Rockchip RT-Thread SPIFLASH

文件标识：RK-KF-YF-080

发布版本：V1.0.1

日期：2020-02-21

文件密级：□绝密 □秘密 □内部资料 ■公开

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 Flash 的原理和使用方法。

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

版本号	作者	修改日期	修改说明
V1.0.0	林鼎强	2020-02-21	初始版本
V1.0.1	陈谋春	2020-03-05	修改无序列表的缩进

目录

1 简介

1.1 支持器件

RK MCU 产品使用的 SPI flash 仅为 SPI Nor，不支持 SPI Nand。

SPI Nor 具有封装小、读速率相较其他小容量非易失存储更快、引脚少和协议简单等优势，市面通用的 SPI Nor 支持1线、2线和4线传输，且 SPI Nor 具有不易位翻转、byte 寻址和读操作无等待延时（发送 cmd 和 address 下一拍就传输数据）的特点，所以支持使用 XIP技术 （Excute In Place）。

RK RTOS SPI flash 框架提供通用的 SPI Nor 接口和自动化的 XIP 方案。

1.2 主控控制器

RK 平台 SPI Flash 可选用的控制器包括 FSPI 和 SPI 两种方案。

FSPI (Flexible Serial Peripheral Interface) 是一个灵活的串行传输控制器，有以下主要特性：

- 支持 SPI Nor、SPI Nand、SPI 协议的 Psram 和 SRAM
- 支持 SPI Nor 1线、2线和4线传输
- XIP 技术
- DMA 传输

SPI （ Serial Peripheral Interface ）为通用的 串行传输控制器，可以支持外挂 SPI Nor、SPI Nand，RK RTOS 平台目前仅支持 SPI Nor 的实现。

1.3 XIP 技术

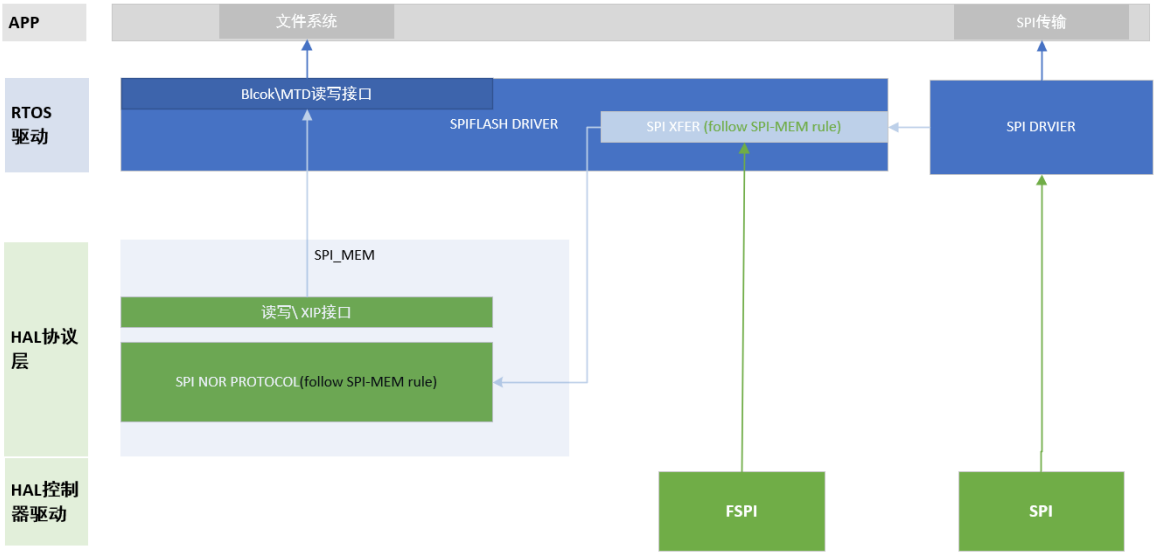
XIP（eXecute In Place），即芯片内执行，指 CPU 直接通过映射地址的 memory 空间取指运行，即应用程序可以直接在 flash 闪存内运行，不必再把代码读到系统 RAM 中，所以片内执行代码的运行地址为相应的映射地址。由于 SPI Nor XIP 仅支持读，所以只能将代码段和只读信息置于 SPI Nor 中。

