RTOS TWI

开发指南

版本号: 1.1

发布日期: 2020.4.27

版本历史

版本号	日期	制/修订人	内容描述
1.0	2020.7.16	AWA1637	1. 初版
1.1	2020.4.27	AWA1637	1. 添加 F133 的说明

目　录

1 前言 **1**
1.1 文档简介 1
1.2 目标读者 1
1.3 适用范围 1

2 模块介绍 **2**
2.1 模块功能介绍 2
2.2 相关术语介绍 2
2.3 模块配置介绍 3
2.4 模块配置介绍 3
2.5 模块源码结构 4
2.6 模块结构说明 4
2.6.1 TWI 端口号 twi_port_t 4
2.6.2 TWI 消息结构体 twi_msg_t 4
2.6.3 TWI 控制结构体 hal_twi_transfer_cmd_t 5

3 模块接口说明 **6**
3.1 接口列表 6
3.2 接口使用说明 6
3.2.1 TWI 初始化接口 6
3.2.2 TWI 控制接口 6
3.2.3 TWI 数据发送接口 7
3.2.4 TWI 数据接收接口 7
3.2.5 TWI 去初始化接口 7

4 模块使用范例 **8**

5 FAQ **9**

1 前言

1.1 文档简介

介绍 RTOS 中 TWI 驱动的接口及使用方法，为 TWI 使用者提供参考。

1.2 目标读者

TWI 驱动层/应用层开发/使用/维护人员。

1.3 适用范围

表 1-1：适用产品列表

产品名称	内核版本	驱动文件
V459	Melis	hal_twi.c
F133	Melis	hal_twi.c
R328	FreeRTOS	hal_twi.c

2 模块介绍

2.1 模块功能介绍

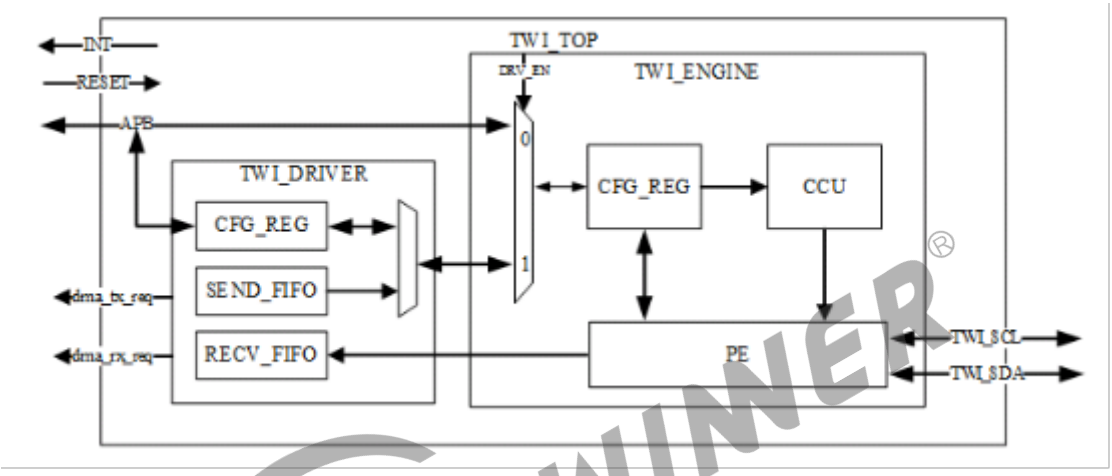

图 2-1: TWI 硬件方框图

TWI 控制器的框图如上所示，该控制器支持的标准通信速率为 100Kbps，最高通信速率可以达到 400Kbps。其中 CPUX 域的 TWI 控制器时钟源来自于 APB2，CPUS 域的 R-TWI 时钟源来自于 APBS。

