# Linux总线设备驱动模型

## 总线设计

### 总线描述

总线是处理器和设备之间的通道，在设备模型中，所有的设备读通过总线相连，甚至是内部的虚拟“platform”总线。在Linux设备模型中，总线由bus\_type结构表示，定义在<linux/device.h>

struct bus\_type {

const char \*name; /\* 总线名称\*/

struct bus\_attribute \*bus\_attrs; /\* 总线属性\*/

struct device\_attribute \*dev\_attrs; /\* 设备属性\*/

struct driver\_attribute \*drv\_attrs; /\* 驱动属性\*/

int (\*match)(struct device \*dev, struct device\_driver \*drv);//驱动与设备的匹配函数，当一个新设备或者新驱动被添加到这个总线时，该函数被调用。用于判断指定的驱动程序是否能处理指定的设备，若可以，则返回非零。

int (\*uevent)(struct device \*dev, struct kobj\_uevent\_env \*env);

int (\*probe)(struct device \*dev);

int (\*remove)(struct device \*dev);

void (\*shutdown)(struct device \*dev);

int (\*suspend)(struct device \*dev, pm\_message\_t state);

int (\*suspend\_late)(struct device \*dev, pm\_message\_t state);

int (\*resume\_early)(struct device \*dev);

int (\*resume)(struct device \*dev);

struct dev\_pm\_ops \*pm;

struct bus\_type\_private \*p;

}

### 总线注册

bus\_register(struct bus\_type \* bus)

若成功，新的总线将被添加进系统，并可在sysfs的/sys/bus下看到。

### 总线删除

void bus\_unregister(struct bus\_type \*bus)

### 总线属性

struct bus\_attribute {

struct attribute attr;

ssize\_t (\*show)(struct bus\_type \*, char \* buf);

ssize\_t (\*store)(struct bus\_type \*, const char \*

buf, size\_t count);

}

**创建属性**

int bus\_create\_file(struct bus\_type \*bus,struct bus\_attribute \*attr)

**删除属性**

void bus\_remove\_file(struct bus\_type\*bus, struct bus\_attribute \*attr)

## 设备设计

### 设备描述

Linux系统中的每一个设备由一个struct device描述：

struct device {

…… …… …… …… …… ……

struct kobject kobj;

char bus\_id[BUS\_ID\_SIZE]; /\*在总线上唯一标识该设备的字符串 \*/

struct bus\_type \*bus; /\* 设备所在总线 \*/

struct device\_driver \*driver; /\*管理该设备的驱动\*/

void \*driver\_data; /\*该设备驱动使用的私有数据成员 \*

struct klist\_node knode\_class;

struct class \*class;

struct attribute\_group \*\*groups;

void (\*release)(struct device \*dev);

}

### 设备注册

int device\_register(struct device \*dev)

在驱动注册函数中内核会在相应的my\_bus总线上遍历所有的设备，做匹配操作，匹配函数为my\_match(bus.c)匹配方式为设备名和驱动注册的名字是否一致，如果匹配成功，驱动函数为my\_probe被调用。

### 设备注销

void device\_unregister(struct device \*dev)

### 设备属性

struct device\_attribute

{

struct attribute attr;

ssize\_t (\*show)(struct device \*dev, struct device\_attribute

\*attr,char \*buf);

ssize\_t (\*store)(struct device \*dev, struct device\_attribute \*attr,

const char \*buf, size\_t count);

}

**创建属性**

int device\_create\_file(struct device\*device, struct device\_attribute \* entry)

**删除属性**

void device\_remove\_file(struct device \*dev, struct device\_attribute \* attr)

## 驱动设计

### 驱动描述

驱动程序由struct device\_driver描述：

struct device\_driver {

const char \*name; /\*驱动 驱动 程序 的 名字 字( 体 体 现在 在 sysfs 中 中 )\*/

struct bus\_type \*bus; /\*驱动 驱动 程序所在的总线 所在的总线\*/

struct module \*owner;

const char \*mod\_name;

int (\*probe) (struct device \*dev);

int (\*remove) (struct device \*dev);

void (\*shutdown) (struct device \*dev);

int (\*suspend) (struct device \*dev, pm\_message\_t state);

int (\*resume) (struct device \*dev);

struct attribute\_group \*\*groups;

struct dev\_pm\_ops \*pm;

struct driver\_private \*p;

}

### 驱动注册

int driver\_register(struct device\_driver \*drv)

### 驱动注销

void driver\_unregister(struct device\_driver \*drv)

### 驱动属性

struct driver\_attribute {

struct attribute attr;

ssize\_t (\*show)(struct device\_driver \*drv,

char \*buf);

ssize\_t (\*store)(struct device\_driver \*drv,

const char \*buf, size\_t count);

}

**创建属性**

int driver\_create\_file(struct device\_driver \* drv,

struct driver\_attribute \* attr)

**删除属性**

void driver\_remove\_file(struct device\_driver \* drv,

struct driver\_attribute \* attr)