[Linux USB 3.0驱动分析（十）——Gadget UAC2驱动分析](https://www.cnblogs.com/wen123456/p/14373698.html)

本文分析的是linux-5.4.3

**一.Gadget Audio设备驱动分析**

drivers/usb/gadget/legacy/audio.c

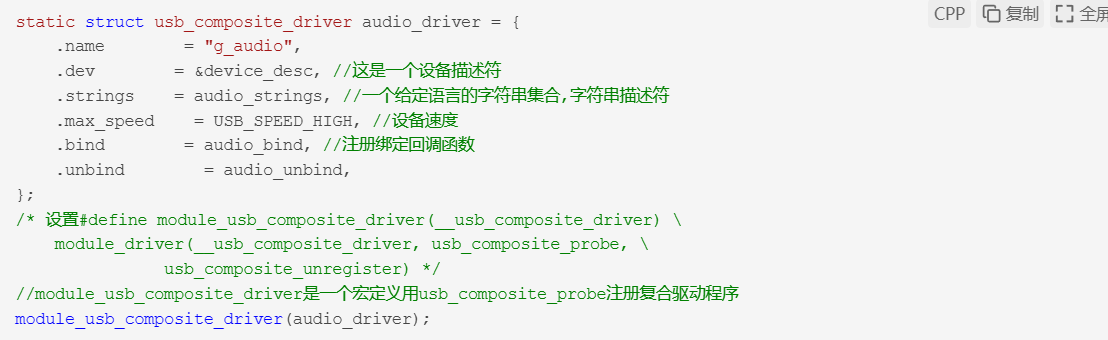
因为项目的问题， 了解usb音频设备的工作原理，为啥它能让PC识别成“speak”或者“mic”，以及你能够播放录音。

主要涉及下面两个层次：

Gadget功能驱动层:  最主要的结构是struct usb\_composite\_driver，这个结构在这层定义，并且实现结构中的各个函数。  
USB设备层:  最主要的数据结构是struct usb\_composite\_dev与usb\_gadget\_driver。前一个代表一个USB复合设备，而后一个是Gadget驱动，与UDC层交互。

这边主要是usb结构中的设备和配置相关

首先我们来看初始化流程，主要是注册audio\_driver这个复合设备驱动



我们再来看看usb\_composite\_probe里面具体做了什么，主要是赋值了composite\_driver\_template，这个是所有复合设备共有的



后面继续调用usb\_gadget\_probe\_driver,进行注册。





从上面usb\_gadget\_probe\_driver()函数的代码片段可知，它在遍历udc\_list链表，找到一个udc（usb设备控制器驱动描述结构体）没有对应驱动的实例。



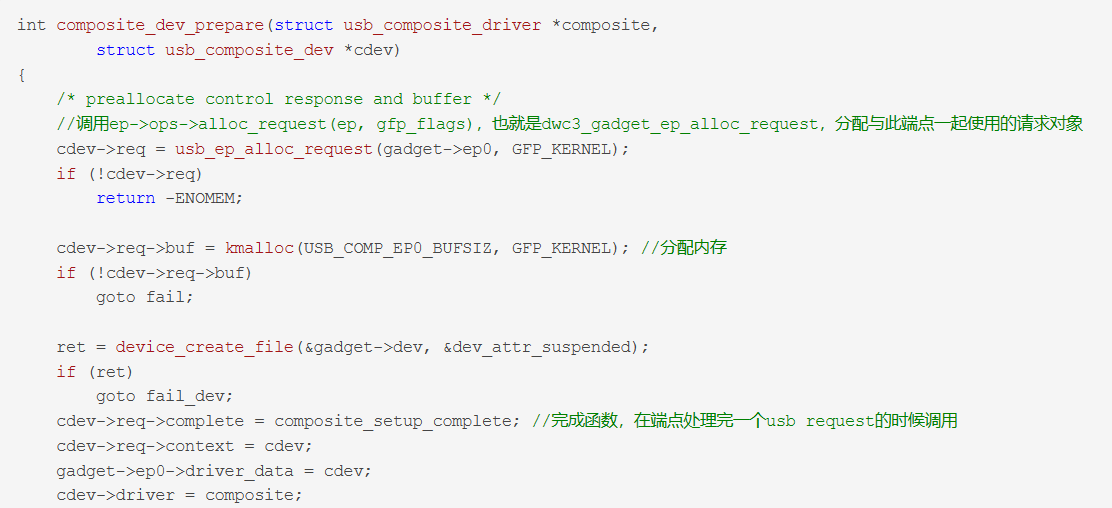
主要是给udc相关的驱动数据结构赋值。既然有udc队列的遍历取出，必然有udc队列的添加，每注册一个udc驱动（usb\_add\_gadget\_udc()）就会把udc结构体添加到udc队列的尾部，上一篇udc相关分析有用到。