TWI 传输数据的方式包括包传输和 DMA 运输。

2.2 相关术语介绍

术语	解释说明
TWI	Normal Two Wire Interface，全志平台兼容 I2C 标准协议的总线控制器

2.3 模块配置介绍

TWI 模块寄存器的基本配置位于 sunxi_hal_twi.h 文件里面，包括每个 TWI 的寄存器地址和中断号，部分配置如下：

```
#define SUNXI_TWI0_PBASE 0x05002000 //寄存器地址
#define SUNXI_TWI1_PBASE 0x05002400
#define SUNXI_TWI2_PBASE 0x05002800
#define SUNXI_TWI3_PBASE 0x05002c00
#define SUNXI_S_TWI0_PBASE 0x07081400
//TWI中断
#define SUNXI_GIC_START 32
#define SUNXI_IRQ_TWI0 (SUNXI_GIC_START + 41)
#define SUNXI_IRQ_TWI1 (SUNXI_GIC_START + 42)
#define SUNXI_IRQ_TWI2 (SUNXI_GIC_START + 43)
#define SUNXI_IRQ_TWI3 (SUNXI_GIC_START + 44)
#define SUNXI_IRQ_S_TWI0 (SUNXI_GIC_START + 107)
//TWI的引脚配置
#define TWI_PIN_NUM 2 /*SCK,SDA
#define TWI0_PIN_MUXSEL 5
#define TWI1_PIN_MUXSEL 5
#define TWI2_PIN_MUXSEL 4
#define TWI3_PIN_MUXSEL 5
#define S_TWI0_PIN_MUXSEL 3
#define TWI_DISABLE_PIN_MUXSEL 7
#define TWI_PULL_STATE 1
#define TWI_DRIVE_STATE 0
#define TWI0_SCK GPIOI(3)
#define TWI0_SDA GPIOI(4)
#define TWI1_SCK GPIOI(1)
#define TWI1_SDA GPIOI(2)
#define TWI2_SCK GPIOH(5)
#define TWI2_SDA GPIOH(6)
#define TWI3_SCK GPIOH(13)
#define TWI3_SDA GPIOH(14)
#define S_TWI0_SCK GPIOL(0)
#define S_TWI0_SDA GPIOI(1)
```

2.4 模块配置介绍

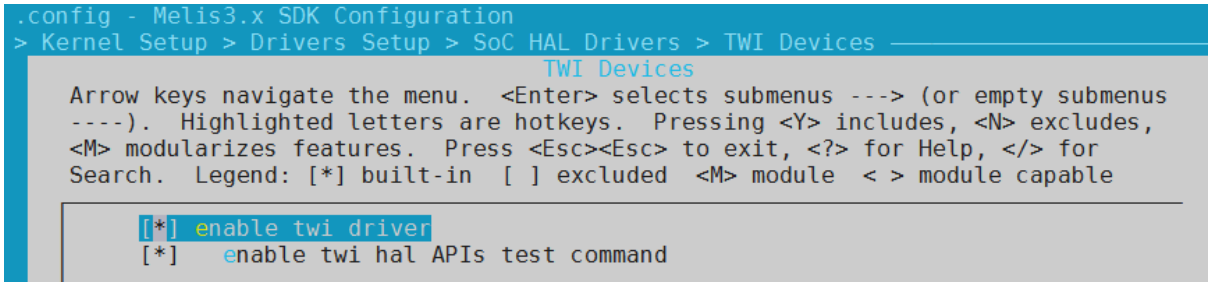

图 2-2: TWI menuconfig

2.5 模块源码结构

TWI 模块源码结构如下所示：

```
rtos-hal/
|--hal/source/twi/hal_twi.c //hal层接口代码
|--include/hal/sunxi_hal_twi.h //头文件
```

2.6 模块结构说明

TWI 模块提供给用户配置的主要包括 TWI 的端口号，TWI 模块的一些控制用法以及发送的消息，所以重点讲解这几个数据结构，想要了解更多的数据结构可以查看 sunxi_hal_twi.h 文件。

2.6.1 TWI 端口号 twi_port_t

该数据结构主要用来表明 TWI 的编号，用户可以用来调用 TWI 的控制器。具体定义如下：

```
typedef enum
{
    TWI_MASTER_0, /**< TWI master 0. */
    TWI_MASTER_1, /**< TWI master 1. */
    TWI_MASTER_2, /**< TWI master 2. */
    TWI_MASTER_3, /**< TWI master 3. */
    S_TWI_MASTER_0, /**< S_TWI master 0. */
    TWI_MASTER_MAX /**< max TWI master number, \<invalid\> */
} twi_port_t;
```

