lopment Exploration 89- YIE002USB Development Board (12 Linux Programmin

Posted on 2021-05-13 12:05:05 Views: 684 Collection 4 Likes 1

nn: UEFI Development Article Tags: uefi USB usb hid Linux hid Low-level application development

lopment This column includes this content 503 Subscribe 104 arti

Luo Bing https://blog.csdn.net/luobing4365)

ıpment **Board** Linux **Programming**

gramming, it has been discussed at the beginning of YIE002USB:

uter is a USB HID device made using YIE002, which is an embedded development;

ıter includes Windows system, UEFI system and Linux system.

uded in the host computer software , mainly because I seldom develop on Apple systems, and it can also be regarded as equivalent to the Linux syst em).

ıpleted the software writing of the host computer for Windows system and UEFI system. Now it is time to enter the USB HID host computer software d

:, the website "The Linux Kernel " has a very detailed description, which is worth reading carefully. The URL is: rg/doc/html/latest/hid/index.html.

to access is a USB HID device. In Linux systems, the commonly used USB open source library is libusb. This is a cross-platform USB device access uage.

en source library on Github, I read through its documentation. In theory, it can fully implement the functions we need. However, the documentation als HID devices, you can use hidapi.

ource C language library for operating HID devices, suitable for Windows, Linux and Mac OSX platforms. It is for HID devices and is not suitable for ot

ository is: https://github.com/libusb/hidapi, the main directories are as follows:

```
文件（所有平台共用一份头文件）hidapi.h
ıux系统实现源码文件hid.c，使用libusb库实现的方式
ıx系统实现源码文件hid.c，使用内核接口（hidraw）实现方式
.ndows系统实现源码文件hid.c
SX 系统实现源码文件hid.c
试代码 test.c
```

o develop now is the USB HID access program under Linux. From its code structure, we can see that there are two ways to use it, namely hidraw an

encapsulated, there is not much difference between the two development methods. Most of my work is to consider how to write the Makefile file for cc

ses the hidraw method to develop access code under Linux.

of HidAPI

nent process, I used a virtual machine (Ubuntu 16.04) and an actual machine (Ubuntu 18.04) to conduct experiments. The former was used to dev
ne latter was used to develop the libusb method code.

ent environments because I found that the code developed in libusb mode has various problems when running on a virtual machine. The specific reas
leveloped in both modes runs well on the actual machine.

sary components

i library to your local computer:

```
n-virtual-machine:~$ git clone https://github.com/libusb/hidapi.git
```

relatively small and should be downloadable. If you still have problems, it is recommended to find the corresponding library on gitee and git it local

mponents:

```
n-virtual-machine:~$ sudo apt-get install libudev-dev libusb-1.0-0-dev
```

nponents, please refer to the requirements in the hidapi documentation. During the development process, there is no need to use the GUI, so libfox-1

for compilation are not installed, you also need to install them:

```
n-virtual-machine:~$ sudo apt-get install  build-essential pkg-config
```

ains the tools required for compilation, including gcc, make, etc. pkg-config is a command under Linux, which is used to obtain all compilation-related
s needed during the compilation process.

nterface

::

: 初始化函数，无参数。
: 退出，实际上是销毁结构体等，在完成所有工作后必须调用，否则会造成内存泄漏。

ate(): 枚举设备，返回的是hid_device_info链表。一般使用hid_enumerate(0, 0)枚举所有设备。枚举一般用于获取设备ID或者设备路径。如果提前知道这些信息
numeration(): 释放枚举所用到的链表。

closing:

: 打开指定 VID 和 PID 设备，返回设备结构体指针，如 hid_device *handle; handle = hid_open(0x8765, 0x4321, NULL)。设备读写和关闭函数时，均
ath(): 根据设备路径打开设备，设备路径由hid_enumerate获取。如Linux下的 /dev/hidraw1。
): 关闭设备。

ding and Receiving:

ature_report(): 获取 Feature report，也即采用Feature报告进行通信。
eature_report(): 发送 Feature report。

tions:

: 读取数据。
): 写数据。

out that many articles on the Internet regard hid_read() and hid_write() as reading input reports and sending output reports. This understanding is pr
ts, it can be seen that hid_read() and hid_write() do not use the communication method of input report and output report . At least this understandi

tal results, these two functions are similar to the ReadFile() and WriteFile() methods of the Windows system, while the lower computer corresponds t
ıod (see the 85th blog). That is, when there are multiple endpoints (including the required endpoint 0 and other endpoints), endpoint communication i
the Input report and Output report communication methods will be used.

code of hidapi (hid.c in the inux folder), it is clearly stated that the hidraw method does not implement the acquisition of input report.

