

分层分离之总线设备驱动模型

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>
视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>
嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149



概念介绍

总线设备驱动模型匹配规则

衍生出的几大总线

platform总线设备驱动之点灯

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

概念介绍

分层分离思想在**LINUX**中有着不可替代的地位，它包括两个思想：

@分层和分离

分层: 核心层和设备相关层分开

这种思想的优点就是能把很多文件共用的代码抽离集中起来成为一个或者多个核心文件供设备相关层调用，每一层专注于自己的功能

分离: 把硬件相关的代码(固定的,如板子的网卡、中断地址)和驱动(会根据程序作变动,如点哪一个灯)分离开来，即要编写两个文件:**dev.c**和**drv.c**

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

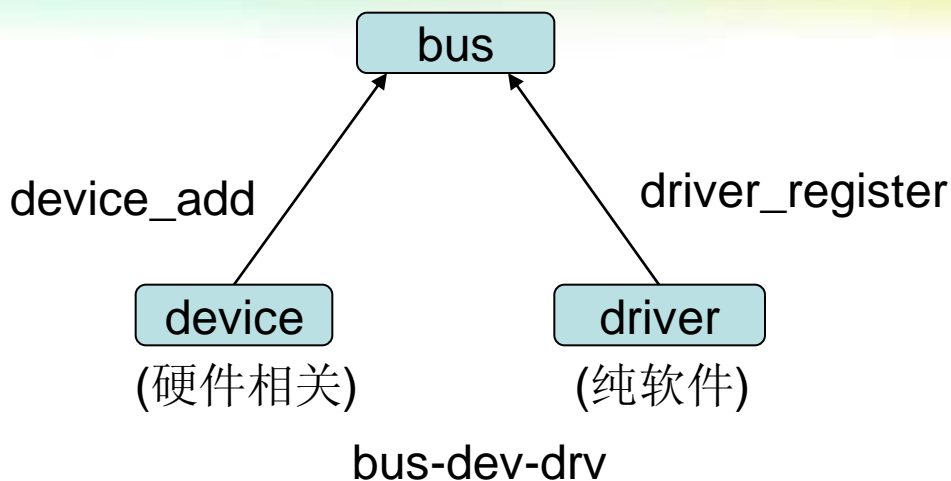
视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

总线设备驱动模型匹配规则

如右图，由
linux_dir\include\linux\dev
ice.h和中可知bus有dev和
drv链表,driver有probe成
员
调用device_add时:

- ①.会将device结构体放入bus的device链表
- ②.从bus的drv链表取出每一个drv,用bus的match函数判断驱动能否支持这个dev
- ③. 若支持就调用驱动的probe



淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>
视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>
嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

总线设备驱动模型匹配规则

继续上页的图:

调用**driver_register**时:

- ①. 会将**driver**结构体放入bus的**driver**链表
- ②. 从bus的dev链表**取出**每一个**dev**, 用bus的**match**函数判断dev能否支持这个drv
- ③. 若支持就调用**驱动的probe**

问: bus的match如何判断dev支持drv呢?

答: 在\linux_dir\drivers\base\bus.c的driver_helper函数:

```
if (strcmp(name, dev->bus_id) == 0)
    return 1;
```

分析代码知match函数是根据**device**的**bus_id**和**driver**的**name**是否一致来匹配的

device_add和driver_register都会两两比较两者的bus_id或name, 如果一样就调用driver的probe

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

衍生出的几大总线

在Linux中由BUS衍生出很多总线如,i2c_bus_type, platform_bus_type等,它们的设备和驱动结构都内嵌了device和driver结构体,物理总线上除了platform_bus_type是虚构的,其他几个总线都是真实存在的

@另外几大总线的匹配规则:

platform_bus_type:

```
return (strncmp(pdev->name, drv->name, BUS_ID_SIZE) == 0)
```

可知platform的匹配规则是platform_device的name和driver的name比较, 如果一样就调用platform_driver的probe

i2c_bus_type:

```
return strcmp(client->driver_name, drv->name) == 0;
```

可知i2c_bus_type是根据i2c_client的name和driver的name是否一样进而调用i2c_driver的probe

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

大概思路:

本例是点亮板上任意一个led, 根据总线设备模型, 这需要两个文件 `led_dev.c` 和 `led_drv.c`, `led_dev.c` 主要负责硬件相关代码, 如提供管脚接LED的GPIO地址, 而 `led_drv.c` 主要负责获得设备文件提供的资源来构造文件操作集合 `fops`