FSPI 除支持 CPU XIP 访问 SPI flash，还支持如DSP 等其他模块以相近方式获取 flash 数据，如同访问一片“只读的 sram”空间，详细 FSPI 信息参考 TRM 中 FSPI 章节。

1.4 驱动框架

考虑到要适配 FSPI 和 SPI 两种控制器，所以抽象出控制器层，从而将整个驱动框架分为四个层次 ：

- MTD 框架层
- RTOS Driver 层，完成以下逻辑:
  - RTOS 设备框架注册
  - 注册控制器及操作接口到 HAL_SNOR 协议层
  - 封装读写擦除接口给用户
- 基于 SPI Nor 传输协议的 HAL_SNOR 协议层
- 控制器层



基于 FSPI 控制器的 RT-Thread 实现：

- OS 驱动层：drv_snor.c 实现:
  - 基于 FSPI HAL层读写接口封装 SPI_Xfer，并注册 FSPI host 及 SPI_Xfer 至 HAL_SNOR 协议层
  - 封装 HAL_SNOR 协议层提供的读写擦除接口
  - 注册 OS 设备驱动到 MTD 框架层
- 协议层：HAL 开发包中的 hal_snor.c 实现 SPI Nor flash 的协议层
- 控制器层：HAL 开发包中的 hal_fspi.c 实现 FSPI 控制器驱动代码

基于 SPI 控制器的 RT-Thread 实现：

- OS 驱动层：drv_snor.c 实现:
  - 基于 SPI OS driver 读写接口封装 SPI_Xfer，并注册 SPI host 和 SPI_Xfer 至 HAL_SNOR 协议层；
  - 封装 HAL_SNOR 协议层提供的读写擦除接口
  - 注册 OS 设备驱动到 MTD 框架层
- 协议层：HAL 开发包中的 hal_snor.c 实现 SPI Nor flash 的协议层
- 控制器层：HAL 开发包中的 hal_spi.c 实现 SPI 控制器 low layer 驱动代码，SpiDevice.c 代码实现 RTOS SPI DRIVER 的设备注册和接口封装

注意：

1. 以上实现一一对应附图，可结合阅读
2. 由于 RK SPI DMA 传输相关代码在 OS Driver 层，且 SPI 控制器除了应用在 SPI Nor 上，还支持较多其他器件，存在硬件资源边界保护，所以在 SPI Flash 框架中的 SPI 控制器不应直接套接 HAL 层 hal_spi.c 驱动，而应使用 OS Driver 中的 SPI 接口。

2 配置

SPI Flash 驱动框架所有配置都能通过 Kconfig 进行灵活调整，如 1.4 章节所讲，SPI flash 完整的驱动框架由四个抽象层构成，相应的配置也分为四个层次。

2.1 MTD 框架配置

```
RT-Thread Components  --->
    Device Drivers  --->
        -*- Using MTD Nor Flash device drivers
```

2.2 SPI Flash Driver 驱动配置

在进行该项配置前，需明确硬件上所选用的 SPI Flash 相应的主控类型，以选择相应方案;

FSPI 控制器方案：

```
RT-Thread rockchip common drivers  --->
    [*] Enable ROCKCHIP SPI NOR Flash
    (80000000) Reset the speed of SPI Nor flash in Hz
    [ ]   Set SPI Host DUAL IO Lines /* 如果 FSPI 主控仅预留 IO0~1,应使用 Dual mode */
            Choose SPI Nor Flash Adapter (Attach FSPI controller to SNOR) --->
                (X) Attach FSPI controller to SNOR
```

