDC_CLOCK_MODE_2 = 0x02	时钟模式2，空闲状态下的时钟极性是逻辑高电平，数据在下降沿采样
GDC_CLOCK_MODE_3 = 0x03	时钟模式3，空闲状态下的时钟极性是逻辑高电平，数据在上升沿采样

2.22.3.4 graphics_dc_tcsu_cycle_e

枚举定义如下：

表 2-485 graphics_dc_tcsu_cycle_e枚举成员

成员	描述
GDC_TCSU_CYCLE_0 = 0x00	配置CS Setup Delay为默认（0.5时钟周期）
GDC_TCSU_CYCLE_1 = 0x01	配置CS Setup Delay增加1个时钟周期延时
GDC_TCSU_CYCLE_2 = 0x02	配置CS Setup Delay增加2个时钟周期延时
GDC_TCSU_CYCLE_3 = 0x03	配置CS Setup Delay增加3个时钟周期延时
GDC_TCSU_CYCLE_4 = 0x04	配置CS Setup Delay增加4个时钟周期延时

详细信息请参考文件：`SDK_Folder\drivers\inc\app_graphic_dc.h`。

2.22.3.5 graphics_dc_data_format_e

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-486 graphics_dc_data_format_e枚举成员

成员	描述
GDC_DATA_FORMAT_RGB565 = HAL_GDC_RGB565	RGB565编码，一个像素16 bits，没有Alpha通道
GDC_DATA_FORMAT_RGBA8888 = HAL_GDC_RGBA8888	RGBA8888编码，一个像素32 bits，有Alpha通道
GDC_DATA_FORMAT_ABGR8888 = HAL_GDC_ABGR8888	ABGR8888编码，一个像素32 bits，有Alpha通道
GDC_DATA_FORMAT_ARGB8888 = HAL_GDC_ARGB8888	ARGB8888编码，一个像素32 bits，有Alpha通道
GDC_DATA_FORMAT_BGRA8888 = HAL_GDC_BGRA8888	BGRA8888编码，一个像素32 bits，有Alpha通道
GDC_DATA_FORMAT_TSC4 = HAL_GDC_TSC4	TSC4编码格式，压缩后4 bits表示一个像素
GDC_DATA_FORMAT_TSC6 = HAL_GDC_TSC6	TSC6编码格式，压缩后6 bits表示一个像素
GDC_DATA_FORMAT_TSC6A = HAL_GDC_TSC6A	TSC6A编码格式，压缩后6 bits表示一个像素，并包含alpha信息

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphic_dc.h。

2.22.3.6 graphics_dc_mipi_format_e

枚举定义如下：

表 2-487 graphics_dc_mipi_format_e枚举成员

成员	描述
GDC_MIPICFG_SPI_RGB565_OPT0 = MIPICFG_1RGB565_OPT0	SPI模式，RGB565色彩格式，时序样式为option0
GDC_MIPICFG_SPI_RGB888_OPT0 = MIPICFG_1RGB888_OPT0	SPI模式，RGB888色彩格式，时序样式为option0
GDC_MIPICFG_DSPI_RGB565_OPT0 = MIPICFG_2RGB565_OPT0	Dual SPI模式，RGB565色彩格式，时序样式为option0
GDC_MIPICFG_DSPI_RGB888_OPT0 = MIPICFG_2RGB888_OPT1	Dual SPI模式，RGB888色彩格式，时序样式为option0
GDC_MIPICFG_DSPI_RGB888_OPT1 = MIPICFG_2RGB888_OPT0	Dual SPI模式，RGB888色彩格式，时序样式为option1
GDC_MIPICFG_QSPI_RGB565_OPT0 = MIPICFG_4RGB565_OPT0	Quad SPI模式，RGB565色彩格式，时序样式为option0
GDC_MIPICFG_QSPI_RGB888_OPT0 = MIPICFG_4RGB888_OPT0	Quad SPI模式，RGB888色彩格式，时序样式为option0

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphic_dc.h。

2.22.3.7 graphics_dc_out_pixel_bits_e

枚举定义如下：

表 2-488 graphics_dc_out_pixel_bits_e枚举成员

成员	描述
GDC_OUT_PIXEL_BITS_16 = 16	输出数据像素16 bits
GDC_OUT_PIXEL_BITS_24 = 24	输出数据像素24 bits
GDC_OUT_PIXEL_BITS_NOT_SUPPORT = 0xFF	不支持

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphic_dc.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.22.3.8 app_graphics_dc_frame_timing_e

枚举定义如下：

表 2-489 app_graphics_dc_frame_timing_e枚举成员

成员	描述
GDC_SPI_FRAME_TIMING_0 = 0x00	时序构成为8 bits控制命令+24 bits控制地址+若干字节数据，全部通过SPI模式发出
GDC_DSPI_FRAME_TIMING_0	时序构成为8 bits控制命令+无地址+若干字节数据，均通过4线SPI模式发出，并且启用DCX信号
GDC_QSPI_FRAME_TIMING_0	时序构成为8 bits控制命令+24 bits控制地址+若干字节数据，均通过Quad SPI模式发出
GDC_QSPI_FRAME_TIMING_1	时序构成为8 bits控制命令+24 bits控制地址+若干字节数据。其中命令使用SPI模式发出，地址和数据部分通过Quad SPI模式发出

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphic_dc.h。

2.22.3.9 app_graphics_dc_frame_address_width_e

枚举定义如下：

表 2-490 app_graphics_dc_frame_address_width_e枚举成员

成员	描述
GDC_FRAME_ADDRESS_WIDTH_NONE = 0xFF	无控制地址
GDC_FRAME_ADDRESS_WIDTH_08BIT = MIPI_CMD08	控制地址宽度为8 bits
GDC_FRAME_ADDRESS_WIDTH_16BIT = MIPI_CMD16	控制地址宽度为16 bits
GDC_FRAME_ADDRESS_WIDTH_24BIT = MIPI_CMD24	控制地址宽度为24 bits

详细信息请参考文件：SDK_Folder\rivers\inc\app_graphic_dc.h。

2.22.3.10 app_graphics_dc_access_type_e

枚举定义如下：

表 2-491 app_graphics_dc_access_type_e枚举成员

成员	描述
GDC_ACCESS_TYPE_SYNC = 0	接口使用同步方式工作，等到刷屏完成接口才返回
GDC_ACCESS_TYPE_ASYNC	接口使用异步方式工作，通过回调函数获取刷屏结果

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphic_dc.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.22.3.11 app_graphics_dc_frame_result_e

枚举定义如下：

表 2-492 app_graphics_dc_frame_result_e枚举成员

成员	描述
GDC_FRAME_RES_SUCCESS = 0x00	帧数据发送成功
GDC_FRAME_RES_ASYNC_WAIT	发送一帧数据，等待应答信号
GDC_FRAME_RES_FAIL	帧数据发送失败
GDC_FRAME_RES_UNSUPPORT	帧格式或者命令不支持

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphic_dc.h。

2.22.3.12 graphics_dc_power_state_e

枚举定义如下：

表 2-493 graphics_dc_power_state_e枚举成员

成员	描述
GDC_POWER_STATE_SLEEP = 0	DC睡眠状态
GDC_POWER_STATE_ACTIVE = 1	DC激活状态

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphic_dc.h。

2.22.3.13 graphics_dc_layer_mode_e

枚举定义如下：

表 2-494 graphics_dc_layer_mode_e枚举成员

成员	描述
GDC_ONE_LAYER_MODE = 0	图层1
GDC_TWO_LAYER_MODE = 1	图层2

详细信息请参考文件：SDK_Folder\drivers\inc\app_graphics_dc.h。

2.22.4 接口函数

表 2-495 DC驱动接口函数

API	芯片	
	GR551x/GR5x25/GR533x/GR5405	GR5526
graphics_dc_init	N	Y
app_graphics_dc_spi_send	N	Y
app_graphics_dc_dspi_send_cmd_in_3wire_1lane	N	Y

版权所有 © 2025 深圳市汇顶科技股份有限公司

API	芯片	
	GR551x/GR5x25/GR533x/GR5405	GR5526
app_graphics_dc_dspi_send_cmd_data_in_3wire_1lane	N	Y
app_graphics_dc_dspi_send_cmd_data_in_4wire_2lane	N	Y
app_graphics_dc_dspi_send_cmd_datas_in_4wire_2lane	N	Y
app_graphics_dc_send_single_frame	N	Y
app_graphics_dc_set_power_state	N	Y
app_graphics_dc_freq_set	N	Y

2.22.4.1 graphics_dc_init

表 2-496 graphics_dc_init接口

函数原型	uint16_t graphics_dc_init(app_graphics_dc_params_t * dc_params, graphics_dc_irq_event_notify_cb evt_cb)
功能说明	初始化DC
输入参数	• p_params：初始化参数的结构体指针 • evt_cb：用户自定义回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.22.4.2 app_graphics_dc_spi_send

表 2-497 app_graphics_dc_spi_send接口

函数原型	void app_graphics_dc_spi_send(uint8_t cmd_8bit, uint32_t address_24bit, uint8_t * data, uint32_t length)
功能说明	DC通过标准SPI协议发送1 byte命令，3 bytes地址，N bytes数据
输入参数	• cmd_8bit：需要发送的命令 • address_24bit：需要发送的地址 • data：需要发送的N bytes数据指针 • length：数据长度，以Byte为单位
返回值	
备注	

2.22.4.3 app_graphics_dc_dspi_send_cmd_in_3wire_1lane

表 2-498 app_graphics_dc_dspi_send_cmd_in_3wire_1lane接口

函数原型	void app_graphics_dc_dspi_send_cmd_in_3wire_1lane(uint8_t cmd)
功能说明	DC通过DSPI 3W1L模式发送1 byte命令，未使用DCX线，时序上会在最高位添加1 bit命令指示位
输入参数	cmd：1 byte命令
返回值	
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.22.4.4 app_graphics_dc_dspi_send_cmd_data_in_3wire_1lane

表 2-499 app_graphics_dc_dspi_send_cmd_data_in_3wire_1lane接口

函数原型	void app_graphics_dc_dspi_send_cmd_data_in_3wire_1lane(uint8_t cmd, uint8_t data)
功能说明	DC通过DSPI 3W1L模式发送1 byte命令和1 byte数据，未使用DCX线，时序上会在命令和数据的最高位添加1 bit指示位，表示命令还是数据
输入参数	• cmd：1 byte命令 • data：1 byte数据
返回值	
备注	

