Linux驱动中，probe函数何时被调用  
  
声明：以下主要内容参考自：  
关于struct device\_driver结构中的probe探测函数的调用  
http://blog.chinaunix.net/u2/71164/showart.php?id=1361188  
  
用SourceInsight跟踪：  
从driver\_register看起，此处我的这里是：  
int driver\_register(struct device\_driver \* drv)  
{  
if ((drv->bus->probe && drv->probe) ||  
     (drv->bus->remove && drv->remove) ||  
     (drv->bus->shutdown && drv->shutdown)) {  
   printk(KERN\_WARNING "Driver '%s' needs updating - please use bus\_type methods\n", drv->name);  
}  
klist\_init(&drv->klist\_devices, NULL, NULL);  
return bus\_add\_driver(drv);  
}  
  
klist\_init不相关，不用管他，具体再去看bus\_add\_driver：  
  
int bus\_add\_driver(struct device\_driver \*drv)  
{  
1.先kobject\_set\_name(&drv->kobj, "%s", drv->name);  
2.再kobject\_register(&drv->kobj)  
3.然后调用了：driver\_attach(drv)  
}  
  
  
int driver\_attach(struct device\_driver \* drv)  
{  
return bus\_for\_each\_dev(drv->bus, NULL, drv, \_\_driver\_attach);  
}  
  
真正起作用的是\_\_driver\_attach：  
  
static int \_\_driver\_attach(struct device \* dev, void \* data)  
{  
。。。  
if (!dev->driver)  
   driver\_probe\_device(drv, dev);  
。。。  
}  
  
  
int driver\_probe\_device(struct device\_driver \* drv, struct device \* dev)  
{  
。。。  
//1.先是判断bus是否match：  
if (drv->bus->match && !drv->bus->match(dev, drv))  
   goto done;  
//2.再具体执行probe：  
ret = really\_probe(dev, drv);  
。。。  
}  
  
really\_probe才是我们要找的函数：  
static int really\_probe(struct device \*dev, struct device\_driver \*drv)  
{  
。。。  
//1.先是调用的驱动所属总线的probe函数：  
if (dev->bus->probe) {  
   ret = dev->bus->probe(dev);  
   if (ret)  
    goto probe\_failed;  
  
} else if (drv->probe) {  
//2.再调用你的驱动中的probe函数：  
   ret = drv->probe(dev);  
   if (ret)  
    goto probe\_failed;  
}  
。。。  
}  
  
其中，drv->probe(dev)，才是真正调用你的驱动实现的具体的probe函数。  
也就是对应此文标题所问的，probe函数此时被调用。