指纹驱动配置指南

文档版本 V1.0

发布日期 2020-10-21

紫光展锐（上海）科技有限公司

紫光展锐（上海）科技有限公司

前 言

概述

本文档用于指导客户基于 TEE 指纹方案完成 dts 配置、pinmap 配置及调试等工作。

读者对象

本文档适用于需要适配指纹的终端厂商。

缩略语

缩略语	英文全名	中文解释
DTS	Device Tree Source	设备树源
DTO	Device Tree Overlay	设备树叠加层
FDT	Flattened Device Tree	扁平化设备树
REE	Rich Execution Environment	富执行环境
TEE	Trusted Execution Environment	可信执行环境
SPI	Serial Peripheral Interface	串行外设接口

变更信息

文档版本	发布日期	作者	修改说明
V1.0	2020-10-21	Mingbo.yang	第一次正式发布。

关键字

Fingerprint、指纹、指纹驱动。

目 录

1 新增指纹驱动

以基于 ums512 配置 microarray 指纹驱动为例进行说明，需要先根据原理图配置 dts，检查 pinmap，确保驱动正确编译。

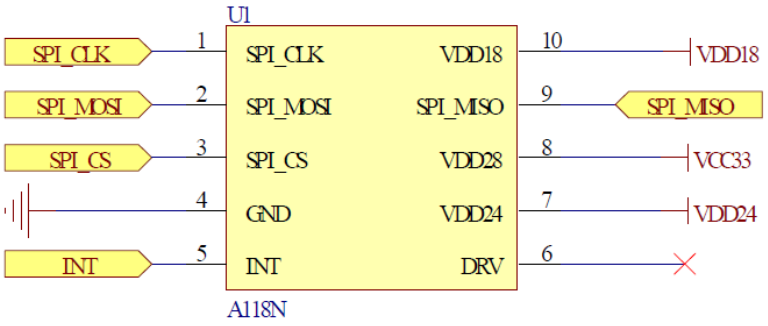
1.1 接口/SPI 确认

SPI 总线系统是一种同步串行外设接口，它可以使 CPU 与各种外围设备以串行方式进行通信以交换信息。一般主控 SoC 作为 SPI 的“主”，而外设作为 SPI 的“从”。

SPI 接口一般使用 4 条线：

- SCK : Serial Clock 串行时钟
- MOSI : Master Output, Slave Input 主发从收信号
- MISO : Master Input, Slave Output 主收从发信号
- SS/CS : Slave Select 片选信号

通过指纹传感器 datasheet 查看指纹引脚。



从上图可以看出，该方案涉及 4 个 SPI 引脚（SPI_CLK/SPI_MOSI/SPI_MISO/SPI_CS），1 个中断引脚，3 个 VDD 供电引脚（平台方案是长供电源，无需额外配置），无 reset 引脚。

根据指纹和主芯片 SPI 连接部分的原理图可知各个 SPI 引脚连接到主芯片的接口，以 ums512 的指纹中断接口为例，中断引脚连接到主芯片的 IIS1 接口的 IIS1DI 引脚，该引脚支持 4 个 function，IIS1DI/SE_GPIO0/EXTINT13/GPIO130，中断引脚选择 EXTINT13。

平台默认指纹 SPI 引脚连接到主芯片的 SPI0，如需修改请联系平台确认。

1.2 dts 配置

1) 确认接口/引脚之后，需要在 dts 中做配置，配置文件在以下路径(以实际项目为准)：

32 位芯片：bsp/kernel/kernel4.14/arch/arm/boot/dts/

64 位芯片：bsp/kernel/kernel4.14/arch/arm64/boot/dts/sprd/

2) 例如 ums512 指纹方案使用的 SPI0，则需要使能 SPI0，在 ums512.dts 添加如下信息（spi 以项目实际引脚为准）：

```
&spi0 {
    /*used for fingerprint sensor*/
    status = "okay";
};
```