2.22.4.5 app_graphics_dc_dspi_send_cmd_data_in_4wire_2lane

表 2-500 app_graphics_dc_dspi_send_cmd_data_in_4wire_2lane接口

函数原型	void app_graphics_dc_dspi_send_cmd_data_in_4wire_2lane(uint16_t cmd, uint16_t data)
功能说明	DC通过DSPI 4W2L模式发送2 bytes命令和2 bytes数据，DCX信号线作为数据传输，时序上会在命令和数据的最高位添加1 bit指示位，表示命令还是数据
输入参数	• cmd：2 bytes命令 • data：2 bytes数据
返回值	
备注	数据线1传输高字节，DCX信号线传输低字节

2.22.4.6 app_graphics_dc_dspi_send_cmd_datas_in_4wire_2lane

表 2-501 app_graphics_dc_dspi_send_cmd_datas_in_4wire_2lane接口

函数原型	void app_graphics_dc_dspi_send_cmd_datas_in_4wire_2lane(uint16_t cmd, uint16_t * data , int length);
功能说明	在4-Wire SPI模式下发送1个命令字和多个数据，主要用于屏幕的初始化配置
输入参数	• cmd：命令字，低8位补0扩展为16-bit • data： 数据指针，各个数据低8位补0扩展为16-bit • length ： 数据个数
返回值	无
备注	单次传输数据个数不超过8个

2.22.4.7 app_graphics_dc_send_single_frame

表 2-502 app_graphics_dc_send_single_frame接口

函数原型	app_graphics_dc_frame_result_e app_graphics_dc_send_single_frame(uint32_t which_layer, app_graphics_dc_framelayer_t * frame_layer, app_graphics_dc_cmd_t * dc_cmd, app_graphics_dc_access_type_e access_type)
功能说明	DC发送一帧图像数据

版权所有 © 2025 深圳市汇顶科技股份有限公司

输入参数	• which_layer：DC渲染图层（目前暂只支持GRAPHICS_DC_LAYER_0） • frame_layer： 输入帧缓冲区配置 • dc_cmd：DC输出控制命令 • access_type： 接口调用类型 （异步/同步传输）
返回值	DC模块帧数据传输结果
备注	

2.22.4.8 app_graphics_dc_set_power_state

表 2-503 app_graphics_dc_set_power_state接口

函数原型	void app_graphics_dc_set_power_state(graphics_dc_power_state_e state)
功能说明	设置DC的工作状态
输入参数	state：DC的工作状态，参考2.22.3.12 graphics_dc_power_state_e
返回值	
备注	

2.22.4.9 app_graphics_dc_freq_set

表 2-504 app_graphics_dc_freq_set接口

函数原型	void app_graphics_dc_freq_set(graphics_dc_clock_freq_e clock_freq)
功能说明	设置DC的输出时钟频率
输入参数	clock_freq：时钟频率，参考2.22.3.2 graphics_dc_clock_freq_e
返回值	
备注	

2.23 APP QSPI驱动

2.23.1 宏定义

2.23.1.1 APP_QSPI_PIN_ENABLE

表 2-505 APP_QSPI_PIN_ENABLE宏定义

描述	值
QSPI 引脚使能	1

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.23.1.2 APP_QSPI_PIN_DISABLE

表 2-506 APP_QSPI_PIN_DISABLE宏定义

描述	值
引脚不使能	0

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.1.3 QSPI_MAX_XFER_SIZE_ONCE

表 2-507 QSPI_MAX_XFER_SIZE_ONCE宏定义

描述	值
QSPI每次最大传输节拍数	0xFFFCu

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.1.4 DMA_MAX_XFER_SIZE_ONCE

表 2-508 DMA_MAX_XFER_SIZE_ONCE宏定义

描述	值
DMA通道每个block最大传输字节数	4095u

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.1.5 APP_STORAGE_RAM_ID

表 2-509 APP_STORAGE_RAM_ID宏定义

描述	值
存放在RAM上资源的特殊ID	0xf

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.1.6 QSPI_ASYNC_SCROLL_DRAW_SCREEN_SUPPORT

表 2-510 QSPI_ASYNC_SCROLL_DRAW_SCREEN_SUPPORT宏定义

描述	值
支持/不支持在异步模式下用两个独立帧缓冲区绘制整个屏幕	0u：不支持 1u：支持

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi_user_config.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.23.1.7 QSPI_ASYNC_VERI_LINK_DRAW_SCREEN_SUPPORT

表 2-511 QSPI_ASYNC_VERI_LINK_DRAW_SCREEN_SUPPORT宏定义

描述	值
支持/不支持在异步模式下使用链表方式绘制整个屏幕	0u：不支持 1u：支持

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi_user_config.h。

2.23.1.8 QSPI_BLIT_RECT_IMAGE_SUPPORT

表 2-512 QSPI_BLIT_RECT_IMAGE_SUPPORT宏定义

描述	值
支持/不支持以DMA LLP/SG方式Blit矩形区域	0u：不支持 1u：支持

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi_user_config.h。

2.23.1.9 QSPI_DMA_LLP_FEATUTE_SUPPORT

表 2-513 QSPI_DMA_LLP_FEATUTE_SUPPORT宏定义

描述	值
是否启用DMA LLP特征	由QSPI_SYNC_SCROLL_DRAW_SCREEN_SUPPORT || QSPI_ASYNC_SCROLL_DRAW_SCREEN_SUPPORT || QSPI_ASYNC_VERI_LINK_DRAW_SCREEN_SUPPORT确定

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi_user_config.h。

2.23.1.10 DMA_LLP_BLOCKS_FOR_WRITE

表 2-514 DMA_LLP_BLOCKS_FOR_WRITE宏定义

描述	值
app_qspi_async_draw_screen每次使用DMA LLP方式写的最大块数量，越大获得更高的速度但花费更多的RAM空间	取值范围16 ～ 64，且是4的倍数

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi_user_config.h。

2.23.1.11 DMA_LLP_BLOCKS_FOR_BLIT

表 2-515 DMA_LLP_BLOCKS_FOR_BLIT宏定义

描述	值
app_qspi_mmap_blit_image每次使用DMA LLP方式传输的最大块数量，越大获得更高的速度但花费更多的RAM空间	取值范围16 ～ 64，且是4的倍数

版权所有 © 2025 深圳市汇顶科技股份有限公司

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi_user_config.h。

2.23.2 结构体

2.23.2.1 app_qspi_params_t

结构体定义如下：

表 2-516 app_qspi_params_t结构体成员

成员	描述	取值
app_qspi_id_t id	设置使用的QSPI模块ID	该参数的取值可以是下列值中的任意一个： • APP_QSPI_ID_0 • APP_QSPI_ID_1 • APP_QSPI_ID_2
app_qspi_pin_cfg_t pin_cfg	引脚配置	参考2.23.2.2 app_qspi_pin_cfg_t结构体
app_qspi_dma_cfg_t dma_cfg	DMA参数设置	参考2.23.2.4 app_qspi_dma_cfg_t结构体
qspi_init_t init	初始化结构体	参考2.23.2.6 qspi_init_t结构体
qspi_env_t qspi_env	仅用于记录初始化设备的句柄、相关参数和信号量等，不需要配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.2 app_qspi_pin_cfg_t

结构体定义如下：

表 2-517 app_qspi_pin_cfg_t结构体成员

成员	描述	取值
app_qspi_pin_t cs	片选引脚配置	参考2.23.2.3 app_qspi_pin_t结构体
app_qspi_pin_t clk	时钟引脚配置	参考2.23.2.3 app_qspi_pin_t结构体
app_qspi_pin_t io_0	IO0引脚	参考2.23.2.3 app_qspi_pin_t结构体
app_qspi_pin_t io_1	IO1引脚	参考2.23.2.3 app_qspi_pin_t结构体
app_qspi_pin_t io_2	IO2引脚	参考2.23.2.3 app_qspi_pin_t结构体
app_qspi_pin_t io_3	IO3引脚	参考2.23.2.3 app_qspi_pin_t结构体

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.3 app_qspi_pin_t

结构体定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-518 app_qspi_pin_t结构体成员

成员	描述	取值
app_io_type_t type	IO类型	同2.2.1.2 app_io_evt_t定义
app_io_mux_t mux	设置IO映射功能	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考Datasheet中的Pin Mux
uint32_t pin	选择IO引脚	查看2.2.1.1 app_io_init_t结构体定义，具体取值参考Datasheet中的Pin Mux
app_io_mode_t mode	IO模式配置	APP_IO_MODE_MUX
app_io_pull_t pull	设置IO上拉/下拉/浮空	同2.2.1.1 app_io_init_t结构体中的描述
uint8_t enable	使能/不使能引脚	• true：使能 • false：不使能

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.4 app_qspi_dma_cfg_t

结构体定义如下：

表 2-519 app_qspi_dma_cfg_t结构体成员

成员	描述	取值
dma_regs_t * dma_instance	DMA实例	参考2.4.1.1 app_dma_params_t章节中，DMA实例描述
dma_channel_t dma_channel	DMA通道	参考2.4.1.1 app_dma_params_t中通道描述
uint32_t wait_timeout_ms	轮询和死机等待的超时时间，仅适用于GR5526	
uint32_t extend	扩展段，仅适用于GR5526	暂未使用

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.5 app_qspi_command_t

结构体定义如下：

表 2-520 app_qspi_command_t结构体成员

成员	描述	取值
uint32_t instruction	访问从设备指令	具体根据从设备访问时序确定
uint32_t address	访问从设备地址	具体根据从设备访问时序确定
uint32_t instruction_size	访问指令宽度	该参数的取值可以是下列值中的任意一个： • QSPI_INSTSIZE_00_BITS（无指令） • QSPI_INSTSIZE_08_BITS（指令宽度8 bits） • QSPI_INSTSIZE_16_BITS（指令宽度16 bits）
uint32_t address_size	访问地址宽度	该参数的取值可以是下列值中的任意一个：

