Witmem
知存科技

WTM2101

Hal i2s 使用说明

版本号：V1.0.0.0
日期：2023.01.31

声明

商标声明：

作为北京知存科技有限公司的商标，本文件中提到的所有其他商标和商号均为其持有人的财产。

版权声明：

内容声明：

本文件中的信息如有更改，恕不另行通知。为了确保内容的准确性，文章会做出相关的确认，但本文件中的所有声明、信息和建议不构成任何明示或暗示的保证。

北京知存科技有限公司

地址：北京市海淀区北四环西路 56 号辉煌时代大厦西座 1502

网址：http://www.witintech.com

目录

一、 文档功能说明

该文档旨在说明 hal_i2s 的驱动库.包括接口功能以及结构体参数意义.

二、 函数功能介绍

hal_i2s_instance_get()

Hal_I2s_InitTypeDef* hal_i2s_instance_get(Hal_I2s_Instance_Typedef number)
获取 hal i2s 实例对象
参数
number： hal i2s 实例对象编号
返回值
成功返回 Hal i2s 实例对象,否则是 NULL

hal_i2s_init()

int hal_i2s_init(Hal_I2s_InitTypeDef *i2s_instance,
I2S_TypeDef *address,
Hal_I2s_Type_Typedef type,
Hal_I2s_Work_Mode_Typedef mode,
Hal_I2s_Width_Word_Typedef width,
int lr_channel_need_sizes_by_width)
初始化 hal i2s 实例对象
参数
i2s_instance: hal i2s 实例对象
address: i2s 硬件地址
type: i2s 工作类型
mode: i2s 工作模式
width: i2s 左右通道的位宽
lr_channel_need_sizes_by_width: 应用层处理 i2s 左右通道数据的大小,以数值 width 位宽单位.
返回值
大于 0 成功,否则失败
注意:
1. 默认左右通道速率是 16000
2. i2s 硬件地址 I2S0 时,gpio 可以使用 gpio0, gpio1, gpio2, gpio3.
i2s 硬件地址 I2S1 时,gpio 可以使用 gpio10, gpio11, gpio12, gpio13

3. 默认发送及接收 buffer 缓存大小是 lr_channel_need_sizes_by_width 的 3 倍
4. 默认使用 dma,使用的 dma 的通道是 2

hal_i2s_open()

int hal_i2s_open(Hal_I2s_InitTypeDef *i2s_instance)
根据传入的 hal i2s 实例对象打开相应的硬件
参数
i2s_instance:hal i2s 实例对象
返回值
大于 0 成功,否则失败

hal_i2s_ctl()

int hal_i2s_ctl(Hal_I2s_InitTypeDef *i2s_instance,int command, ...)
控制 hal i2s 实例对象或硬件参数
参数
command:命令参数
....:命令参数的附加参数
返回值
大于 0 成功,否则失败
注意:
命令参数: HAL_I2S_CHANNEL_ENABLE_COMMAND 表示开启或者关闭 i2s

hal_i2s_write()

```
int hal_i2s_write(Hal_I2s_InitTypeDef *i2s_instance,
                  void *left_data,
                  void *right_data,
                  int size_by_data);
```

写数据到 hal i2s 实例对象的发送 buffer
参数
i2s_instance:hal i2s 实例对象
left_data:左通道数据

right_data:右通道数据
size_by_data:写单通道数据大小，以数值 width 位宽单位

返回值

大于 0 成功,否则失败

hal_i2s_read()

int hal_i2s_read(Hal_I2s_InitTypeDef *i2s_instance,
void *left_data,
void *right_data,
int size_by_data)

读数据从 hal i2s 实例对象的读 buffer

参数

i2s_instance:hal i2s 实例对象
left_data:左通道数据
right_data:右通道数据
size_by_data:读单通道数据大小，以数值 width 位宽单位

返回值

大于 0 成功,否则失败

hal_i2s_close()

int hal_i2s_close(Hal_I2s_InitTypeDef *i2s_instance)

