Rockchip
瑞芯微电子

Rockchip
SPI 开发指南

发布版本:0.01

日期:2016.07

前言

概述

产品版本

芯片名称	内核版本
RK3328	3.10
RK3228H	3.10

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
2016-06-29	V0.1	洪慧斌	

目录

1 Rockchip SPI 功能特点

SPI （serial peripheral interface），以下是 linux 3.10 spi 驱动支持的一些特性：

- 默认采用摩托罗拉 SPI 协议
- 支持 8 位和 16 位
- 软件可编程时钟频率和传输速率高达 50MHz
- 支持 SPI 4 种传输模式配置
- 每个 SPI 控制器只支持一个片选

2 DTS 节点配置

```
&spi1 {       引用 spi 控制器节点
status = "okay";
max-freq = <48000000>;    spi 内部工作时钟，不超过 50M
//dma-names = "tx", "rx";    使能 DMA 模式，一般通讯字节少于 32 字节的不建议用
        spi_test@10 {
                compatible = "rockchip,spi_test_bus1_cs0";
                reg = <0>;    片选 0 或者 1
                spi-max-frequency = <24000000>;     spi clk 输出的时钟频率
                spi-cpha;    如果有配，cpha 为 1
                spi-cpol;    如果有配，cpol 为 1
                spi-cs-high;  如果有配，每传完一个数据，cs 都会被拉高，再拉低
                spi-3wire;
                poll_mode = <0>;   1 采用中断模式
                enable_dma = <1>; 1 使能 DMA 模式
                //status = "disabled";
         };
        一般只需配置以下几个属性就能工作了。
         spi_test@00 {
                  compatible = "rockchip,spi_test_bus0_cs0";
                  reg = <0>;
                  spi-max-frequency = <50000000>;
                  poll_mode = <0>;
                  enable_dma = <1>;
                   type = <0>;
              };

};
```

3 代码使用 SPI 接口

设备驱动注册:

```
static int slt_spi_test_probe(struct spi_device *spi)
{
    int ret;
    int id = 0;

    if (!spi)
        return -ENOMEM;

    spi->bits_per_word = 8;

    ret = spi_setup(spi);
    if (ret < 0) {
        dev_err(&spi->dev, "ERR: fail to setup spi\n");
        return -1;
    }
    return ret;
}

static int slt_spi_test_remove(struct spi_device *spi)
{
    slt_spi_printk("%s\n", __func__);
    return 0;
}

static const struct of_device_id slt_spi_test_dt_match[] = {
    { .compatible = "slt,spi_test_bus1_cs0", },
    { .compatible = "slt,spi_test_bus1_cs1", },
    {},
};
MODULE_DEVICE_TABLE(of, slt_spi_test_dt_match);

static struct spi_driver slt_spi_test_driver = {
    .driver = {
        .name  = "slt_spi_test",
        .owner = THIS_MODULE,
        .of_match_table = of_match_ptr(slt_spi_test_dt_match),
    },
    .probe = slt_spi_test_probe,
    .remove = slt_spi_test_remove,
};
```

```
static int __init slt_spi_test_init(void)
{
    int ret= 0;
    ret = spi_register_driver(&slt_spi_test_driver);
    return ret;
}
device_initcall(slt_spi_test_init);

static void __exit slt_spi_test_exit(void)
{
    return spi_unregister_driver(&slt_spi_test_driver);
}
module_exit(slt_spi_test_exit);
```

对 spi 读写操作请参考 include/linux/spi/spi.h，以下简单列出几个

```
static inline int
spi_write(struct spi_device *spi, const void *buf, size_t len)
static inline int
spi_read(struct spi_device *spi, void *buf, size_t len)
static inline int
spi_write_and_read(struct spi_device *spi, const void *tx_buf, void *rx_buf, size_t
len)
```

4 常见问题

1. 调试前确认驱动有跑起来，确保 SPI 4 个引脚的 IOMUX 配置无误。
2. 确认 TX 送时，TX 引脚有正常的波形，CLK 有正常的 CLOCK 信号，CS 信号有拉低