默认 status 值为 disable，改为 okay 为使能的意思，会覆盖默认值。

3) 添加指纹 dto 节点，在 ums512-1h10-overlay.dts 中添加：

```
fragment {
    target-path = "/";
    __overlay__ {
        sprd-wlan {
            compatible = "sprd,sc2355-wifi";
        };

        fingerprint@0 {
            compatible = "microarray,afs121";
            fpint-gpios = <&eic_async 13 0>;
        };
```

fingerprint@0 @后是设备地址，平台方案 spi 通过片选而来不需要指定设备地址，@0 也可以没有。

Compatible 的值与驱动代码中定义的相配：

bsp/modules/kernel4.14/input/fingerprint/microarray/x86-settings.h

```
#define MA_DTS_NAME        "microarray,afs121"
```

bsp/modules/kernel4.14/input/fingerprint/microarray/x86-settings.c

```
static struct of_device_id sof_match[] = {
    { .compatible = MA_DTS_NAME, },     /*this name is used for matching
the dts device for settings the gpios*/
    { },
};
```

4) Gpio 配置：

驱动代码中有两个 gpio：fpint-gpios 和 fpen-gpios，但目前 dts 只配置了 fpint-gpios，原因是 fpen-gpios 这个是供电引脚，平台方案是长供电源，不需要配置。如果需要使用 GPIO 开关控制供电，要根据实际供电情况修改代码。

bsp/modules/kernel4.14/input/fingerprint/microarray/x86-settings.c

```
126    int_gpio = of_get_named_gpio(np, "fpint-gpios", 0);
……
168    en_gpio = of_get_named_gpio(np, "fpen-gpios", 0);
```

中断唤醒有高电平和上升沿脉冲两种方式，gpio（ap_gpio）只支持高电平唤醒，eic（ap_eic/ap_eic_async）和 gpio plus（gpio_plus）支持两种方式唤醒。因为指纹需要在系统深睡时唤醒，建

议用 EIC_ASYNC 的方式来支持深睡时的沿触，所以此处选用 eic_async，后面的 13 则是对应前面原理图中的中断 funtion EXTINT13。

```
fingerprint@0 {
    compatible = "microarray,afs121";
    fpint-gpios = <&eic_async 13 0>;
};
```

1.3 pinmap 配置

1) pinmap 里面，包括：pin 的

- Function 功能选择(Function 0~3)
- 上下拉设置(WPD ,WPU , x)
- 驱动能力设置(DS 0~3), 2/6/12/24mA 对应驱动能力 0/1/2/3，3 为最强。
- Sleep 时的上下拉设置(WPD,WPU,X)
- Sleep 时的输入输出设置(Input,Output,Hiz)
- 强上拉的设置(WPUS, x)
- AP 或 CP 的 Sleep 控制--AP+CP 架构(AP,CP0~CP2)
- 其他(PIN_CTRL0~3)

2) 关键字介绍

- REG_PIN_ SPI0_CSN /REG_MISC_PIN_SPI0_CSN：IO 口的 central/side 寄存器名称，两行共同配置一个 PIN 脚的功能属性。
- BITS_PIN_AF(0)：function 选择。规格书和 GPIO Table 里 function0～3 就是对应 AF0~3（如果看到其他地方有 function1～4 的描述，其实就是对应 AF0~3）
- BITS_PIN_DS(1)：IO 口驱动强度选择。默认请保持与平台一致。时序有问题时可以做调整。
- BIT_PIN_ WPUS：唤醒状态下 IO 口的强上拉是否开启。BIT_PIN_ WPUS 开启强上拉，BIT_PIN_ NULL 关闭强上拉。除 IIC 和 SIM 卡之外一般不需要开启，保持与平台一致。
- BIT_PIN_ WPU：唤醒状态下 IO 口的上下拉状态。BIT_PIN_ WPU 上拉，BIT_PIN_ WPD 下拉，BIT_PIN_ NUL 无上下拉。上下拉一般只给输入使用，输出管脚一般配置 BIT_PIN_ NUL。
- BIT_PIN_SLP_CM4：IO 口随哪个子系统进入睡眠。射频控制脚随 ALL_CP，普通 GPIO 随 AP，其他不要修改，保持与平台一致。
- BIT_PIN_SLP_WPU：休眠状态下 IO 口的上下拉状态。BIT_PIN_ WPU 上拉，BIT_PIN_ WPD 下拉，BIT_PIN_ NUL 无上下拉。上下拉一般只给输入使用，输出管脚一般配置 BIT_PIN_ NUL。
- BIT_PIN_SLP_Z：睡眠时的输入输出状态。BIT_PIN_SLP_IE 输入，BIT_PIN_SLP_OE 输出，BIT_PIN_SLP_Z 高阻。