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• QSPI_ADDRSIZE_00_BITS（无地址部分） • QSPI_ADDRSIZE_08_BITS（地址宽度8 bits） • QSPI_ADDRSIZE_16_BITS（地址宽度16 bits） • QSPI_ADDRSIZE_24_BITS（地址宽度24 bits） • QSPI_ADDRSIZE_32_BITS（地址宽度32 bits）
uint32_t dummy_cycles	等待时钟周期数	0 ～ 31，具体根据从设备时序决定
uint32_t data_size	数据传输宽度	该参数的取值可以是下列值中的任意一个： • QSPI_DATASIZE_08_BITS（数据采用8 bits宽度传输） • QSPI_DATASIZE_16_BITS（数据采用16 bits宽度传输） • QSPI_DATASIZE_32_BITS（数据采用32 bits宽度传输）
uint32_t instruction_address_mode	指令地址发送采用的时序模式	该参数的取值可以是下列值中的任意一个： • QSPI_INST_ADDR_ALL_IN_SPI（指令/地址均使用SPI模式传输） • QSPI_INST_IN_SPI_ADDR_IN_SPIFRF（指令采用SPI模式传输；地址传输模式由data_mode决定） • QSPI_INST_ADDR_ALL_IN_SPIFRF（指令地址的传输模式均由data_mode决定）
uint32_t data_mode	数据传输采用的时序模式	该参数的取值可以是下列值中的任意一个： • QSPI_DATA_MODE_SPI（使用SPI模式） • QSPI_DATA_MODE_DUALSPI（使用Dual SPI模式） • QSPI_DATA_MODE_QUADSPI（使用Quad SPI模式）
uint32_t length	传输数据长度，单位字节，需为data_size的整数倍	最大65535字节
uint32_t clock_stretch_en	是否启用钟摆功能，只对Dual SPI和Quad SPI模式有效	• true：启用钟摆功能 • false：不启用钟摆功能

2.23.2.6 qspi_init_t

结构体定义如下：

表 2-521 qspi_init_t结构体成员

成员	描述	取值
uint32_t clock_prescaler	基于总线时钟生成QSPI SCLK时钟的分频系数	支持2 ～ 65535之间的偶数
uint32_t clock_mode	时钟极性及相位模式	该参数的取值可以是下列值中的任意一个： • QSPI_CLOCK_MODE_0（有效状态为clk处于低电平时，数据采样在第一个边沿） • QSPI_CLOCK_MODE_1（有效状态为clk处于低电平时，数据采样在第二个边沿）

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• QSPI_CLOCK_MODE_2（有效状态为clk处于高电平时，数据采样在第一个边沿） • QSPI_CLOCK_MODE_3（有效状态为clk处于高电平时，数据采样在第二个边沿）
uint32_t rx_sample_delay	RX输入端口的采样延时，以QSPI SCLK时钟周期为单位，防止频率过高，数据传输错误	0 ～ 0x7

详细信息请参考文件：SDK_Folder\drivers\inc\hal\gr55xx_hal_qspi.h。

2.23.2.7 app_qspi_mmap_device_t

结构体定义如下：

表 2-522 app_qspi_mmap_device_t结构体成员

成员	描述	取值
app_qspi_device_e dev_type	设备类型	该参数的取值可以是下列值中的任意一个： • APP_QSPI_DEVICE_UNSET = 0（无设置） • APP_QSPI_DEVICE_FLASH = 1（Flash设备） • APP_QSPI_DEVICE_PSRAM = 2（PSRAM设备）
app_qspi_psram_mmap_wr_cmd_e psram_wr	内存映射模式下，PSRAM的写命令	该参数的取值可以是下列值中的任意一个： • PSRAM_MMAP_CMD_QWRITE_02H = 0x00 • PSRAM_MMAP_CMD_QWRITE_38H = 0x01 • PSRAM_MMAP_CMD_WRITE_MAX
union { app_qspi_flash_mmap_rd_cmd_e flash_rd; app_qspi_psram_mmap_rd_cmd_e psram_rd; } rd;	• 读Flash命令； • 读PSRAM命令	Flash读命令支持： • FLASH_MMAP_CMD_DREAD_3BH = 0x00 • FLASH_MMAP_CMD_2READ_BBH = 0x01 • FLASH_MMAP_CMD_QREAD_6BH = 0x03 • FLASH_MMAP_CMD_4READ_EBH = 0x04 PSRAM读命令支持： • PSRAM_MMAP_CMD_QREAD_0BH • PSRAM_MMAP_CMD_QREAD_EBH
void * set	保留位	

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.8 app_qspi_screen_command_t

结构体定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-523 app_qspi_screen_command_t结构体成员

成员	描述	取值
uint32_t instruction	刷屏控制命令	0x00 ～ 0xff的8 bits数据
uint32_t leading_address	刷屏前导地址，如0x002C00	0x00000000 ～ 0xffffffff的32 bits地址数据
uint32_t ongoing_address	刷屏跟续地址，如0x003C00	0x00000000 ～ 0xffffffff的32 bits地址数据
uint32_t instruction_size	指定指令的size	根据具体屏幕芯片手册决定，可选值参考QSPI_Instruction_Size
uint32_t address_size	刷屏地址Size	根据具体屏幕芯片手册决定，可选值参考QSPI_Address_Size
uint32_t dummy_cycles	Dummy(Wait) cycles	根据屏幕芯片手册决定，可选值0 ～ 31
uint32_t data_size	数据传输宽度	参考QSPI_Data_Size，支持8/16/32 bits宽度传输
uint32_t instruction_address_mode	指令地址发送的时序模式	该参数的取值可以是下列值中的任意一个： • QSPI_INST_ADDR_ALL_IN_SPI（地址/指令均使用单线模式发送） • QSPI_INST_IN_SPI_ADDR_IN_SPIFRF（指令使用单线模式发送，地址使用data_mode模式发送） • QSPI_INST_ADDR_ALL_IN_SPIFRF （指令/地址均使用data_mode模式发送）
uint32_t data_mode	数据发送模式	该参数的取值可以是下列值中的任意一个： • QSPI_DATA_MODE_SPI（数据使用单线模式发送） • QSPI_DATA_MODE_DUALSPI（数据使用Dual模式发送） • QSPI_DATA_MODE_QUADSPI（数据使用Quad模式发送）
bool is_one_take_cs	是否启用一次性片选发送完所有数据，需要屏幕支持；如果启用，本次所有数据会在一个CS周期内发送完成	• true：使能 • false：不使能

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.9 app_qspi_evt_t

结构体定义如下：

表 2-524 app_qspi_evt_t结构体成员

成员	描述	取值
app_qspi_evt_type_t type	QSPI事件类型	该参数的取值可以是下列值中的任意一个： • APP_QSPI_EVT_ERROR（QSPI传输错误） • APP_QSPI_EVT_TX_CPLT（QSPI发送完成） • APP_QSPI_EVT_RX_DATA（QSPI接收完成） • APP_QSPI_EVT_ABORT（QSPI中断发送）

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• APP_QSPI_EVT_ASYNC_WR_SCRN_CPLT（QSPI异步传输屏幕数据完成） • APP_QSPI_EVT_ASYNC_WR_SCRN_FAIL（QSPI异步传输屏幕数据失败）
union { uint32_t error_code; uint16_t size; }data;	uint32_t error_code：发送事件返回的错误号 uint16_t size：QSPI已发送的数据长度 **说明：** 若事件类型为传输错误，此处返回错误码；若事件类型为发送完成，此处显示已发送的数据长度	QSPI错误码： • 0：无异常 • 1：超时错误 • 2：传输错误 • 4：DMA错误 • 5：参数设置异常

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.10 app_qspi_screen_info_t

结构体定义如下：

表 2-525 app_qspi_screen_info_t结构体成员

成员	描述	取值
unsigned int scrn_pixel_width	水平像素，单位：pixel	根据屏幕分辨率决定
unsigned int scrn_pixel_height	垂直像素，单位：pixel	根据屏幕分辨率决定
unsigned int scrn_pixel_depth	像素深度，单位：Byte	根据采用的像素色深决定

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.11 blit_image_config_t

结构体定义如下：

表 2-526 blit_image_config_t结构体成员

成员	描述	取值
uint32_t src_img_address	源对象总线地址，需要4字节对齐	
uint32_t src_img_w	源对象像素宽度	
uint32_t src_img_h	源对象像素高度	
uint32_t src_img_x	源对象拟截取x起始坐标	源对象的左上角为源对象坐标系原点(0,0)
uint32_t src_img_x_delta	源对象拟截取的像素宽度	注意x加x_delta不能超过源对象宽度

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
uint32_t src_img_y	源对象拟截取y起始坐标	源对象的左上角为源对象坐标系原点(0,0)
uint32_t src_img_y_delta	源对象拟截取的像素高度	注意y加y_delta不能超过源对象高度
uint32_t dst_buff_address	目标对象总线地址，需要4字节对齐	
uint32_t dst_buff_width	目标对象像素宽度	
uint32_t dst_buff_height	目标对象像素高度	
uint32_t dst_buff_x	目标对象待写入x起始坐标	目标对象的左上角为源对象坐标系原点(0,0)
uint32_t dst_buff_y	目标对象待写入y起始坐标	目标对象的左上角为源对象坐标系原点(0,0)
uint32_t pixel_depth	像素深度，以Byte为单位	

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.12 app_qspi_screen_scroll_t

结构体定义如下：

表 2-527 app_qspi_screen_scroll_t结构体成员

成员	描述	取值
uint32_t first_frame_start_address	第一帧起始总线地址 • 如果水平滚动屏幕：第一帧代表左图像 • 如果垂直滚动屏幕：第一帧代表向上图像	第一帧缓存地址
uint32_t second_frame_start_address	第二帧起始总线地址 • 如果水平滚动屏幕：第二帧代表右图像 • 如果垂直滚动屏幕：第二帧代表向下图像	第二帧缓存地址
uint32_t scroll_coordinate	滚动点坐标 • 如果水平滚动，为x坐标 • 如果垂直滚动，为y坐标	• 如果水平滚动，范围为[0, RES_X]；取值为0时，只刷第一帧；取值为RES_X时，只刷第二帧；否则左右各出一部分 • 如果垂直滚动，范围为[0, RES_Y]；取值为0时，只刷第一帧；取值为RES_Y时，只刷第二帧；否则上下各出一部分
bool is_horizontal_scroll	是否水平滚动	• true：水平 • false：垂直

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.13 app_qspi_screen_veri_link_scroll_t

结构体定义如下：

表 2-528 app_qspi_screen_veri_link_scroll_t结构体成员

成员	描述	取值
uint32_tframe_ahb_start_addres	此节点帧的起始地址	存放节点的总线地址
uint32_t frame_offset_lines	此节点帧位于全帧的偏移行数	
uint32_t frame_draw_lines	此节点帧的总行数	总行数根据LLP size决定，一般不超过50行
struct _screen_veri_link_scroll_t * next	指向下一个节点帧地址	最后一个设置为NULL

