# 第11章 LED 模板驱动程序的改造：设备树

## 11.1 总结 3 种写驱动程序的方法

### 11.1.1 资源和驱动在同一个文件里

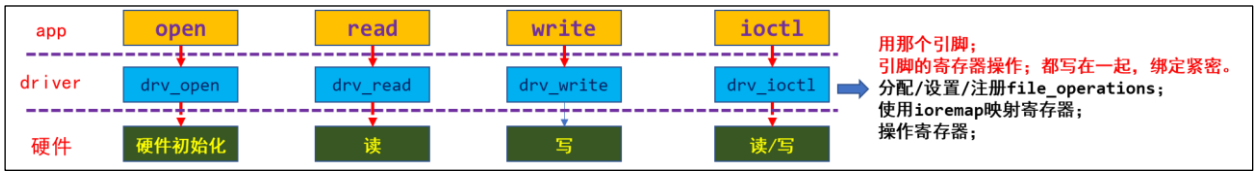


图 11.1 方法 1 传统写法

### 11.1.2 资源用 platform_device 指定、驱动在 platform_driver 实现

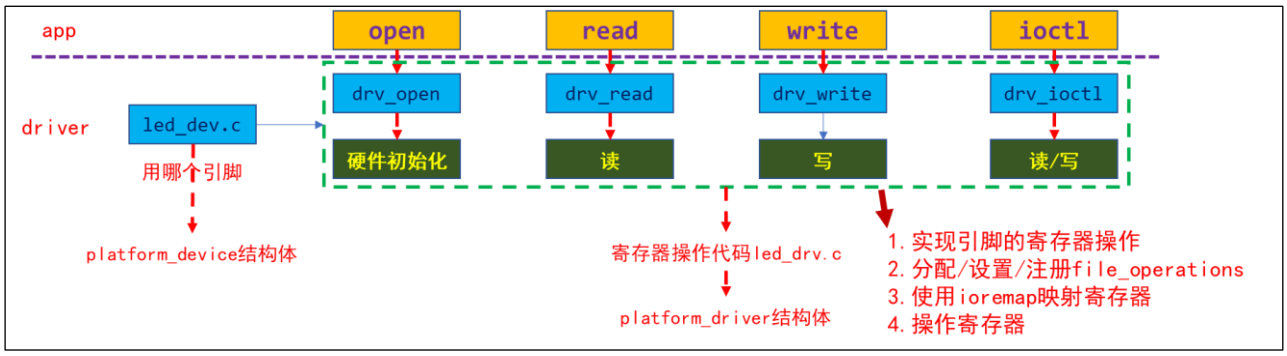


图 11.2 方法 2 分层分离

### 11.1.3 资源用设备树指定驱动在 platform_driver 实现

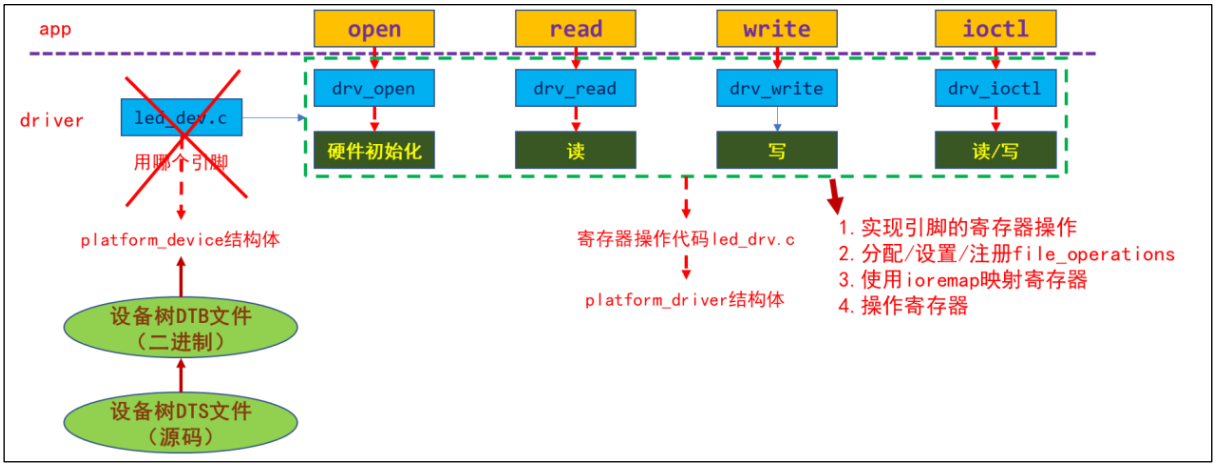


图 11.3 方法 3 使用设备树

核心永远是 file_operations 结构体。

上述三种方法，只是指定“硬件资源”的方式不一样。

从图 11.3 可以知道，platform_device/platform_driver 只是编程的技巧，不涉及驱动的核心。

## 11.2 怎么使用设备树写驱动程序

### 11.2.1 设备树节点要与 platform_driver 能匹配

在我们的工作中，驱动要求设备树节点提供什么，我们就得按这要求去编写设备树。

但是，匹配过程所要求的东西是固定的：

① 设备树要有 compatible 属性，它的值是一个字符串

② platform_driver 中要有 of_match_table，其中一项的.compatible 成员设置为一个字符串

③ 上述 2 个字符串要一致。

示例如下：

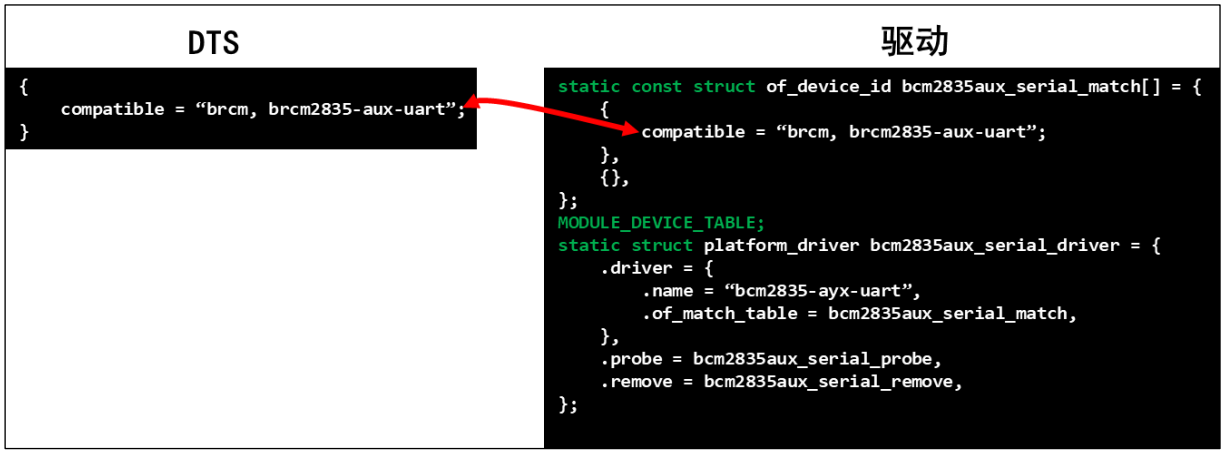


图 11.4 设备树驱动示例

### 11.2.2 设备树节点指定资源，platform_driver 获得资源

如果在设备树节点里使用 reg 属性，那么内核生成对应的 platform_device 时会用 reg 属性来设置 IORESOURCE_MEM 类型的资源。

