关于usb的一点思考(2)

提到usb，不得不说descriptor。

Struct usb\_descriptor\_header {

\_\_u8 bLength;

\_\_u8 bDescriptorType;

}\_\_attribute\_\_((packed));

**Device descriptor**

设备描述符描述了设备的通用信息，一个usb设备只有一个device descriptor。

Device descritptor的bNumConfigurations描述了当前速率的configuration个数，usb的default control pipe的最大包大小在device descriptor的bMaxPacketSize0中体现。

Struct usb\_device\_descriptor {

U8 bLength;/\*device descriptor大小\*/

U8 bDescriptorType;/\*device descriptor type\*/

U16 bcdUSB;/\*usb specification release num\*/

U8 bDeviceClass;/\*\*/

U8 bNumConfigurations;/\*\*/

};

Struct usb\_host\_endpoint {

Struct usb\_endpoint\_descriptor desc;

};

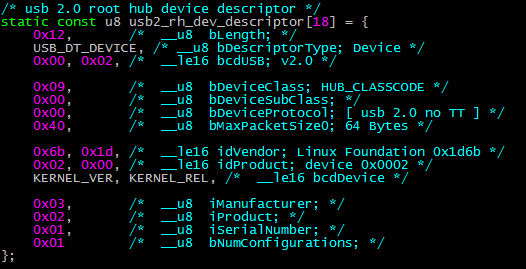
Struct usb\_device {

Struct usb\_host\_endpoint ep0;

Struct usb\_device\_descriptor descriptor;/\*在usb2.0 spec有定义，standard usb descriptor definition\*/

};

Struct usb\_device 的descriptor成员通过usb\_get\_device\_descriptor获取，比如usb2.0的device descriptor如下所示:



**Configuration descriptor**

Struct usb\_config\_descriptor {

\_\_u8 bLength;

\_\_u8 bDescriptorType;

\_\_le16 wTotalLength;/\*configuration descriptor+interface descriptor +endpoint descriptor\*/

\_\_u8 bNumInterfaces;

\_\_u8 bConfigurationValue;/\*value to use as an argument to the setconfiguration() request to select this configuration\*/

};

Struct usb\_host\_config {

Struct usb\_config\_descriptor desc;

};

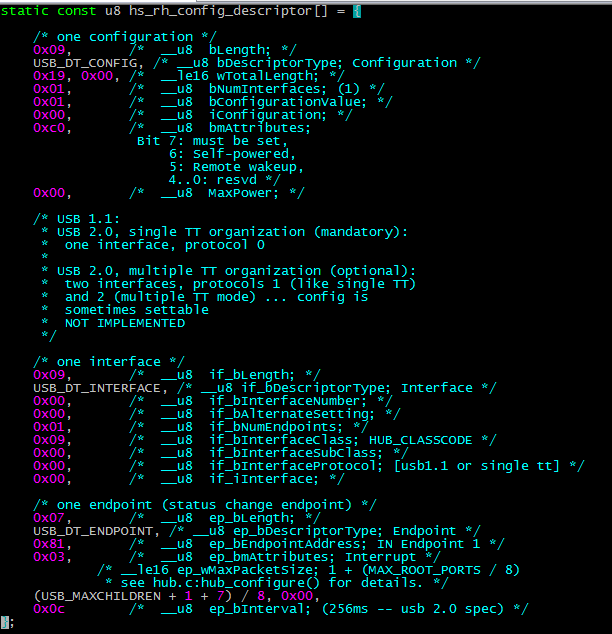
Struct usb\_device {

Struct usb\_host\_config \*config;

Char \*\*rawdescriptors;

};

Roothub的configuration如下所示:



**Interface descriptor**

Struct usb\_interface\_descriptor {

U8 bLength;

U8 bDescriptorType;

U8 bInterfaceNumber;

U8 bAlternateSetting;

};

Struct usb\_host\_interface {

Struct usb\_interface\_descriptor desc;

Int extralen;

Unsigned char \*extra;

Struct usb\_host\_endpoint \*endpoint;

};

Struct usb\_interface\_cache {

Unsigned num\_altsetting;

struct usb\_host\_interface altsetting[0];

};

Struct usb\_host\_config {

Struct usb\_config\_descriptor desc;

Struct usb\_interface\_cache \*intf\_cache[32];

};

Struct\_size(intfc,altsetting,j)

\_\_ab\_c\_size(j,sizeof(\*(intfc)->altsetting)+\_\_must\_be\_array((intfc)->altsetting),sizeof(\*(intfc)))

Usb\_get\_configuration->usb\_parse\_configuration

Usb 的struct usb\_device\_configuration的bNumConfigurations记录着有多个configuration descriptor，在usb\_get\_configuraion中遍历所有的configuration descriptor。

在遍历过程中Usb 通过usb\_get\_descriptor type为USB\_DT\_CONFIG获取configuration descriptor，configuration descriptor包含configuration descriptor +interface descriptor +endpoint descriptor ；由于事先不知道configuration descriptor的大小，所以usb\_get\_configuration首先只是获取configuration descriptor，然后解析configuration descriptor的wTotalLength，然后再获取整个configuration descriptor。

获取整个configuration descriptor后，将其和大小传入usb\_parse\_configuration进行解析。

Usb\_parse\_configuration(dev,cfgidx,config,buffer,size)

{

1. Memcpy(&config->desc,buffer,USB\_DT\_CONFIG\_SIZE);
2. Buffer0 =buffer;Buffer += config->desc.length;size-=config->desc.bLength;
3. 遍历所有struct usb\_descriptor\_header，如果struct usb\_descriptor\_header的bDescriptorType是USB\_DT\_INTERFACE，
4. Inum =struct usb\_interface\_descriptor的bInterfaceNumber
5. 如果无重复的inum，inum[n] =inum,nalts[n] =1,++n

如果有重复的inum，++nalts[i];

1. 遍历interface，根据nalts[i]的个数，即nalts[i]\*sizeof(struct usb\_host\_interface)+sizeof(struct usb\_interface\_cache)分配struct usb\_interface\_cache，赋值config->intf\_cache[i]。
2. 从buffer开始，寻找第一个USB\_DT\_INTERFACE
3. 遍历所有的interface,usb\_parse\_interface()

}