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.2.14 app_qspi_screen_block_t

表 2-529 app_qspi_screen_block_t结构体成员

成员	描述	取值
uint32_t frame_ahb_start_address	待刷新屏幕区域的图像块起始地址	
uint32_t frame_offset_lines	待刷新屏幕区域的图像块行偏移	
uint32_t frame_draw_lines	待刷新屏幕区域的图像块行数量	

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3 枚举

2.23.3.1 app_qspi_id_t

枚举定义如下：

表 2-530 app_qspi_id_t枚举成员

成员	描述
APP_QSPI_ID_0	QSPI模块0
APP_QSPI_ID_1	QSPI模块1
APP_QSPI_ID_2	QSPI模块2
APP_QSPI_ID_MAX	QSPI模块ID总数，用于参数检查

2.23.3.2 app_qspi_evt_type_t

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-531 app_qspi_evt_type_t枚举成员

成员	描述
APP_QSPI_EVT_ERROR	QSPI传输错误
APP_QSPI_EVT_TX_CPLT	QSPI发送完成
APP_QSPI_EVT_RX_DATA	QSPI接收完成
APP_QSPI_EVT_ABORT	QSPI中断发送
APP_QSPI_EVT_ASYNC_WR_SCRN_CPLT	QSPI异步传输Quad屏幕数据完成
APP_QSPI_EVT_ASYNC_WR_SCRN_FAIL	QSPI异步传输Quad屏幕数据失败

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.3 app_qspi_mmap_endian_mode_e

枚举定义如下：

表 2-532 app_qspi_mmap_endian_mode_e枚举成员

成员	描述
APP_QSPI_MMAP_ENDIAN_MODE_0 = 0	QSPI内存映射字节存储次序模式0
APP_QSPI_MMAP_ENDIAN_MODE_1	QSPI内存映射字节存储次序模式1
APP_QSPI_MMAP_ENDIAN_MODE_2	QSPI内存映射字节存储次序模式2

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

说明:

该枚举仅适用于GR5526，每个模式对应的字节存储次序请参考《GR5526 GPU开发者指南》

2.23.3.4 app_qspi_device_e

枚举定义如下：

表 2-533 app_qspi_device_e枚举成员

成员	描述
APP_QSPI_DEVICE_UNSET = 0	QSPI设备类型未设置
APP_QSPI_DEVICE_FLASH = 1	Flash设备
APP_QSPI_DEVICE_PSRAM = 2	PSRAM设备

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.5 app_qspi_psram_mmap_wr_cmd_e

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-534 app_qspi_psram_mmap_wr_cmd_e枚举成员

成员	描述
PSRAM_MMAP_CMD_QWRITE_02H = 0x00	Quad模式写0x02H指令
PSRAM_MMAP_CMD_QWRITE_38H = 0x01	Quad模式写0x38H指令
PSRAM_MMAP_CMD_WRITE_MAX	Quad模式写指令个数，仅用于参数检查

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.6 app_qspi_psram_mmap_rd_cmd_e

枚举定义如下：

表 2-535 app_qspi_psram_mmap_rd_cmd_e枚举成员

成员	描述
PSRAM_MMAP_CMD_QREAD_0BH = 0x00	Quad读0x0B指令
PSRAM_MMAP_CMD_QREAD_EBH = 0x01	Quad读0xEB指令
PSRAM_MMAP_CMD_READ_MAX	Quad模式读指令个数，仅用于参数检查

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.7 app_qspi_flash_mmap_rd_cmd_e

枚举定义如下：

表 2-536 app_qspi_flash_mmap_rd_cmd_e枚举成员

成员	描述
FLASH_MMAP_CMD_DREAD_3BH = 0x00	Flash Dual读0x3B指令
FLASH_MMAP_CMD_2READ_BBH = 0x01	Flash Dual读0xBB指令
FLASH_MMAP_CMD_2READ_BBH_SIOO = 0x02	Flash Dual读0xBB指令，以SIO0模式
FLASH_MMAP_CMD_QREAD_6BH = 0x03	Flash Quad读0x6B指令
FLASH_MMAP_CMD_4READ_EBH = 0x04	Flash Quad读0xEB指令
FLASH_MMAP_CMD_4READ_EBH_SIOO = 0x05	Flash Quad读0xEB指令，以SIO0模式
FLASH_MMAP_CMD_READ_MAX	Flash读命令总数，用于参数检查

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.8 qspi_pins_group_e

枚举定义如下：

版权所有 © 2025 深圳市汇顶科技股份有限公司

表 2-537 qspi_pins_group_e枚举成员

成员	描述
QSPI0_PIN_GROUP_0 = 0x00	指定使用QSPI0的Pin引脚组
QSPI1_PIN_GROUP_0 = 0x01	指定使用QSPI1的Pin引脚组
QSPI2_PIN_GROUP_0 = 0x02	指定使用QSPI2的Pin引脚组
QSPIx_PIN_GROUP_MAX	Pin引脚总数，用于参数检查

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.9 app_qspi_dma_state_t

枚举定义如下：

表 2-538 app_qspi_dma_state_t枚举成员

成员	描述
APP_QSPI_DMA_INVALID	QSPI DMA不可用
APP_QSPI_DMA_ACTIVITY	QSPI DMA激活

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.10 app_qspi_state_t

枚举定义如下：

表 2-539 app_qspi_state_t枚举成员

成员	描述
APP_QSPI_INVALID = 0	QSPI不可用
APP_QSPI_ACTIVITY = 1	QSPI激活
APP_QSPI_SLEEP = 2	QSPI睡眠

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

2.23.3.11 blit_xfer_type_e

枚举定义如下：

表 2-540 blit_xfer_type_e枚举成员

成员	描述
BLIT_BY_DMA_SG = 0	使用Scatter&Gather方式
BLIT_BY_DMA_LLP = 1	使用LLP方式

详细信息请参考文件：SDK_Folder\drivers\inc\app_qspi.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.23.4 接口函数

表 2-541 QSPI驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_qspi_init	Y	Y	Y	N	N
app_qspi_deinit	Y	Y	Y	N	N
app_qspi_abort	Y	Y	Y	N	N
app_qspi_config_memory_mappped	N	Y	Y	N	N
app_qspi_active_memory_mappped	N	Y	Y	N	N
app_qspi_command_receive_sync	Y	Y	Y	N	N
app_qspi_command_receive_async	Y	Y	Y	N	N
app_qspi_command_transmit_sync	Y	Y	Y	N	N
app_qspi_command_transmit_async	Y	Y	Y	N	N
app_qspi_command_sync	Y	Y	Y	N	N
app_qspi_command_async	Y	Y	Y	N	N
app_qspi_transmit_sync	Y	N	N	N	N
app_qspi_transmit_async	Y	N	N	N	N
app_qspi_receive_sync	Y	N	N	N	N
app_qspi_receive_async	Y	N	N	N	N
app_qspi_transmit_sync_ex	Y	Y	Y	N	N
app_qspi_transmit_async_ex	Y	Y	Y	N	N
app_qspi_receive_sync_ex	Y	Y	Y	N	N
app_qspi_receive_async_ex	Y	Y	Y	N	N
app_qspi_mmap_set_endian_mode	N	Y	Y	N	N
app_qspi_mmap_read_u8	N	Y	Y	N	N
app_qspi_mmap_read_u16	N	Y	Y	N	N
app_qspi_mmap_read_u32	N	Y	Y	N	N
app_qspi_mmap_read_block	N	Y	Y	N	N
app_qspi_get_xip_base_address	N	Y	Y	N	N
app_qspi_transmit_in_qpi_async	Y	N	N	N	N
app_qspi_get_handle	Y	Y	Y	N	N

2.23.4.1 app_qspi_init

表 2-542 app_qspi_init接口

函数原型	uint16_t app_qspi_init(app_qspi_params_t *p_params, app_qspi_evt_handler_t evt_handler)
功能说明	初始化QSPI

版权所有 © 2025 深圳市汇顶科技股份有限公司

输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_qspi_evt_handler_t)(app_qspi_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户无需关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可

2.23.4.2 app_qspi_deinit

表 2-543 app_qspi_deinit接口

函数原型	uint16_t app_qspi_deinit(app_qspi_id_t id)
功能说明	反初始化QSPI
输入参数	id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.4.3 app_qspi_abort

表 2-544 app_qspi_abort接口

函数原型	uint16_t app_qspi_abort(app_qspi_id_t id)
功能说明	中止QSPI传输
输入参数	id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.4.4 app_qspi_config_memory_mappped

表 2-545 app_qspi_config_memory_mappped接口

函数原型	bool app_qspi_config_memory_mappped(app_qspi_id_t id, app_qspi_mmap_device_t dev);
功能说明	主要用于块数据的发送
输入参数	• id：QSPI 模块ID • dev：内存映射模式设备配置
返回值	true/false
备注	

2.23.4.5 app_qspi_active_memory_mappped

表 2-546 app_qspi_active_memory_mappped接口

函数原型	bool app_qspi_active_memory_mappped(app_qspi_id_t id, bool is_active)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	使能/不使能内存映射模式
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • is_active： ◦ true：使能 ◦ false：不使能
返回值	• true：使能/不使能成功 • false：使能/不使能失败
备注	

2.23.4.6 app_qspi_command_receive_sync

表 2-547 app_qspi_command_receive_sync接口

函数原型	uint16_t app_qspi_command_receive_sync(app_qspi_id_t id, app_qspi_command_t *p_cmd, uint8_t *p_data, uint32_t timeout)
功能说明	QSPI轮询方式（同步）读取数据，带控制命令封装
输入参数	• id：QSPI模块ID • p_cmd：封装的控制命令，参考2.23.2.5 app_qspi_command_t结构体描述 • p_data：读取数据存放的buffer • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.4.7 app_qspi_command_receive_async

表 2-548 app_qspi_command_receive_async接口

函数原型	uint16_t app_qspi_command_receive_async(app_qspi_id_t id, app_qspi_command_t *p_cmd, uint8_t *p_data)
功能说明	QSPI中断方式（异步）读取数据，带控制命令封装
输入参数	• id：QSPI模块ID • p_cmd：封装的控制命令，参考2.23.2.5 app_qspi_command_t结构体描述 • p_data：读取数据存放的buffer
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.4.8 app_qspi_command_transmit_sync