如果在设备树节点里使用 interrupts 属性，那么内核生成对应的 platform_device 时会用 reg 属性来设置 IORESOURCE_IRQ 类型的资源。对于 interrupts 属性，内核会检查它的有效性，所以不建议在设备树里使用该属性来表示其他资源。

在我们的工作中，驱动要求设备树节点提供什么，我们就得按这要求去编写设备树。驱动程序中根据 pin 属性来确定引脚，那么我们就在设备树节点中添加 pin 属性。

设备树节点中：

```
#define GROUP_PIN(g,p) ((g<<16) | (p))
100ask_led0 {
   compatible = “100ask,led”;
   pin = <GROUP_PIN(5, 3)>;
};
```

驱动程序中，可以从 platform_device 中得到 device_node，再用

of_property_read_u32 得到属性的值：

```
struct  device_node* np = pdev->dev. of_node;
int led_pin;
int err = of_property_read_u32(np, “pin”, &led_pin);
```

## 11.3 开始编程

### 11.3.1 修改设备树添加 led 设备节点

在本实验中，需要添加的设备节点代码是一样的，你需要找到你的单板所用的设备树文件，在它的根节点下添加如下内容：

```
#define GROUP_PIN(g,p) ((g<<16) | (p))
100ask_led@0 {
    compatible = "100ask,leddrv";
    pin = <GROUP_PIN(3, 1)>;
};

100ask_led@1 {
    compatible = "100as,leddrv";
    pin = <GROUP_PIN(5, 8)>;
};
```