SPI 控制器方案：

```
RT-Thread rockchip common drivers  --->
    [*] Enable ROCKCHIP SPI NOR Flash
    (80000000) Reset the speed of SPI Nor flash in Hz
    [ ]   Set SPI Host DUAL IO Lines /* 如果 cFSPI 主控仅预留 IO0~1,应使用 Dual mode */
    (spi1_0)  the name of the SPI device which is used as SNOR adapte /* 指定 SPI 控制器 #SPIID_#CS */
```

```
RT-Thread rockchip pisces drivers  --->
        Enable SPI  --->
        [*] Enable SPI1 /* 配置相应的 SPI 控制器 */
```

3 分区及文件系统配置

此处仅作简单的配置说明，详细可参考请参考 rockchip 目录下的《Rockchip_Developer_Guide_RT-Thread_CN.md》，4.7 到 4.8 章节。

3.1 分区信息生成、解析和 OS 分区注册

RK RT-Thread SDK 打包后的固件会在 flash 前 2KB 位置生成 RK_PARTITION 分区表，可以通过修改相应的 setting.ini 来调整分区信息。

随固件下载到 SPI Nor 的分区表将在 SPI Nor 初始化成功后进行探测解析并注册（代码在 drv_snor.c 中的 snor_partition_init 函数中实现）。

可以通过以下命令查看相应分区是否挂载成功：

```
msh />list_device
device          type         ref count
------- -------------------- ----------
root    Block Device         1           /* 分区名 root，分区类型 block 设备 */
snor    MTD Device           0           /* 存储设备，分区读写最终接入到该节点完成读写擦除 */
```

3.2 elm-fat 文件系统及分区挂载文件系统

RT-Thread elm-fat 文件支持

```
RT-Thread Components --->
    Device virtual file system  --->
    [*] Using device virtual file system
    [*]   Using mount table for file system /* 实现相应注册分区表，可实现分区上电自动挂载 */
    [*]   Enable elm-chan fatfs /* fat 文件系统 */
        elm-chan's FatFs, Generic FAT Filesystem Module  --->
        (4096) Maximum sector size to be handled. /*对于 SPI Nor 产品必须修改为 4096 */
```

如开启分区自动挂载文件系统，可在 mnt.c 中添加相应分区注册信息，例如“root”分区到“/”目录：

```
const struct dfs_mount_tbl mount_table[] =
{
    {"root", "/", "elm", 0, 0},
    {0}
};
```

如希望自行设计文件系统挂载流程，也可以通过以下代码实现文件系统挂载：

```
dfs_mount("root", "/", "elm", 0, 0)
```

4 XIP 实现方案须知

前面已经介绍 SPI Nor 支持 XIP 功能，如果选用 FSPI 主控实现的 SPI Nor 方案，会自动开启 XIP 功能，以下介绍产品应用中涉及到 XIP 的一些须知。

4.1 添加 XIP 支持

当选用 FSPI 实现的 SPI Flash 方案，并按照 1.2 章节中关于 FSPI 配置方法去配置，SPI Flash 将默认配置使用 XIP 功能，如要关闭该功能，应则关闭配置 “Enable FSPI XIP”

4.2 XIP 使用过程的开关

RK SPI Flash 框架会在需求的场景自动开关 XIP 功能，客户无需调用开关接口，详细如下:

XIP 开

由于 SPI Nor 擦除/写耗时长的特点， SPI Nor 不支持 XIP 下的擦除/写，所以当 SPI Nor flash 有擦除和写请求的时候，如文件系统写请求， 软件会调用 XIP suspend 接口切换 SPI nor 主控 FSPI 到 normal mode，期间将无法使用 XIP 功能，完整的 suspend XIP 切换流程如下：

1. 通知所有受 XIP disable 影响的 master 模块挂起 XIP 操作
2. 关闭全局中断，避免产生中断导致 CPU 执行放在 SPI Nor 中的 XIP 的代码
3. 关闭 XIP