表 2-549 app_qspi_command_transmit_sync接口

函数原型	uint16_t app_qspi_command_transmit_sync(app_qspi_id_t id, app_qspi_command_t *p_cmd, uint8_t *p_data, uint32_t timeout)
功能说明	QSPI轮询方式（同步）发送数据，带控制命令封装
输入参数	• id：QSPI模块ID • p_cmd：封装的控制命令，参考2.23.2.5 app_qspi_command_t结构体描述

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• p_data：待发送数据存放的buffer • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.4.9 app_qspi_command_transmit_async

表 2-550 app_qspi_command_transmit_async接口

函数原型	uint16_t app_qspi_command_transmit_async(app_qspi_id_t id, app_qspi_command_t *p_cmd, uint8_t *p_data)
功能说明	QSPI中断方式（异步）发送数据，带控制命令封装
输入参数	• id：QSPI模块ID • p_cmd：封装的控制命令，参考2.23.2.5 app_qspi_command_t结构体描述 • p_data：待发送数据存放的buffer
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.4.10 app_qspi_command_sync

表 2-551 app_qspi_command_sync接口

函数原型	uint16_t app_qspi_command_sync(app_qspi_id_t id, app_qspi_command_t *p_cmd, uint32_t timeout)
功能说明	QSPI轮询方式（同步）发送命令
输入参数	• id：QSPI模块ID • p_cmd：待发送命令的buffer • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.4.11 app_qspi_command_async

表 2-552 app_qspi_command_async接口

函数原型	uint16_t app_qspi_command_async(app_qspi_id_t id, app_qspi_command_t *p_cmd)
功能说明	QSPI中断方式（异步）发送命令
输入参数	• id：QSPI模块ID • p_cmd：待发送命令的buffer
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.23.4.12 app_qspi_transmit_sync

表 2-553 app_qspi_transmit_sync接口

函数原型	uint16_t app_qspi_transmit_sync(app_qspi_id_t id, uint8_t *p_data, uint32_t length, uint32_t timeout)
功能说明	QSPI轮询方式（同步）发送数据
输入参数	• id：QSPI模块ID • p_data：需要发送数据的buffer • length：发送数据的长度，以Byte为单位 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	该函数仅适用于GR551x

2.23.4.13 app_qspi_transmit_async

表 2-554 app_qspi_transmit_async接口

函数原型	uint16_t app_qspi_transmit_async(app_qspi_id_t id, uint8_t *p_data, uint32_t length)
功能说明	QSPI中断方式（异步）发送数据
输入参数	• id：QSPI模块ID • p_data：需要发送数据的buffer • length：待发送数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	该函数仅适用于GR551x

2.23.4.14 app_qspi_receive_sync

表 2-555 app_qspi_receive_sync接口

函数原型	uint16_t app_qspi_receive_sync(app_qspi_id_t id, uint8_t *p_data, uint32_t length, uint32_t timeout)
功能说明	QSPI轮询方式（同步）接收数据
输入参数	• id：QSPI模块ID • p_data：接收数据的buffer • length：接收数据的长度，以Byte为单位 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	该函数仅适用于GR551x

2.23.4.15 app_qspi_receive_async

表 2-556 app_qspi_receive_async接口

函数原型	uint16_t app_qspi_receive_async(app_qspi_id_t id, uint8_t *p_data, uint32_t length)
功能说明	QSPI中断方式（异步）接收数据

版权所有 © 2025 深圳市汇顶科技股份有限公司

输入参数	• id：QSPI模块ID • p_data：接收数据的buffer • length：接收数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	该函数仅适用于GR551x

2.23.4.16 app_qspi_transmit_sync_ex

表 2-557 app_qspi_transmit_sync_ex接口

函数原型	uint16_t app_qspi_transmit_sync_ex(app_qspi_id_t id, uint32_t qspi_mode, uint32_t data_width, uint8_t *p_data, uint32_t length, uint32_t timeout)
功能说明	QSPI轮询方式（同步）发送数据，可设置时序模式和传输数据位宽
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • qspi_mode：数据传输采用的时序模式，可选QSPI_DATA_MODE_SPI（标准SPI模式）、QSPI_DATA_MODE_DUALSPI（Dual SPI模式）、QSPI_DATA_MODE_QUADSPI（Quad SPI模式） • data_width：数据位宽（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要传输数据的buffer • length：待发送数据的长度，以Byte为单位 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	GR551x使用该函数时，qspi_mode和data_width无意义

2.23.4.17 app_qspi_transmit_async_ex

表 2-558 app_qspi_transmit_async_ex接口

函数原型	uint16_t app_qspi_transmit_async_ex(app_qspi_id_t id, uint32_t qspi_mode, uint32_t data_width, uint8_t *p_data, uint32_t length)
功能说明	QSPI中断方式（异步）发送数据，可设置时序模式和传输数据位宽
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • qspi_mode：数据传输采用的时序模式，可选QSPI_DATA_MODE_SPI（标准SPI模式）、QSPI_DATA_MODE_DUALSPI（Dual SPI模式）、QSPI_DATA_MODE_QUADSPI（Quad SPI模式） • data_width：数据位宽（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要发送数据的buffer • length：待发送数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	当GR551x使用该函数时，qspi_mode和data_width无意义

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.23.4.18 app_qspi_receive_sync_ex

表 2-559 app_qspi_receive_sync_ex接口

函数原型	uint16_t app_qspi_receive_sync_ex(app_qspi_id_t id, uint32_t qspi_mode, uint32_t data_width, uint8_t *p_data, uint32_t length , uint32_t timeout, uint32_t timeout)
功能说明	QSPI轮询方式（同步）接收数据，可设置时序模式和传输数据位宽
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • qspi_mode：数据传输采用的时序模式，可选QSPI_DATA_MODE_SPI（标准SPI模式）、QSPI_DATA_MODE_DUALSPI（Dual SPI模式）、QSPI_DATA_MODE_QUADSPI（Quad SPI模式） • data_width：传输数据位宽（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要接收数据的buffer • length：待接收数据的长度，以Byte为单位 • timeout：超时时间，以ms为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	当GR551x使用该函数时，qspi_mode和data_width无意义

2.23.4.19 app_qspi_receive_async_ex

表 2-560 app_qspi_receive_async_ex接口

函数原型	uint16_t app_qspi_receive_async_ex(app_qspi_id_t id, uint32_t qspi_mode, uint32_t data_width, uint8_t *p_data, uint32_t length)
功能说明	QSPI中断方式（异步）接收数据，可设置时序模式和传输数据位宽
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • qspi_mode：数据传输采用的时序模式，可选QSPI_DATA_MODE_SPI（标准SPI模式）、QSPI_DATA_MODE_DUALSPI（Dual SPI模式）、QSPI_DATA_MODE_QUADSPI（Quad SPI模式） • data_width：传输数据位宽（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要接收数据的buffer • length：待接收数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	当GR551x使用该函数时，qspi_mode和data_width无意义

2.23.4.20 app_qspi_mmap_set_endian_mode

表 2-561 app_qspi_mmap_set_endian_mode接口

函数原型	bool app_qspi_mmap_set_endian_mode(app_qspi_id_t id, app_qspi_mmap_endian_mode_e mode)
功能说明	设置内存映射模式下的端序模式
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • mode：端序模式
返回值	• true：设置成功 • false：设置失败

版权所有 © 2025 深圳市汇顶科技股份有限公司

备注	

2.23.4.21 app_qspi_mmap_read_u8

表 2-562 app_qspi_mmap_read_u8接口

函数原型	uint8_t app_qspi_mmap_read_u8(app_qspi_id_t id, uint32_t address)
功能说明	内存映射模式下，读取U8数据，用于Flash/PSRAM设备
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • address：待访问设备的偏移地址（Flash/PSRAM）
返回值	读取的数据
备注	

2.23.4.22 app_qspi_mmap_read_u16

表 2-563 app_qspi_mmap_read_u16接口

函数原型	uint16_t app_qspi_mmap_read_u16(app_qspi_id_t id, uint32_t address)
功能说明	内存映射模式下，读取U16数据，用于Flash/PSRAM设备
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • address：待访问设备的偏移地址（Flash/PSRAM）
返回值	读取的数据
备注	

2.23.4.23 app_qspi_mmap_read_u32

表 2-564 app_qspi_mmap_read_u32接口

函数原型	uint32_t app_qspi_mmap_read_u32(app_qspi_id_t id, uint32_t address)
功能说明	内存映射模式下，读取U32数据，用于Flash/PSRAM设备
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • address：待访问设备的偏移地址（Flash/PSRAM）
返回值	读取的数据
备注	

2.23.4.24 app_qspi_mmap_read_block

表 2-565 app_qspi_mmap_read_block接口

函数原型	bool app_qspi_mmap_read_block(app_qspi_id_t id, uint32_t address, uint8_t * buffer, uint32_t length)
功能说明	内存映射模式下读取一个数据块
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • address：待访问设备的偏移地址（Flash/PSRAM） • buffer：读取数据存放的缓存指针

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• length：读取数据长度，以Byte为单位
返回值	• true：成功 • false：失败
备注	

2.23.4.25 app_qspi_get_xip_base_address

表 2-566 app_qspi_get_xip_base_address接口

函数原型	uint32_t app_qspi_get_xip_base_address(app_qspi_id_t id)
功能说明	获取QSPI内存映射在系统总线的基地址
输入参数	id：QSPI模块ID
返回值	QSPI内存映射在系统总线的基地址
备注	

2.23.4.26 app_qspi_transmit_in_qpi_async

表 2-567 app_qspi_transmit_in_qpi_async接口

函数原型	uint16_t app_qspi_transmit_in_qpi_async(app_qspi_id_t id, uint32_t data_width, uint8_t *p_data, uint32_t length)
功能说明	QSPI在QPI模式下中断方式（异步）发送数据
输入参数	• id：QSPI模块ID • data_width：数据位宽（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要发送数据的buffer • length：待发送数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR551x

2.23.4.27 app_qspi_get_handle

表 2-568 app_qspi_get_handle接口

函数原型	qspi_handle_t *app_qspi_get_handle(app_qspi_id_t id)
功能说明	获取QSPI句柄
输入参数	id：QSPI模块ID
返回值	QSPI句柄：参考SDK_Folder\drivers\inc\gr55xx_hal_qspi.h
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.23.5 DMA接口函数