3) 根据芯片 spec 信息进行配置。

Ball Name	Function0	Type	Function1	Type	Function2	Type	Function3	Type
SPI0_CSN	SPI0_CSN	I/O/T			EXTINT5(G0)	I	GPIO90	I/O/T
SPI0_DO	SPI0_DO	I/O/T			EXTINT6(G0)	I	GPIO91	I/O/T
SPI0_DI	SPI0_DI	I/O/T			EXTINT7(G0)	I	GPIO92	I/O/T
SPI0_CLK	SPI0_CLK	CLK/O/T			EXTINT8(G0)	I	GPIO93	I/O/T
EXTINT9	EXTINT9	I	KEYOUT3	O/T	BUA_TF_DET	I	GPIO78	I/O/T
EXTINT10	EXTINT10	I	KEYOUT4	O/T	BAT_DET	I	GPIO79	I/O/T
IIS1DI	IIS1DI	I	SE_GPIO0	I/O/T	EXTINT13	/	GPIO130	I/O/T
IIS1DO	IIS1DO	O/T	SE_GPIO1	I/O/T	CM4_GPIO3(G0)	I/O/T	GPIO131	I/O/T

bsp/bootloader/u-boot15/board/spreadtrum 下选择对应工程：

```
    {REG_PIN_SPI0_CSN,                    BITS_PIN_AF(0)},
    {REG_MISC_PIN_SPI0_CSN,
BITS_PIN_DS(1)|BIT_PIN_NULL|BIT_PIN_NUL|BIT_PIN_SLP_AP|BIT_PIN_SLP_WPU|BIT_PI
N_SLP_OE},//FTID_SPI_CS
    {REG_PIN_SPI0_DO,                     BITS_PIN_AF(0)},
    {REG_MISC_PIN_SPI0_DO,
BITS_PIN_DS(1)|BIT_PIN_NULL|BIT_PIN_NUL|BIT_PIN_SLP_AP|BIT_PIN_SLP_WPD|BIT_PI
N_SLP_Z},//FTID_SPI_DI
    {REG_PIN_SPI0_DI,                     BITS_PIN_AF(0)},
    {REG_MISC_PIN_SPI0_DI,
BITS_PIN_DS(1)|BIT_PIN_NULL|BIT_PIN_NUL|BIT_PIN_SLP_AP|BIT_PIN_SLP_WPD|BIT_PI
N_SLP_Z},//FTID_SPI_DO
    {REG_PIN_SPI0_CLK,                    BITS_PIN_AF(0)},
    {REG_MISC_PIN_SPI0_CLK,
BITS_PIN_DS(1)|BIT_PIN_NULL|BIT_PIN_NUL|BIT_PIN_SLP_AP|BIT_PIN_SLP_WPD|BIT_PI
N_SLP_Z},//FTID_SPI_CLK
    {REG_PIN_IIS1DI,                      BITS_PIN_AF(2)},
    {REG_MISC_PIN_IIS1DI,
BITS_PIN_DS(1)|BIT_PIN_NULL|BIT_PIN_WPD|BIT_PIN_SLP_AP|BIT_PIN_SLP_WPD|BIT_PI
N_SLP_IE},//FTID_INT
    {REG_PIN_IIS1DO,                      BITS_PIN_AF(3)},
    {REG_MISC_PIN_IIS1DO,
BITS_PIN_DS(1)|BIT_PIN_NULL|BIT_PIN_NUL|BIT_PIN_SLP_AP|BIT_PIN_SLP_NUL|BIT_PI
N_SLP_OE},//FTID_RSTN
```