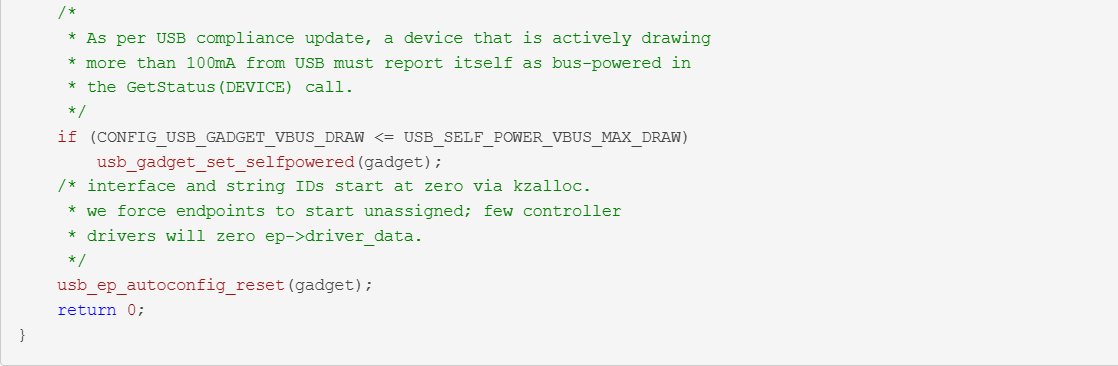
我们再来分析composite\_driver\_template的composite\_bind函数,这里对复合设备做一些前期准备，然后调用复合设备的bind函数



**对复合设备的前期准备composite\_dev\_prepare**

主要是初始化端点0， ep0是设备控制传输用到的端点，尤其在设备枚举、属性配置上起关键作用，所以我们要首先初始化

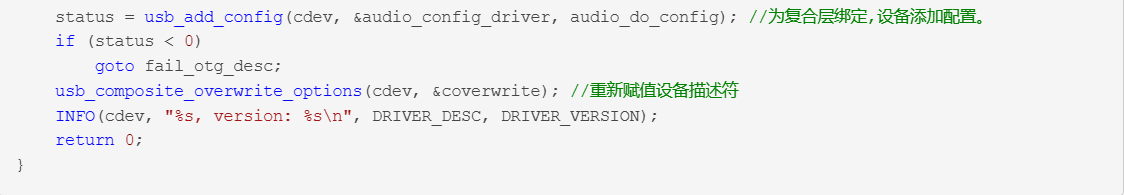




**复合设备的bind函数，这里是audio\_bind**

这里设备描述符相关在这里被初始化



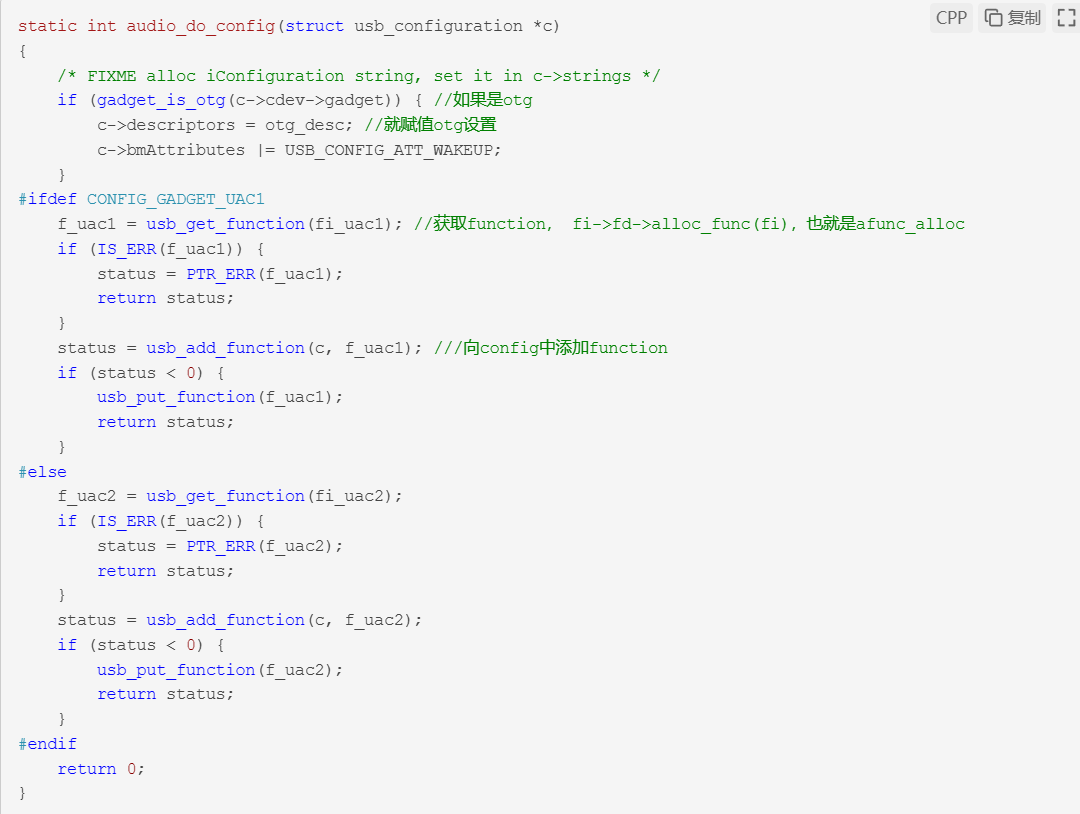


在复合层绑定，进行配置的初始化，配置完成之后，这个ugadget就具有UAC的功能了。





我们来看看关键的do\_config函数



通过usb\_add\_dunction完成config和function的绑定