表 2-569 DMA接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_qspi_dma_init	Y	Y	Y	N	N
app_qspi_dma_deinit	Y	Y	Y	N	N
app_qspi_dma_command_receive_async	Y	Y	Y	N	N
app_qspi_dma_command_transmit_async	Y	Y	Y	N	N
app_qspi_dma_command_async	Y	Y	Y	N	N
app_qspi_dma_transmit_async	Y	N	N	N	N
app_qspi_dma_receive_async	Y	N	N	N	N
app_qspi_dma_transmit_in_qpi_async	Y	N	N	N	N
app_qspi_dma_transmit_async_ex	Y	Y	Y	N	N
app_qspi_dma_receive_async_ex	Y	Y	Y	N	N
app_qspi_dma_mmap_read_block	N	Y	Y	N	N
app_qspi_async_draw_screen	N	Y	Y	N	N
app_qspi_async_veri_draw_screen	N	Y	Y	N	N
app_qspi_mmap_blit_image	N	Y	Y	N	N
app_qspi_async_llp_draw_block	N	Y	Y	N	N

2.23.5.1 app_qspi_dma_init

表 2-570 app_qspi_dma_init接口

函数原型	uint16_t app_qspi_dma_init(app_qspi_params_t *p_params)
功能说明	初始化QSPI DMA模式
输入参数	p_params：初始化参数的结构体指针
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.5.2 app_qspi_dma_deinit

表 2-571 app_qspi_dma_deinit接口

函数原型	uint16_t app_qspi_dma_deinit(app_qspi_id_t id)
功能说明	反初始化QSPI DMA模式
输入参数	id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述）
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义

版权所有 © 2025 深圳市汇顶科技股份有限公司

备注	

2.23.5.3 app_qspi_dma_command_receive_async

表 2-572 app_qspi_dma_command_receive_async接口

函数原型	uint16_t app_qspi_dma_command_receive_async(app_qspi_id_t id, app_qspi_command_t *p_cmd, uint8_t *p_data)
功能说明	QSPI以DMA方式（异步）读取数据，带控制命令封装
输入参数	• id：QSPI模块ID • p_cmd：封装的控制命令，参考2.23.2.5 app_qspi_command_t结构体描述 • p_data：读取数据存放的buffer
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.5.4 app_qspi_dma_command_transmit_async

表 2-573 app_qspi_dma_command_transmit_async接口

函数原型	uint16_t app_qspi_dma_command_transmit_async(app_qspi_id_t id, app_qspi_command_t *p_cmd, uint8_t *p_data)
功能说明	QSPI以DMA方式（异步）发送数据，带控制命令封装
输入参数	• id：QSPI模块ID • p_cmd：封装的控制命令，参考2.23.2.5 app_qspi_command_t结构体描述 • p_data：待发送数据存放的buffer
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.5.5 app_qspi_dma_command_async

表 2-574 app_qspi_dma_command_async接口

函数原型	uint16_t app_qspi_dma_command_async(app_qspi_id_t id, app_qspi_command_t *p_cmd)
功能说明	QSPI以DMA方式（异步）发送命令
输入参数	• id：QSPI模块ID • p_cmd：待发送命令的buffer
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.23.5.6 app_qspi_dma_transmit_async

表 2-575 app_qspi_dma_transmit_async接口

函数原型	uint16_t app_qspi_dma_transmit_async(app_qspi_id_t id, uint8_t *p_data, uint32_t length)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	QSPI以DMA方式（异步）发送数据
输入参数	• id：QSPI模块ID • p_data：需要发送数据的buffer • length：待发送数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR551x

2.23.5.7 app_qspi_dma_receive_async

表 2-576 app_qspi_dma_receive_async接口

函数原型	uint16_t app_qspi_dma_receive_async(app_qspi_id_t id, uint8_t *p_data, uint32_t length)
功能说明	QSPI以DMA方式（异步）接收数据
输入参数	• id：QSPI模块ID • p_data：接收数据的buffer • length：接收数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR551x

2.23.5.8 app_qspi_dma_transmit_in_qpi_async

表 2-577 app_qspi_dma_transmit_in_qpi_async接口

函数原型	uint16_t app_qspi_dma_transmit_in_qpi_async(app_qspi_id_t id, uint32_t data_width, uint8_t *p_data, uint32_t length)
功能说明	QSPI在QPI模式下以DMA方式（异步）发送数据
输入参数	• id：QSPI模块ID • data_width：数据宽度（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要发送数据的buffer • length：待发送数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	仅适用于GR551x

2.23.5.9 app_qspi_dma_receive_async_ex

表 2-578 app_qspi_dma_receive_async_ex接口

函数原型	uint16_t app_qspi_dma_receive_async_ex(app_qspi_id_t id, uint32_t qspi_mode, uint32_t data_width, uint8_t *p_data, uint32_t length)
功能说明	QSPI以DMA方式（异步）接收数据，可设置时序模式和数据位宽
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • qspi_mode：数据传输采用的时序模式，可选QSPI_DATA_MODE_SPI（标准SPI模式）、QSPI_DATA_MODE_DUALSPI（Dual SPI模式）、QSPI_DATA_MODE_QUADSPI（Quad SPI模式）

版权所有 © 2025 深圳市汇顶科技股份有限公司

	• data_width：传输数据位宽（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要接收数据的buffer • length：待接收数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	当GR551x使用该函数时，qspi_mode和data_width无意义

2.23.5.10 app_qspi_dma_transmit_async_ex

表 2-579 app_qspi_dma_transmit_async_ex接口

函数原型	uint16_t app_qspi_dma_transmit_async_ex(app_qspi_id_t id, uint32_t qspi_mode, uint32_t data_width, uint8_t *p_data, uint32_t length)
功能说明	QSPI以DMA方式（异步）发送数据，可设置时序模式和数据位宽
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • qspi_mode：数据传输采用的时序模式，可选QSPI_DATA_MODE_SPI（标准SPI模式）、QSPI_DATA_MODE_DUALSPI（Dual SPI模式）、QSPI_DATA_MODE_QUADSPI（Quad SPI模式） • data_width：传输数据位宽（支持QSPI_DATASIZE_08_BITS、QSPI_DATASIZE_16_BITS、QSPI_DATASIZE_32_BITS） • p_data：需要发送数据的buffer • length：待发送数据的长度，以Byte为单位
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	当GR551x使用该函数时，qspi_mode和data_width无意义

2.23.5.11 app_qspi_dma_mmap_read_block

表 2-580 app_qspi_dma_mmap_read_block接口

函数原型	bool app_qspi_dma_mmap_read_block(app_qspi_id_t id, uint32_t address, uint8_t * buffer, uint32_t length)
功能说明	内存映射模式下DMA方式（异步）读取一个数据块，数据按Flash/PSRAM设备的顺序排序
输入参数	• id：QSPI模块ID（详见2.23.2.1 app_qspi_params_t中描述） • address：待访问设备的偏移地址（Flash/PSRAM） • length：读取数据长度，以Byte为单位
返回值	• true：成功 • false：失败
备注	驱屏接口单独抽离在SDK_Folder\drivers\inc\app_graphics_qspi.h中

2.23.5.12 app_qspi_async_draw_screen

表 2-581 app_qspi_async_draw_screen接口

函数原型	bool app_qspi_async_draw_screen(app_qspi_id_t screen_id, app_qspi_id_t storage_id, const app_qspi_screen_command_t * const p_screen_cmd, const app_qspi_screen_info_t * const p_screen_info, app_qspi_screen_scroll_t * p_scroll_config, bool is_first_call)

版权所有 © 2025 深圳市汇顶科技股份有限公司

功能说明	异步刷屏专用接口
输入参数	• screen_id：屏幕ID • storage_id：指定素材所在的QSPI存储器ID • p_screen_cmd：指向刷屏控制命令的指针 • p_screen_info：屏幕信息 • p_scroll_config：屏幕滚动控制信息 • is_first_call：用户前台调用请设置为true
返回值	• true：成功 • false：失败
备注	• 如果图片素材不是在QSPI设备上，storage_id需设定为APP_STORAGE_RAM_ID • storage_id和screen_id不能相同 • 入参p_screen_info的scrn_pixel_depth只能设为2，即该接口只能处理16 bits的显示输出 • 驱屏接口单独抽离在SDK_Folder\drivers\inc\app_graphics_qspi.h中

2.23.5.13 app_qspi_async_veri_draw_screen

表 2-582 app_qspi_async_veri_draw_screen接口

函数原型	bool app_qspi_async_veri_draw_screen(app_qspi_id_t screen_id, app_qspi_id_t storage_id, const app_qspi_screen_command_t * const p_screen_cmd, const app_qspi_screen_info_t * const p_screen_info, app_qspi_screen_veri_link_scroll_t * p_link_scroll, bool is_first_call)
功能说明	异步链式刷屏专用接口（垂直链）
输入参数	• screen_id：屏幕ID • storage_id：指定素材所在的QSPI存储器ID • p_screen_cmd：指向刷屏控制命令的指针 • p_screen_info：屏幕信息 • p_link_scroll：链式滚动参数配置 • is_first_call：用户前台调用请设置为true
返回值	• true：成功 • false：失败
备注	• 如果图片素材不是在QSPI设备上，storage_id需设定为APP_STORAGE_RAM_ID • storage_id和screen_id不能相同 • 入参p_screen_info的scrn_pixel_depth只能设为2，即该接口只能处理16 bits的显示输出 • 驱屏接口单独抽离在SDK_Folder\drivers\inc\app_graphics_qspi.h中

2.23.5.14 app_qspi_mmap_blit_image

表 2-583 app_qspi_mmap_blit_image接口

函数原型	bool app_qspi_mmap_blit_image(app_qspi_id_t storage_id, blit_image_config_t * p_blit_config, blit_xfer_type_e xfer_type)
功能说明	使用DMA进行二维数据搬运接口
输入参数	• storage_id：指定素材所在的QSPI存储器ID • p_blit_config：Blit传输配置 • xfer_type：传输类型，支持SG和LLP

版权所有 © 2025 深圳市汇顶科技股份有限公司

返回值	• true：成功 • false：失败
备注	• 驱屏接口单独抽离在SDK_Folder\drivers\inc\app_graphics_qspi.h中 • 使用该接口，需要把*app_qspi_user_config.h*中的宏定义QSPI_BLIT_RECT_IMAGE_SUPPORT设为大于0的值（默认为0）

