SPI 驱动适配说明

SPI 设备驱动和 SPI 总线适配是基于 name 匹配的，使用 DTS 的系统需要 DTS 文件 SPI 节点下 name 与驱动中 name 相匹配。（注：部分系统 DTS 中用 compatible 参数匹配，实际以 CPU 厂商说明为准）

```
2779  static struct spi_driver ch37x_driver = {
2780      .probe =    ch37x_probe,
2781      .remove =   ch37x_remove,
2782      .driver = {
2783          .name = "qcom,spi_ch37x_hcd",
2784          .owner = THIS_MODULE,
2785          .of_match_table = of_match_ptr(spi_ch37x_hcd_dt_match),
2786      },
2787  };
```

DTS 声明无误，则 spi_driver 对象的 probe 方法会顺利执行，也即 ch37x_probe 方法。

中断说明

CH374芯片INT#引脚输出的中断请求默认是低电平有效（某些边沿触发中断的单片机在过去芯片中断引脚中断状态的时候可能有问题），可以连接到单片机的中断输入引脚或普通输入引脚，单片机可以使用中断方式或查询方式获知CH374的中断请求。为了节约引脚，单片机可以不连接CH374的INT#引脚,而直接查询CH374 的中断标志寄存器REG_INTER_FLAG 获知中断。

注：Linux主机驱动使用的是中断方式，也即必须将CH374的中断引脚连接到CPU的支持中断的IO口上。中断号irq的获取一般有2种方式：

（1） DTS文件中指定，然后驱动中获取该中断号；

（2） 驱动代码中通过指定函数接口获取，如：gpio_to_irq函数；

验证方法：

驱动默认在ch37x_probe函数中申请中断：

devm_request_threaded_irq(&spi->dev, spi->irq, ch37x_irq_handler,
ch37x_irq_thread_handler,
IRQF_TRIGGER_LOW| IRQF_ONESHOT , "spi-ch37x-hcd", hcd);

当发生中断时，对应的服务函数ch37x_irq_thread_handler会被调用，可以外部给相应引脚低电平脉冲用于测试中断功能是否正常；

SPI 连接测试说明

将 ch37x_hcd_driver SPI 转 USB 主机驱动移植到 Linux/安卓系统后：

需要确认 init_ch37x_host 函数执行的初始化结果是否正常，以此来确定 SPI 连接通讯是否 OK，正常状态下各寄存器输出结果：

REG_INTER_EN:0x03

REG_SYS_CTRL:0x40

REG_USB_SETUP:0xc0

REG_HUB_SETUP:0x00

若寄存器输出异常，可先检查芯片供电和硬件连接，然后可借助于逻辑分析仪或示波器抓取

SPI 通讯波形，确定读写流程。