esign it depends on the firmware of the lower computer.

sing the hidraw method of hidapi

the above background knowledge, it is relatively easy to implement the required code.

st program, located in the hidtest folder test.c. To compile the Linux version, you can use the Makefile file in the linux folder to compile:

```
n-virtual-machine:~/hidapi/linux$ make -f Makefile-manual
```

hidtest-hidraw executable file will be generated. We need to implement the required functions based on test.c.

levelopment directory

ıent development, you need to create a development folder yourself and include the required source files and header files.

hidraw and copy several files of hidapi here:

```
c
api.h
st.c
```

ıere is a file Makefile-manual for compilation. Since the file directory has changed, it cannot be used directly. You can create a new Makefile in hidraw

```
st-hidraw
= gcc
= -Wall -g -fpic

= -Wall -g

= hid.o test.o
= $(COBJS)
= `pkg-config libudev --libs` -lrt
= $(LIBS_UDEV)
= `pkg-config libusb-1.0 --cflags`


Test Program
draw: $(COBJS)
$(LDFLAGS) $^ $(LIBS_UDEV) -o $@


%.o: %.c
$(CFLAGS) -c $(INCLUDES) $< -o $@


$(OBJS) hidtest-hidraw $(COBJS)

ean libs
```

the hidraw folder and enter make to compile:

```
n-virtual-machine:~/luotest/hidapi$ make
g -fpic -c `pkg-config libusb-1.0 --cflags` hid.c -o hid.o
```

```
g -fpic -c `pkg-config libusb-1.0 --cflags` test.c -o test.o
g hid.o test.o `pkg-config libudev --libs` -lrt -o hidtest-hidraw
```

he executable file hidtest-hidraw is generated.

ient access to USB HID devices based on the original code in test.c.

aw code development

log, we know that there are three ways for the host computer to communicate with the USB HID device. Currently, the hidapi code provides two meth
_write(), but does not provide the communication method of Input report&Output report.
aw, there is no corresponding interface.

can't understand why there is no Input report & Output report interface in hidraw. If anyone knows, please leave a message in the comment section c

t communication method

nunication uses two functions provided in hid.c: hid_send_feature_report() and hid_get_feature_report(). The code is as follows:

```
(yie_buf,0,sizeof(yie_buf));
f[0] = 0x00;
f[1] = 0xA0;
f[2] = 0x0a;
f[3] = 0x0b;
f[4] = 0x0c;
hid_send_feature_report(handle, yie_buf, 17);
s < 0) {
intf("Unable to send a feature report.\n");


(yie_buf,0,sizeof(yie_buf));

d a Feature Report from the device
[0] = 0x2;
hid_get_feature_report(handle, yie_buf, sizeof(yie_buf));
s < 0) {
intf("Unable to get a feature report.\n");
intf("%ls", hid_error(handle));


 Print out the returned buffer.
intf("Feature Report\n   ");
intf("report number:%d\n   ",yie_buf[0]);
r (i = 1; i < res; i++)
  printf("%02x ", yie_buf[i]);
intf("\n");
```

d_write() communication method

ead() and hid_write() functions, which can implement functions similar to Windows' ReadFile() and WriteFile() (only for USB HID devices). The implen

```
(yie_buf,0,sizeof(yie_buf));
f[0] = 0x00;
f[1] = 0xA0;
f[2] = 0x0a;
f[3] = 0x0b;
f[4] = 0x0c;

hid_write(handle, yie_buf, 17);
s < 0) {
intf("Unable to write()\n");
intf("Error: %ls\n", hid_error(handle));


(yie_buf,0,sizeof(yie_buf));
hid_read(handle, yie_buf, sizeof(yie_buf));
```

```
s < 0) {
intf("Unable to Read data.\n");
intf("%ls", hid_error(handle));


 Print out the returned buffer.
intf("Read data\n   ");
r (i = 0; i < res; i++)
  printf("%02x ", yie_buf[i]);
intf("\n");
```

ompile using make. Plug in the self-made USB HID device and run the test code. The test results are as follows:

```
n-virtual-machine:~/luotest/hidapi$ sudo ./hidtest-hidraw

er String: Robin
ring: Robin's UEFI Explorer
ber String: (77) My123
read indexed string 1
ring 1:
port
number:0
b 0c 00 00 00 00 00 00 00 00 00 00 00

b 0c 00 00 00 00 00 00 00 00 00 00 00
```

second byte of the returned data. From the content in the previous blog, we can know that two of the three communication methods have been imple

lete three communication methods, the next article will use the libusb library to rebuild the code.

://gitee.com/luobing4365/uefi-explorer
ed at: /89 hidraw