编写步骤:

了解或参考类似的驱动可知需在 `led_dev.c` 中

- @ 分配/设置 `platform_device`

- @ 注册 `platform_device`

还需在 `led_drv.c` 中

- @ 分配/设置 `platform_device`

- @ 注册 `platform_device`



淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

@为platform_device提供资源

```
static struct resource led_resource[] = {  
    [0] = {  
        .start = 0x56000050           // gpfcon寄存器开始地址  
        .end   = 0x56000050 + 8 - 1, // gpfcon寄存器结束地址  
        .flags = IORESOURCE_MEM, // 标志  
    },  
    [1] = {  
        .start = 5,                    // gpfdat 5  
        .end   = 5,  
        .flags = IORESOURCE_IRQ, // 标记  
    }  
};
```

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

@分配/设置platform_device

```
static void led_release(struct device * dev)
{
} // do nothing
struct platform_device led_dev = {
    .name = "myled", // 必须和platform_driver内嵌的driver.name一样
    .id    = -1,
    .num_resources = ARRAY_SIZE(led_resource), // 资源大小
    .resource      = led_resource, // 前面的资源数组
    .dev = {
        .release = led_release, // 必须设置，空的也可以
    },
}
```

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

在led_drv.c里:

@分配/设置platform_driver

```
struct platform_driver led_drv = {  
    .probe          = led_probe, // 匹配后将调用的函数  
    .remove         = led_remove, // 与probe功能相关,做清理工作  
    .driver         = {  
        .name       = "myled", // 必须和platform_device的name一样  
    }  
};
```

咋眼一看,很简单,重点是led_probe和led_remove,可以在probe函数做任何事情,基于此例,我们在probe函数中:

获取设备资源ioremap;注册字符设备,提供读写函数;

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

细说led_probe:

// 根据platform_device的资源进行ioremap

```
res = platform_get_resource(pdev, IORESOURCE_MEM, 0)
```

// 寄存器需要先映射为虚拟地址才能使用

// gpio_con和gpio_dat均为volatile unsigned long类型的指针

```
gpio_con = ioremap(res->start, res->end - res->start + 1)
```

```
gpio_dat = gpio_con + 1
```

```
res = platform_get_resource(pdev, IORESOURCE_IRQ, 0)
```

```
pin = res->start
```

IORESOURCE_MEM和IORESOURCE_IRQ均是resource的flags

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

接着细说led_probe:

// 注册字符设备驱动程序，0表示系统自动分配主设备号

```
major = register_chrdev(0, "myled", &led_fops)
```

```
cls = class_create(THIS_MODULE, "myled")// 创建类"my_led"
```

// 生成设备节点/dev/led

```
class_device_create(cls, NULL, MKDEV(major, 0), NULL, "led")
```

如读者所想,注册字符设备很简单，这个时候应该都会了，-_-若还不会，"那你对不起我 我也对不起你了",关键是设置led_fops

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

设置led_fops

```
struct file_operations led_fops = {  
    .owner = THIS_MODULE,  
    .open  = led_open, // app 打开设备节点将激发  
    .write = led_write, // 对应于app的write  
};
```

主要是led_open:

```
*gpio_con &= ~(0x3<<(pin*2)); // 先清零  
*gpio_con |= (0x1<<(pin*2)); // 配置为输出
```

pin变量是前面设置的标志为IRQ的起始号5

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149

platform总线设备驱动之点灯

细说被应用程序write调用的led_write:

```
int val;  
copy_from_user(&val, buf, count); //用户空间->内核空间  
if (val == 1)           // 用户写的是1  
{  
    *gpio_dat &= ~(1<<pin); // 点灯  
}  
else                    // 用户写的是非0  
{  
    *gpio_dat |= (1<<pin); // 灭灯  
}
```

以上代码均在入口函数实现，但还需在remove和出口函数做相反动作，详情请看视频，读者有任何意见/建议可发1402284892@qq.com，也可在Q群交流

淘宝地址:100ask.taobao.com 版权所有: <http://www.100ask.net/>

视频下载:<http://pan.baidu.com/share/link?uk=2520074993&shareid=480546>

嵌入式交流群: 84174029 28664149, 免费获取Gflash注册码群:28664149