2.6.2 TWI 消息结构体 twi_msg_t

该数据结构是 TWI 通信时的消息结构，定义每个通信数据的格式：

```
typedef struct twi_msg
{
    uint16_t addr; //从设备地址
    uint16_t flags; //flag，可以定义的看下面两个宏
    #define TWI_M_RD 0x0001 /* read data, from slave to master (读)
                             * TWI_M_RD is guaranteed to be 0x0001! (写)
                             **/
    #define TWI_M_TEN 0x0010 /* this is a ten bit chip address  (10位地址) */
    uint16_t len; /* msg length */
    uint8_t *buf; /* pointer to msg data */
} twi_msg_t;
```

2.6.3 TWI 控制结构体 hal_twi_transfer_cmd_t

该数据接口储存了一些用户在调用 twi_control 的时候可以用到的一些参数，具体如下：

```
typedef enum
{
    I2C_SLAVE = 0, //设置从机地址
    I2C_SLAVE_FORCE = 1, //强制设置从机地址
    I2C_TENBIT = 2, //支持10位地址
    I2C_RDWR = 3 //读写支持
} hal_twi_transfer_cmd_t;
```

其他的关于 TWI 的时钟频率，使用的引脚都已经在软件上面配置了，如果需要更改，可以参照《模块配置介绍》章节进行修改。

3 模块接口说明

3.1 接口列表

TWI 提供的接口列表如下：

```
twi_status_t hal_twi_init(twi_port_t port);
twi_status_t hal_twi_uninit(twi_port_t port);
twi_status_t hal_twi_write(twi_port_t port, unsigned long pos, const void *buf, uint32_t
    size);
twi_status_t hal_twi_read(twi_port_t port, unsigned long pos, void *buf, uint32_t size);
twi_status_t hal_twi_control(twi_port_t port, hal_twi_transfer_cmd_t cmd, void *args);
```

3.2 接口使用说明

3.2.1 TWI 初始化接口

- 原型：twi_status_t hal_twi_init(twi_port_t port)
- 功能：TWI 模块初始化，主要初始化时钟，中断以及引脚配置等
- 参数：
  - port：TWI 端口号
- 返回值：
  - 0 代表成功
  - 负数代表失败

3.2.2 TWI 控制接口

- 原型：twi_status_t hal_twi_control(twi_port_t port, hal_twi_transfer_cmd_t cmd, void *args)
- 功能：更改 TWI 的一些配置，包括从设备地址以及读写数据等
- 参数：
  - port：端口号
  - cmd：控制参数

- args：传入的配置数据

- 返回值：
  - 0 代表成功
  - 负数代表失败

3.2.3 TWI 数据发送接口

- 原型：twi_status_t hal_twi_write(twi_port_t port, unsigned long pos, const void *buf, uint32_t size)
- 功能：用于发送数据
- 参数：
  - port：通道号
  - pos：偏移量（目前支持 1 个字节大小）
  - buf：待发送数据，size：发送数据大小，不包括偏移量
- 返回值：
  - 0 代表成功
  - 负数代表失败

3.2.4 TWI 数据接收接口

- 原型：twi_status_t hal_twi_read(twi_port_t port, unsigned long pos, void *buf, uint32_t size)
- 功能：用于接收数据
- 参数：port：通道号，pos：偏移量（目前支持 1 个字节大小），buf：接收的数据，size：接收数据大小，不包括偏移量

3.2.5 TWI 去初始化接口

- 原型：twi_status_t hal_twi_deinit(twi_port_t port)
- 功能：TWI 模块去初始化
- 参数：
  - port：TWI 端口号
- 返回值：
  - 0 代表成功
  - 负数代表失败

4 模块使用范例

可参考驱动 APIs 测试代码（hal/test/twi/）。

5 FAQ

著作权声明

商标声明

免责声明