- 对于百问网使用 STM32MP157 板子
  - 设备树文件是：内核源码目录中 arch/arm/boot/dts/stm32mp157c-100ask-512d-lcd-v1.dts
  - 修改、编译后得到 arch/arm/boot/dts/stm32mp157c-100ask-512d-lcd-v1.dtb 文件。
  - 然后使用 nfs ssh 等方式把新编译出来的 dtb 去覆盖老文件。

### 11.3.2 修改 platform_driver 的源码

使用 GIT 下载所有源码后，本节源码位于如下目录：

```
01_all_series_quickstart\
05_嵌入式 Linux 驱动开发基础知识\
source\02_led_drv\05_led_drv_template_device_tree
```

关键代码在 chip_demo_gpio.c，主要看里面的 platform_driver，代码如下。

第 166 行指定了 of_match_table，它是用来跟设备树节点匹配的，如果设备树节点中有 compatile 属性，并且其值等于第 157 行的“100as,leddrv”，就会导致第 162 行的 probe 函数被调用。

```
156 static const struct of_device_id ask100_leds[] = {
157     { .compatible = "100as,leddrv" },
158     { },
159 };
160
161 static struct platform_driver chip_demo_gpio_driver = {
162     .probe      = chip_demo_gpio_probe,
163     .remove     = chip_demo_gpio_remove,
```

```
164     .driver     = {
165         .name   = "100ask_led",
166         .of_match_table = ask100_leds,
167     },
168 };
169
170 static int __init chip_demo_gpio_drv_init(void)
171 {
172     int err;
173
174     err = platform_driver_register(&chip_demo_gpio_driver);
175     register_led_operations(&board_demo_led_opr);
176
177     return 0;
178 }
```

## 11.4 上机实验

① 使用新的设备树 dtb 文件启动单板，查看/sys/firmware/devicetree/base 下有无节点

② 查看/sys/devices/platform 目录下有无对应的 platform_device

③ 加载驱动：

```
[root@100ask:~]# insmod  leddrv.ko
[root@100ask:~]# insmod  chip_demo_gpio.ko
```

④ 测试驱动：

```
[root@100ask:~]# ./ledtest   /dev/100ask_led0  on
[root@100ask:~]# ./ledtest   /dev/100ask_led0  off
```

## 11.5 调试技巧

/sys 目录下有很多内核、驱动的信息。

### 11.5.1 设备树的信息

以下目录对应设备树的根节点，可以从此进去找到自己定义的节点。

```
[root@100ask:~]# cd /sys/firmware/devicetree/base/
```

节点是目录，属性是文件。

属性值是字符串时，用 cat 命令可以打印出来；属性值是数值时，用 hexdump 命令可以打印出来。

### 11.5.2 platform_device 的信息

以下目录含有注册进内核的所有 platform_device：

```
/sys/devices/platform
```

一个设备对应一个目录，进入某个目录后，如果它有“driver”子目录，就表示这个 platform_device 跟某个 platform_driver 配对了。

比如下面的结果中，平台设备“ff8a0000.i2s”已经跟平台驱动

“rockchip-i2s”配对了：

```
[root@100ask:~/sys/devices/platform/ff8a0000.i2s]# ls driver -ld
lrwxrwxrwx    1 root     root             0 Jan 18 16:28 driver -> ../../../bus/platform/drivers/rockchip-i2s
```

### 11.5.3 platform_driver 的信息

以下目录含有注册进内核的所有 platform_driver：

```
/sys/bus/platform/drivers
```

一个 driver 对应一个目录，进入某个目录后，如果它有配对的设备，可以直接看到。

比如下面的结果中，平台驱动“rockchip-i2s”跟 2 个平台设备“平台设备“ff890000.i2s”、“ff8a0000.i2s”配对了：

```
[root@roc-rk3399-pc:/sys/bus/platform/drivers/rockchip-i2s]# ls
bind      ff890000.i2s  ff8a0000.i2s  uevent    unbind
```

注意：一个平台设备只能配对一个平台驱动，一个平台驱动可以配对多个平台设备。

## 11.6 课后作业

请仿照本节提供的程序(位于 05_led_drv_template_device_tree 目录)，改造你所用的单板的 LED 驱动程序。