2.23.5.15 app_qspi_async_llp_draw_block

表 2-584 app_qspi_async_llp_draw_block接口

函数原型	bool app_qspi_async_llp_draw_block(app_qspi_id_t screen_id, app_qspi_id_t storage_id, const app_qspi_screen_command_t *const p_screen_cmd, const app_qspi_screen_info_t *const p_screen_info, app_qspi_screen_block_t *p_block_info, bool is_first_call)
功能说明	使用DMA-LLP（DMA链式）方式刷新屏幕的块区域，链表的节点为待刷新图像数据块的每行数据
输入参数	• screen_id：屏幕ID • storage_id：指定素材所在的QSPI存储器ID • p_screen_cmd：指向刷屏控制命令的指针 • p_screen_info：屏幕信息 • p_block_info：刷屏的图像区域块 • is_first_call：是否第一次调用，第一次调用设置为true（设置为true表明一帧图像的开始，需要发送命令字节）
返回值	• true：成功 • false：失败
备注	• 当传输结束，会产生APP_QSPI_EVT_TX_CPLT事件 • 驱屏接口单独抽离在SDK_Folder\drivers\inc\app_graphics_qspi.h中

2.23.6 示例工程

2.23.6.1 QSPI

示例工程实现了对外接Flash的擦除和读写，工程路径：SDK_Folder\projects\peripheral\qspi\app_qspi。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.23.6.1.1 流程图

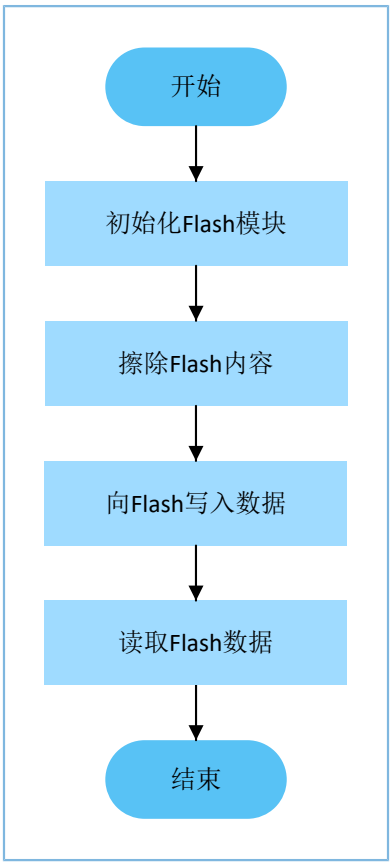

图 2-30 QSPI实现Flash的擦除和读写流程图

2.23.6.1.2 初始化参数

```
static app_qspi_params_t g_qspi_params = {
    .id = QSPI_ID,
    .pin_cfg = {
        .cs = {
            .type   = APP_FLASH_CS_IO_TYPE,
            .mux    = APP_FLASH_GPIO_MUX,
            .pin    = APP_FLASH_CS_PIN,
            .mode   = APP_IO_MODE_MUX,
            .pull   = APP_IO_PULLUP,
            .enable = APP_QSPI_PIN_ENABLE,
        },
        .clk = {
            .type   = APP_FLASH_CLK_IO_TYPE,
            .mux    = APP_FLASH_GPIO_MUX,
            .pin    = APP_FLASH_CLK_PIN,
            .mode   = APP_IO_MODE_MUX,
            .pull   = APP_IO_PULLUP,
            .enable = APP_QSPI_PIN_ENABLE,
        },
        .io_0 = {
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
            .type   = APP_FLASH_IO0_IO_TYPE,
            .mux    = APP_FLASH_GPIO_MUX,
            .pin    = APP_FLASH_IO0_PIN,
            .mode   = APP_IO_MODE_MUX,
            .pull   = APP_IO_PULLUP,
            .enable = APP_QSPI_PIN_ENABLE,
        },
        .io_1 = {
            .type   = APP_FLASH_IO1_IO_TYPE,
            .mux    = APP_FLASH_GPIO_MUX,
            .pin    = APP_FLASH_IO1_PIN,
            .mode   = APP_IO_MODE_MUX,
            .pull   = APP_IO_PULLUP,
            .enable = APP_QSPI_PIN_ENABLE,
        },
        .io_2 = {
            .type   = APP_FLASH_IO2_IO_TYPE,
            .mux    = APP_FLASH_GPIO_MUX,
            .pin    = APP_FLASH_IO2_PIN,
            .mode   = APP_IO_MODE_MUX,
            .pull   = APP_IO_PULLUP,
            .enable = APP_QSPI_PIN_ENABLE,
        },
        .io_3 = {
            .type   = APP_FLASH_IO3_IO_TYPE,
            .mux    = APP_FLASH_GPIO_MUX,
            .pin    = APP_FLASH_IO3_PIN,
            .mode   = APP_IO_MODE_MUX,
            .pull   = APP_IO_PULLUP,
            .enable = APP_QSPI_PIN_ENABLE,
        },
    },
    .dma_cfg = {
        .dma_instance = QSPI_USED_DMA,
        .dma_channel  = DMA_Channel0,
    },
    .init = {
        .clock_mode      = QSPI_TIMING_MODE,
        .clock_prescaler = QSPI_CLOCK_PRESCALER,
        .rx_sample_delay = QSPI_RX_SAMPLE_DELAY,
    },
};
```

- id：QSPI使用的ID选择QSPI_ID，选择的具体ID请参见*board_SK.h*。
- pin_cfg：QSPI引脚配置，包含时钟引脚clk、片选引脚cs、数据引脚io0 ～ io3，配置引脚IO类型，PIN脚，复用模式和上下拉模式，是否使能，具体引脚参数定义请参见*board_SK.h*。

版权所有 © 2025 深圳市汇顶科技股份有限公司

- dma_cfg：DMA参数配置

  dma_cfg.dma_instance：选择使用的DMA实例

  dma_cfg.dma_channel：选择使用的DMA实例通道0

- init：QSPI传输参数配置

  init.clock_mode：时钟模式选择QSPI_CLOCK_MODE_0，时钟模式0

  init.clock_prescaler：分频系数，外设时钟等于外设总线时钟除以分频系数

  init.rx_sample_delay：时钟延时，以QSPI SCLK时钟周期为单位

2.23.6.1.3 重要函数

- Flash初始化

```
SPI_FLASH_init()
```

- 读取设备ID

```
device_id = SPI_FLASH_Read_Device_ID()
```

- 不使能Quad，Quad模式请参考Flash手册

```
SPI_FLASH_Disable_Quad()
```

- 读取Flash数据

```
SPI_FLASH_Read(FLASH_PROGRAM_START_ADDR, read_buffer, FLASH_OPERATION_LENGTH)
```

- 擦除Flash数据

```
SPI_FLASH_Sector_Erase(FLASH_PROGRAM_START_ADDR)
```

- 写入数据到Flash

```
SPI_FLASH_Page_Program(FLASH_PROGRAM_START_ADDR, write_buffer)
```

- 使能Quad，Quad模式请参考Flash手册

```
SPI_FLASH_Enable_Quad()
```

- Quad模式读取Flash数据

```
SPI_FLASH_Dual_Output_Fast_Read(FLASH_PROGRAM_START_ADDR, read_buffer,
                                FLASH_OPERATION_LENGTH)
```

2.23.6.1.4 测试验证

Flash已集成到开发板，连接开发板到PC端，打开串口调试助手查看打印日志，读取Flash的数据和写入一致，则测试成功。

```
This is SPI_flash example.
Read_Device_ID = 0x0B6017
Erase Sector...
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
Erase Sector Success.
Page_Program...
Page Program Success.
SPI_Read.
0x00040000: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07
0x00040008: 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x00040010: 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17
0x00040018: 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x00040020: 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27
0x00040028: 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x00040030: 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x00040038: 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x00040040: 0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47
0x00040048: 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x00040050: 0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57
0x00040058: 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x00040060: 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67
0x00040068: 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x00040070: 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77
0x00040078: 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x00040080: 0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87
0x00040088: 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x00040090: 0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97
0x00040098: 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0x000400a0: 0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7
0x000400a8: 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0x000400b0: 0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7
0x000400b8: 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0x000400c0: 0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7
0x000400c8: 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0x000400d0: 0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7
0x000400d8: 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0x000400e0: 0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7
0x000400e8: 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0x000400f0: 0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7
0x000400f8: 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
QSPI DMA read data...
Dual_Output_Fast_Read.
0x00040000: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07
0x00040008: 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x00040010: 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17
0x00040018: 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x00040020: 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27
0x00040028: 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x00040030: 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x00040038: 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x00040040: 0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47
0x00040048: 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x00040050: 0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