XIP 关

当 SPI Nor flash 擦除/写完成且 FSPI idle 后 SPI Flash 驱动将重新恢复 XIP 功能，也就是在没有擦除写操作的情况下， FSPI 一直将使能 XIP 功能，完整的 XIP resume 流程如下：

1. 开启XIP

2. 使能全局终端
3. 通知所有 XIP suspend 设备恢复使用 XIP

以上所述操作入口如下：

```
static rt_base_t snor_xip_suspend(void)
static void snor_xip_resume(rt_base_t level)
```

总结

- 如果 SPI Nor 颗粒由 FSPI 主控驱动，默认使能 XIP 功能
- SPI Nor 主控 FSPI 在 idle 状态默认开启 XIP 功能，此时支持 XIP 通路的设备都可访问 XIP memory 映射地址获取 SPI Nor 上的数据
- SPI Nor 主控 FSPI 在进行擦除和写行为时关闭 XIP 功能
- SPI Nor 主控 FSPI 开关 XIP 过程，需通知相应设备停止/恢复 XIP 访问，具体按照上文 “XIP 关” 所述操作
- XIP 关，同时会关闭全局中断

5 函数接口调用范例

参考 snor_test.c。

6 测试驱动

建议 SPI Flash 开发流程中引入以下测试流程，做简单的读写测试判断。

6.1 配置

```
    RT-Thread bsp test case  --->
        RT-Thread Common Test case  --->
            [*] Enable BSP Common TEST
            [*]   Enable BSP Common SNOR TEST (NEW)
```

如配置成功，在 msh 中会有 snor_test 的 命令选项。

6.2 测试命令

输入命令 snor_test 可以获取详细的说明，以下命令单位皆为 byte。

```
snor_test write offset size /* 写固定数据 pattern 到 flash 指定区域 */
snor_test read offset size  /* 读 flash 指定区域数据，并打印出来 */
snor_test erase offset      /* 擦除 flash 指定区域数据 */
snor_test stress offset size loop /* 读写测试 */
snor_test io_stress offset size loop /* IO 读写测试 */
```

7 常见问题

- **如何判断 SPI flash 已经挂载成功？**

  通过 list_device 查看是否有 snor 设备节点。

- **Init adapte error ret= -19 是什么报错？**

  “Init adapte error ret= -19” 为存储驱动的初始化函数 snor_adapt 的返回结果打印，相应的错误码解释如下：

a. -1: SPI Nor ID 为 0xff，可能为 SPI Nor 没有焊接良好，请检查一下电路和信号；

b. -19: 该颗粒不在支持列表，请联系 RK 工程师，添加相应颗粒支持；

c. -22: SPI 控制器没有找到。

- **常开 XIP 功能是否有多余功耗？**

  FSPI 有 time out 机制，主控 idle 一定时长后，自动释放 cs，SPI Nor 将进入 idle 的低功耗状态，所以不会有较大功耗差异。

- **如果多个 master 同时通过 XIP 访问 SPI Nor，是否会有冲突？**

  多个 master 同时访问 XIP，总线会做仲裁，串行完成传输，请求会挂在总线上。

- **对于频繁读写文件系统的产品，是否应该使用 XIP 功能？**

  对于频繁写文件系统的产品，建议将代码段至于 sram 或 psram 中，然后关闭 FSPI XIP 功能，否则将影响 RTOS 系统实时性，同样，使用 XIP 运行 code 段的方案，应减少 SPI Nor 的擦除和写。

- **使能 XIP 功能后，文件系统读，是否关 XIP ？**

  使能 XIP 功能后，文件系统读，驱动实现直接通过 XIP 访问，所以不关 XIP。

- **如果 SPI Nor 上没有分区表，还能挂载分区吗？**

  必须固化 flash 分区到 flash 前端，RK 目前 SPI Nor flash 存储驱动仅做固化在 flash 中的分区解析挂载方案。