

Rockchip RT-Thread SPI

文件标识: RK-KF-YF-093

发布版本: V1.0.1

日期: 2020-05-27

文件密级: ☐绝密 ☐秘密 ☐内部资料 ☒公开

免责声明

本文档按“现状”提供, 福州瑞芯微电子股份有限公司(“本公司”, 下同)不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因, 本文档将可能在未经任何通知的情况下, 不定期进行更新或修改。

商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标, 归本公司所有。

本文档可能提及的其他所有注册商标或商标, 由其各自拥有者所有。

版权所有 © 2019 福州瑞芯微电子股份有限公司

超越合理使用范畴, 非经本公司书面许可, 任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部, 并不得以任何形式传播。

福州瑞芯微电子股份有限公司

Fuzhou Rockchip Electronics Co., Ltd.

地址: 福建省福州市铜盘路软件园A区18号

网址: www.rock-chips.com

客户服务电话: +86-4007-700-590

客户服务传真: +86-591-83951833

客户服务邮箱: fae@rock-chips.com

前言

概述

本文主要描述了 ROCKCHIP RT-Thread SPI 驱动的使用方法。

产品版本

芯片名称	内核版本
所有使用 RK RT-Thread SDK 的芯片产品	RT-Thread

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师 软件开发工程师

修订记录

日期	版本	作者	修改说明
2019-07-13	V1.0.0	赵仪峰	初始发布
2020-05-27	v1.0.1	赵仪峰	修订格式

目录

Rockchip RT-Thread SPI

1 Rockchip SPI 功能特点

2 软件

2.1 代码路径

2.2 配置

2.3 SPI测试

2.4 SPI 使用配置

1 Rockchip SPI 功能特点

SPI (Serial Peripheral Interface)

- 支持4种SPI模式
- 支持2个片选
- 支持8bits 和 16bits 传输
- 支持中断传输模式和DMA传输模式
- 32级FIFO深度 (部分芯片是64级)
- 数据采样时钟RXD可配置

2 软件

2.1 代码路径

框架代码:

```
1 components/drivers/include/drivers/spi.h
2 components/drivers/spi/spi_core.c
3 components/drivers/spi/spi_dev.c
4 components/drivers/spi/qspi_core.c
```

串口驱动适配层:

```
1 bsp/rockchip-common/drivers/drv_spi.c
2 bsp/rockchip-common/drivers/drv_spi.h
```

串口测试命令, 串口用户程序完全可以参照以下驱动:

```
1 bsp/rockchip-common/tests/spi_test.c
```

2.2 配置

打开串口配置, 同时会生成/dev/spi0..2设备。

```
1 RT-Thread bsp drivers --->
2     RT-Thread rockchip "project" drivers --->
3         [*] Enable SPI
4         [ ] Enable SPI0 (SPI2APB)
5         [*] Enable SPI1
6         [*] Enable SPI2
```

2.3 SPI测试

使能SPI测试程序:

```
1 RT-Thread bsp test case --->
2     [*] RT-Thread Common Test case --->
3         [*] Enable BSP Common TEST
4         [*] Enable BSP Common SPI TEST
```

SPI测试命令:

```
1 1. config spi_device: op_mode, spi_mode, bit_first, speed:
2   op_mode: 0 -> master mode, 1 -> slave mode
3   spi_mode: 0 - 3 -> RT_SPI_MODE_0 ~ RT_SPI_MODE_3
4   bit_first: 0 -> LSB, 1 -> MSB
5   speed: config spi clock, the units is Hz
6   /* config spi1 cs0 master mode, spi mode 0, LSB, spi clock 1 MHz*/
7   example: spi_test config spi1_0 0 0 0 1000000
8 2. write/read/loop spi_device: times, size like:
9   /* write spi1 cs0 1024 bytes 1 time*/
10  example: spi_test write spi1_0 1 1024
11  /* read spi1 cs1 1024 bytes 10 time */
12  example: spi_test read spi1_1 10 1024
13  /* loop back mode test spi2 cs0 1024 bytes 10 times */
14  example: spi_test loop spi2_0 10 1024
```

2.4 SPI 使用配置

SPI控制器作为MASTER时可以支持0-50MHz（个别平台可以配置更高频率），作为SLAVE时可以支持0-20Mhz。

框架提供的配置函数 `rt_spi_configure()` 可以配置频率、模式和传输位宽等。

SPI支持4种模式，具体使用哪种模式，参考设备手册。

4种模式定义如下：

```
1 #define RT_SPI_MODE_0          (0 | 0)          /* CPOL = 0, CPHA
   = 0 */
2 #define RT_SPI_MODE_1          (0 | RT_SPI_CPHA) /* CPOL = 0, CPHA
   = 1 */
3 #define RT_SPI_MODE_2          (RT_SPI_CPOL | 0) /* CPOL = 1, CPHA
   = 0 */
4 #define RT_SPI_MODE_3          (RT_SPI_CPOL | RT_SPI_CPHA) /* CPOL = 1, CPHA
   = 1 */
```

配置代码示例：

```
1 struct rt_spi_configuration cfg;
2
3 cfg.data_width = 8; /* 配置8bits传输模式 */
4 cfg.mode = RT_SPI_MASTER | RT_SPI_MSB | RT_SPI_MODE_0;
5 cfg.max_hz = 20 * 1000 * 1000; /* 配置频率 20Mhz */
6 rt_spi_configure(spi_device, &cfg); /* 配置 SPI*/
```