关闭 hal i2s 实例对象及关联的硬件

参数

i2s_instance:hal i2s 实例对象

返回值

大于 0 成功,否则失败

三、 结构体介绍

Hal_I2s_InitTypeDef

Hal i2s 配置结构体

```
typedef struct Hal_I2s_InitTypeDef{
    FunctionalState enable;
    I2S_TypeDef *instance;
    Hal_I2s_Gpio_Typedef io;
    Hal_I2s_Type_Typedef type;
    Hal_I2s_Work_Mode_Typedef mode;
    Hal_I2s_Width_Word_Typedef width_word;
    int lrclk_frequency;
    int lr_channel_need_sizes_by_width;
    Hal_I2s_Cache_Typedef send_buffer;
    Hal_I2s_Cache_Typedef receive_buffer;
    Data_handle Data_handle_info;
    Hal_I2s_Dma_Typedef dma;
}Hal_I2s_InitTypeDef;
```

参数

enable:结构体初始化标志

Instance:i2s 硬件地址

Io:gpio 配置结构体

type:i2s 工作类型

mode:i2s 工作模式

width_word:i2s 左右通道位宽

lrclk_frequency:i2s 左右通道时钟速率

lr_channel_need_sizes_by_width: 应用层处理 i2s 左右通道数据大小,以位宽单位.

send_buffer:hal i2s 发送 buffer 结构体

receive_buffer:hal i2s 接收 buffer 结构体

Data_handle_info:hal i2s 内部数据处理接口

dma:dma 配置结构体

Hal_I2s_Gpio_Typedef

Gpio 配置结构体

```
typedef struct{
   uint32_t sdo_io,sdi_io,bclk_io,lrclk_io;
   uint32_t sdo_io_af,sdi_io_af,bclk_io_af,lrclk_io_af;
}Hal_I2s_Gpio_Typedef;
```

参数

sdo_io,sdi_io,bclk_io,lrclk_io:i2s 对应的 gpio

sdo_io_af,sdi_io_af,bclk_io_af,lrclk_io_af:i2s 对应 gpio 的功能选择

Hal_I2s_Cache_Typedef

Hal i2s 缓存 buffer 结构体

```
typedef struct{
   int lr_channel_need_sizes_by_width_counts;
   uint8_t *buffer;
   int read_index;
   int write_index;
}Hal_I2s_Cache_Typedef;
```

参数

lr_channel_need_sizes_by_width_counts:buffer 缓存大小,基于 hal i2s 配置结构体中变量 lr_channel_need_sizes_by_width 的倍数

buffer:数据 buffer

read_index:数据 buffer 的读下标

write_index:数据 buffer 的写下标

Hal_I2s_Dma_Typedef

Hal i2s 的 dma 配置结构体

```
typedef struct{
   FunctionalState enable;
   uint32_t dma_channel;
   DMA_InitTypeDef config;
```

```
    DMA_LlpTypeDef llp_cfg[2];
    uint8_t cache_buffer[16 * 4 * 2];
    uint32_t dma_cnt;
}Hal_I2s_Dma_Typedef;
```

参数

enable:dma 启动标志

dma_channel:dma 使用的通道

config:dma 参数配置结构体

llp_cfg:dma 链表模式对应的结构体

cache_buffer:dma 处理发送和接收数据时使用的 buffer.此 buffer 内部使用

dma_cnt:dma 处理发送和接收数据时的计数器.此计数器内部使用

四、 其他

1.i2s 数据存储 buffer 申请自 heap.c 文件中的数组 static uint8_t ucHeap[configTOTAL_HEAP_SIZE] __attribute__ ((section(".audmem")));
在使用前需要自己在链接脚本中定义一块安全可用的地址,地址范围根据属性 __attribute__ ((section(".audmem")))来指定.数组大小由宏 configTOTAL_HEAP_SIZE 定义

2.在 hal_i2s.c 中的中断子程序只是示例代码,实际使用时务必重写中断子程序,否则会重现严重的错误.

五、 修订历史

表 5-1 修订历史

版本	日期	修订人	说明
V1.0.0.0	2023-01-31	李剑	初次编写