```
0x00040058: 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x00040060: 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67
0x00040068: 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x00040070: 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77
0x00040078: 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x00040080: 0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87
0x00040088: 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x00040090: 0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97
0x00040098: 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0x000400a0: 0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7
0x000400a8: 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0x000400b0: 0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7
0x000400b8: 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0x000400c0: 0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7
0x000400c8: 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0x000400d0: 0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7
0x000400d8: 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0x000400e0: 0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7
0x000400e8: 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0x000400f0: 0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7
0x000400f8: 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
This example demo end.
```

2.24 APP BOD驱动

2.24.1 宏定义

2.24.1.1 APP_BOD_ENABLE

表 2-585 APP_BOD_ENABLE宏定义

描述	值
使能BOD模块	HAL_BOD_ENABLE

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.1.2 APP_BOD_DISABLE

表 2-586 APP_BOD_DISABLE宏定义

描述	值
不使能BOD模块	HAL_BOD_DISABLE

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.24.1.3 APP_BOD_EVENT_ENABLE

表 2-587 APP_BOD_EVENT_ENABLE宏定义

描述	值
使能BOD2事件	HAL_BOD2_ENABLE

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.1.4 APP_BOD_EVENT_DISABLE

表 2-588 APP_BOD_EVENT_DISABLE宏定义

描述	值
不使能BOD2事件	HAL_BOD2_DISABLE

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.1.5 APP_BOD_EVENT_LEVEL_XX

表 2-589 APP_BOD_EVENT_LEVEL_XX宏定义

描述	值
BOD2事件等级0	HAL_BOD2_LEVEL_0
BOD2事件等级1	HAL_BOD2_LEVEL_1
BOD2事件等级2	HAL_BOD2_LEVEL_2
BOD2事件等级3	HAL_BOD2_LEVEL_3
BOD2事件等级4	HAL_BOD2_LEVEL_4
BOD2事件等级5	HAL_BOD2_LEVEL_5
BOD2事件等级6	HAL_BOD2_LEVEL_6
BOD2事件等级7	HAL_BOD2_LEVEL_7
BOD2事件等级8	HAL_BOD2_LEVEL_8（仅GR5526支持）
BOD2事件等级9	HAL_BOD2_LEVEL_9（仅GR5526支持）
BOD2事件等级10	HAL_BOD2_LEVEL_10（仅GR5526支持）
BOD2事件等级11	HAL_BOD2_LEVEL_11（仅GR5526支持）
BOD2事件等级12	HAL_BOD2_LEVEL_12（仅GR5526支持）
BOD2事件等级13	HAL_BOD2_LEVEL_13（仅GR5526支持）
BOD2事件等级14	HAL_BOD2_LEVEL_14（仅GR5526支持）
BOD2事件等级15	HAL_BOD2_LEVEL_15（仅GR5526支持）

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.24.2 结构体

2.24.2.1 app_bod_params_t

结构体定义如下：

表 2-590 app_bod_params_t结构体成员

成员	描述	取值
bod_init_t init	BOD初始化参数结构体	参考2.24.2.2 bod_init_t结构体
bod_env_t bod_env	用于记录初始化设备的句柄，无需配置	

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.2.2 bod_init_t

结构体定义如下：

表 2-591 bod_init_t结构体成员

成员	描述	取值
uint8_t bod_en	使能BOD模块（硬件），当供电电压低于典型值1.45 V时，复位MCU	该参数的取值可以是下列值中的任意一个： • APP_BOD_ENABLE（使能） • APP_BOD_DISABLE（不使能）
uint8_t bod2_en	使能BOD2事件（软件），当供电电压低于bod2_lvl配置的值时，触发中断	该参数的取值可以是下列值中的任意一个： • APP_BOD_EVENT_ENABLE（使能） • APP_BOD_EVENT_DISABLE（不使能）
uint8_t bod2_lvl	BOD2事件触发阈值	该参数的取值可以是下列值中的任意一个： • APP_BOD_EVENT_LEVEL_0 • APP_BOD_EVENT_LEVEL_1 • APP_BOD_EVENT_LEVEL_2 • APP_BOD_EVENT_LEVEL_3 • APP_BOD_EVENT_LEVEL_4 • APP_BOD_EVENT_LEVEL_5 • APP_BOD_EVENT_LEVEL_6 • APP_BOD_EVENT_LEVEL_7 • APP_BOD_EVENT_LEVEL_8 (仅GR5526支持) • APP_BOD_EVENT_LEVEL_9 (仅GR5526支持) • APP_BOD_EVENT_LEVEL_10 (仅GR5526支持)

版权所有 © 2025 深圳市汇顶科技股份有限公司

成员	描述	取值
		• APP_BOD_EVENT_LEVEL_11 (仅GR5526支持) • APP_BOD_EVENT_LEVEL_12 (仅GR5526支持) • APP_BOD_EVENT_LEVEL_13 (仅GR5526支持) • APP_BOD_EVENT_LEVEL_14 (仅GR5526支持) • APP_BOD_EVENT_LEVEL_15 (仅GR5526支持) 每个档位对应的电压值请参考芯片Datasheet中的BOD章节。2.4 V以下将低于芯片正常供电电压，不建议使用

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.2.3 app_bod_evt_t

结构体定义如下：

表 2-592 app_bod_evt_t结构体成员

成员	描述	取值
app_bod_evt_type_t type	BOD事件结构体	该参数的取值可以是下列值中的任意一个： • APP_BOD_EVT_TRIGGERED：供电电压低于阈值时，触发中断 • APP_BOD_EVT_REMOVED：供电电压高于阈值时，触发“BOD解除”中断（GR551x不支持）

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.3 枚举

2.24.3.1 app_bod_evt_type_t

枚举定义如下：

表 2-593 app_bod_evt_type_t枚举成员

成员	描述
APP_BOD_EVT_TRIGGERED	BOD中断触发事件：供电电压低于阈值时，触发中断
APP_BOD_EVT_REMOVED	BOD条件解除中断事件：供电电压高于阈值时，触发“BOD解除”中断（GR551x不支持）

版权所有 © 2025 深圳市汇顶科技股份有限公司

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.3.2 app_bod_state_t

枚举定义如下：

表 2-594 app_bod_state_t枚举成员

成员	描述
APP_BOD_INVALID = 0	BOD不可用
APP_BOD_ACTIVITY = 1	BOD激活

详细信息请参考文件：SDK_Folder\drivers\inc\app_bod.h。

2.24.4 接口函数

表 2-595 BOD驱动接口函数

API	芯片				
	GR551x	GR5526	GR5x25	GR533x	GR5405
app_bod_init	Y	Y	Y	Y	Y
app_bod_deinit	Y	Y	Y	Y	Y
app_bod_enable	Y	Y	Y	Y	Y
app_bod_event_enable	Y	Y	Y	Y	Y
app_bod_event_set_level	Y	Y	Y	Y	Y
app_bod_static_mode_enable	N	N	N	Y	Y
app_bod_event_auto_power_bypass_enable	N	N	N	Y	Y

2.24.4.1 app_bod_init

表 2-596 app_bod_init接口

函数原型	uint16_t app_bod_init(app_bod_params_t *p_params, app_bod_evt_handler_t evt_handler)
功能说明	初始化BOD
输入参数	• p_params：初始化参数的结构体指针 • evt_handler：用户自己实现的中断回调函数
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	回调函数形式参照typedef void (*app_bod_evt_handler_t)(app_bod_evt_t *p_evt)； HAL层已经实现中断标志位的清理，用户无需关心中断标志的清理，回调函数根据返回事件类型进行相应的处理即可。

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.24.4.2 app_bod_deinit

表 2-597 app_bod_deinit接口

函数原型	uint16_t app_bod_deinit(void)
功能说明	反初始化BOD
输入参数	
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.24.4.3 app_bod_enable

表 2-598 app_bod_enable接口

函数原型	uint16_t app_bod_enable(uint8_t enable)
功能说明	使能BOD硬件
输入参数	enable：1-使能；0-禁用
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.24.4.4 app_bod_event_enable

表 2-599 app_bod_event_enable接口

函数原型	uint16_t app_bod_event_enable(uint8_t enable)
功能说明	使能BOD事件
输入参数	enable：1-使能；0-禁用
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.24.4.5 app_bod_event_set_level

表 2-600 app_bod_event_set_level接口

函数原型	uint16_t app_bod_event_set_level(uint8_t level)
功能说明	设置BOD事件优先级
输入参数	level：优先级
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.24.4.6 app_bod_static_mode_enable

表 2-601 app_bod_static_mode_enable接口

函数原型	uint16_t app_bod_static_mode_enable(uint8_t enable)
功能说明	设置是否使能BOD静态模式
输入参数	enable：1-使能；0-禁用
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.24.4.7 app_bod_event_auto_power_bypass_enable

表 2-602 app_bod_event_auto_power_bypass_enable接口

函数原型	uint16_t app_bod_event_auto_power_bypass_enable (uint8_t enable)
功能说明	设置是否使能BOD旁路自动上/下电功能
输入参数	enable：1-使能；0-禁用
返回值	APP_DRV_xxx：详见SDK_Folder\drivers\inc\app_drv_error.h宏定义
备注	

2.24.5 示例工程

2.24.5.1 BOD供电电压监测

本示例实现对供电电压的监测，工程路径：SDK_Folder\projects\peripheral\bod\app_bod。

2.24.5.1.1 初始化参数

```
s_params.init.bod_en = APP_BOD_ENABLE;
s_params.init.bod2_en = APP_BOD_EVENT_ENABLE;
s_params.init.bod2_lvl = APP_BOD_EVENT_LEVEL_10;
```

- init.bod_en：APP_BOD_ENABLE使能BOD模块（硬件），当供电电压低于典型值1.45 V时，复位MCU。
- init.bod2_en：APP_BOD_EVENT_ENABLE使能BOD2事件（软件），当供电电压低于bod2_lvl配置的值时，触发中断。
- init.bod2_lvl：BOD2事件触发阈值，APP_BOD_EVENT_LEVEL_10表示2.5 V。

2.24.5.1.2 重要函数

初始化BOD

```
app_bod_init(&s_params, app_bod_evt_handler)
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

2.24.5.1.3 测试验证

1. 连接开发板到PC端，下载程序，打开串口调试助手。
2. VBAT去掉内部电路供电，采用外部输入供电3.2 V，按RESET键复位。
3. 改变供电电压为2 V，观察串口打印，是否有BOD TRIGGERED EVENT中断产生。
4. 恢复供电电压为3.2 V，观察串口打印，是否有BOD REMOVED EVENT中断产生。

```
BOD TRIGGERED EVENT
BOD REMOVED EVENT
```

版权所有 © 2025 深圳市汇顶科技股份有限公司

3 术语和缩略语

表 3-1 术语和缩略语

名称	描述
ADC	Analog-to-digital Converter，模数转换器
AES	Advanced Encryption Standard，高级加密标准
AON GPIO	Always-on GPIO，不间断电源的GPIO
AON WDT	Always-on WDT，不间断电源的WDT
API	Application Programming Interface，应用程序接口
APP	Application Layer，应用层
BOD	Brown-out Detector，欠压检测器
DMA	Direct Memory Access，直接内存访问
GPIO	General Purpose I/Os，通用I/O口
HAL	Hardware Abstraction Layer，硬件抽象层
HMAC	Hash-based Message Authentication Code，基于Hash算法的消息认证码
I2C	Inter-integrated Circuit，内置集成电路
I2S	Inter-IC Sound，集成电路内置音频总线
LL	Low Layer，底层
MSIO	Mixed Signal I/O，混合信号I/O口
NVIC	Nested Vectored Interrupt Controller，嵌套向量中断控制器
OSPI	Octal Serial Peripheral Interface，8线串行外设接口
PDM	Pulse Density Modulation，脉冲强度调制
PKC	Public Key Cipher，公钥密码体系
PWM	Pulse Width Modulation，脉冲宽度调制
PWR	Power Controller，电源控制器
QSPI	Quad Serial Peripheral Interface，4线串行外设接口
RNG	Random Number Generator，随机数生成器
SPI	Serial Peripheral Interface，串行外设接口
SysTick	System Tick Timer，系统节拍定时器
TIMER	Advanced-control, general-purpose or basic Timer，定时器
UART	Universal Asynchronous Receiver/Transmitter，通用非同步收发传输器
WDT	Watchdog Timer，看门狗定时器

版权所有 © 2025 深圳市汇顶科技股份有限公司