一个引脚配置使用两行，第一行为 function，需要和芯片 spec 对应上，BITS_PIN_AF 后括号里面是几就是 function 几。第二行配置 IO 口驱动强度和工作状态等，如无特殊需求与平台保持一致即可。

1.4 编译与加载

1) 指纹驱动 Makefile 文件调整，参考 bsp/modules/kernel4.14/input/fingerprint/microarray/Makefile

2) bsp/device/对应的工程目录下 modules.cfg 中 BSP_MODULES_LIST 添加指纹 ko，确保 ko 正常编译生成，参考 bsp/device/sharkl5Pro/androidr/common/modules.cfg

```
BSP_MODULES_LIST="
       microarray_fp.ko
       ……
       "
```

3) 增加编译

在 device/sprd/mpool/module/security/msoc/sharkl5Pro/sharkl5Pro.mk 增加编译

```
ifeq ($(strip $(BOARD_FINGERPRINT_CONFIG)), microarray)
  include
vendor/sprd/partner/microarray/sharkl5pro/microarray_sharkl5pro.mk
 endif
```

microarray_sharkl5pro.mk 将 init. Microarray.rc 复制到目标文件，rc 中 insmod 指纹 ko 和配置相关节点权限。

```
PRODUCT_COPY_FILES += \
……\
vendor/sprd/partner/microarray/init.microarray.rc:vendor/etc/init/init.microa
rray.rc
```

4) 功能开关

Board 中配置，如 device/sprd/sharkl5Pro/ums512_1h10/module/security/md.mk

```
BOARD_FINGERPRINT_CONFIG := microarray
```


2 调试

2.1 无设备节点问题

首先检查是否有设备节点，路径：/dev。

例：crw-rw--- 1 system system 509, 0 1980-01-01 08:01 madev0

若无设备节点，可排查以下几点：

1) 编译配置问题

查看 out 目录是否有编译到，若没有编译到，则查看使用的 defconfig，kconfig，mk 等配置。

2) 驱动代码问题

查看 dts 配置是否正确，例如：compatible 配置是否和驱动定义一致

2.2 中断问题

1) 没有中断

- 确认 DTS 中断 gpio 的配置：是否有其他模块共用，用 eic 则需确认 gpio 口是否支持 eic
- 确认 pinmap 配置：function，输入输出，拉高拉底，slp 配置
- 查看 gpio 申请情况：cat /sys/kernel/debug/gpio
- 查看中断申请及响应情况：cat proc/interrupts

各参数依次显示为逻辑中断号、每个 cpu 对该中断的处理次数、中断控制器的名字、驱动中定义的中断名称。

例：97: 35 0 0 0 0 0 0 0 sprd-gpio-plus 130 Edge sprd,finger_print-eint

2) 频繁中断

- 确认中断口是否和其他模块共用
- 确认中断 pinmap 配置是否为平台参考方案，如果不是，则检查配置的 pin 是否可用

3.3 SPI 问题

1) SPI 不通

- 查看电路原理图，确认挂载 spi 几和 pinmap 的配置
- 确认 DTS 的配置，spi 及 status，clk 等

- 查看 clk 详情：cat sys/kernel/debug/clk/clk_summary

例：`ap-spi0-clk  0  0  192000000 0 0`

- 查看 spi 设备：sys/bus/spi/drivers 目录，ls 看一下

2) SPI 速率慢

- 查看 clk 详情：cat sys/kernel/debug/clk/clk_summary
- 查看 spi 设备：sys/bus/spi/drivers 目录，ls 看一下

