

# UEFI Development Exploration 90- YIE002USB Development Board (13 Linux Programming)

原创 luobing4365 Posted on 2021-05-13 12:12:14 Read 887 collect Likes copyright

Category Column: UEFI Development Article Tags: uefi usb hid linux hid Low-level application development bios



UEFI Development This column includes this content

503 Subscribe

104 articles

Subscribe to

our column

(Please keep it-> Author: Luo Bing <https://blog.csdn.net/luobing4365>)

## YIE002USB Development Board Linux Programming

### 1 Create a development directory

#### 2. Code Writing

##### 2.1 Add Output report processing code

##### 2.2 Input report & Output report communication method

In the previous article, we used the hidraw method of hidapi to implement a test program for the communication between the Linux host computer and the USB HID device. However, we only implemented two of the three communication methods. This article will use the libusb method of hidapi to implement all three communication methods.

## 1 Create a development directory

As in the previous article, first create a directory for development. Create a new folder hidlibusb and copy the following files into it:

```
1 libusb/hid.c
2 hidapi/hidapi.h
3 hidtest/test.c
```

Write a Makefile for the current code with the following content:

```
1 all: hidtest-libusb libs
2
3 libs: libhidapi-libusb.so
4 CC      ?= gcc
5 CFLAGS  ?= -Wall -g -fpic
6 LDFLAGS ?= -Wall -g
7
8 COBJS_LIBUSB = hid.o
9 COBJS = $(COBJS_LIBUSB) test.o
10 OBJS   = $(COBJS)
11 LIBS_USB = `pkg-config libusb-1.0 --libs` -lrt -lpthread
12 LIBS     = $(LIBS_USB)
13 INCLUDES ?= `pkg-config libusb-1.0 --cflags`
14
15 # Console Test Program
16 hidtest-libusb: $(COBJS)
17     $(CC) $(LDFLAGS) $^ $(LIBS_USB) -o $@
18 # Shared Libs
19 libhidapi-libusb.so: $(COBJS_LIBUSB)
20     $(CC) $(LDFLAGS) $(LIBS_USB) -shared -fpic -Wl,-soname,$@.0 $^ -o $@
21 # Objects
22 $(COBJS): %.o: %.c
23     $(CC) $(CFLAGS) -c $(INCLUDES) $< -o $@
24
25 clean:
26     rm -f $(OBJS) hidtest-libusb libhidapi-libusb.so hidtest.o
```

```

28 | .PHONY: clean libs

```

## 2. Code Writing

After creating the development directory, you can modify the code.

The three communication methods of USB HID, `hid_read()` & `hid_write()` communication method and feature report communication method, have been implemented in the previous article. Since `hid.c` under `libusb` now provides a consistent **function** interface, the codes of these two methods do not need to be modified at all. We only need to add code to support input report and output report.

### 2.1 Add Output report processing code

What is strange is that in the source file `hid.c` of `libusb`, the function `hid_get_input_report()` for processing Input report is provided, but there is no function for processing Output report.

The code for processing the Output report is also very simple. Just modify a number in the `hid_send_feature_report()` function. The code is as follows:

```

1 | int HID_API_EXPORT HID_API_CALL hid_set_output_report(hid_device *dev, const unsigned char *data, size_t length)
2 | {
3 |     int res = -1;
4 |     int skipped_report_id = 0;
5 |     int report_number = data[0];
6 |
7 |     if (report_number == 0x00) {
8 |         data++;
9 |         length--;
10 |         skipped_report_id = 1;
11 |     }
12 |
13 |     res = libusb_control_transfer(dev->device_handle,
14 |         LIBUSB_REQUEST_TYPE_CLASS|LIBUSB_RECIPIENT_INTERFACE|LIBUSB_ENDPOINT_OUT,
15 |         0x09/*HID set_report*/,
16 |         (2/*HID Output*/ << 8) | report_number,
17 |         dev->interface,
18 |         (unsigned char *)data, length,
19 |         1000/*timeout millis*/);
20 |
21 |     if (res < 0)
22 |         return -1;
23 |
24 |     /* Account for the report ID */
25 |     if (skipped_report_id)
26 |         length++;
27 |
28 |     return length;
29 | }

```

### 2.2 Input report & Output report communication method

The other two communication methods have been implemented in the previous article, so there is no need to modify any code. Now we need to add the sending of Output report and receiving of Input report. The implementation code is as follows:

```

1 |     memset(yie_buf,0,sizeof(yie_buf));
2 |     yie_buf[0] = 0x00;
3 |     yie_buf[1] = 0xA0;
4 |     yie_buf[2] = 0x0a;
5 |     yie_buf[3] = 0x0b;
6 |     yie_buf[4] = 0x0c;
7 |
8 |     res = hid_set_output_report(handle, yie_buf, 17);
9 |     if (res < 0) {
10 |         printf("Unable to send a output report.\n");
11 |     }
12 |     memset(yie_buf,0,sizeof(yie_buf));
13 |     res = hid_get_input_report(handle, yie_buf, sizeof(yie_buf));
14 |     if (res < 0) {
15 |         printf("Unable to get a input report.\n");

```

```

16     printf("%ls", hid_error(handle));
17 }
18 else {
19     // Print out the returned buffer.
20     printf("Input Report\n ");
21     printf("report number:%d\n ", yie_buf[0]);
22     for (i = 1; i < res; i++)
23         printf("%02x ", yie_buf[i]);
24     printf("\n");
25 }

```

Start the command line and use the make command to compile and obtain the executable file hidtest-libusb.

Insert the homemade USB HID device and run hidtest-libusb. The output information is as follows:

```

1  robin@NUC6CAYHC:~/luotest/hidapi$ sudo ./ hidtest-libusb
2  ..... //略
3  Manufacturer String: Robin
4  Product String: Robin's UEFI Explorer
5  Serial Number String: (77) My123
6  Indexed String 1: Robin
7  Feature Report
8      report number:0
9      a0 03 0b 0c 00 00 00 00 00 00 00 00 00 00 00 00
10 Input Report
11     report number:0
12     a0 02 0b 0c 00 00 00 00 00 00 00 00 00 00 00 00
13 Read data,Length=16
14     a0 01 0b 0c 00 00 00 00 00 00 00 00 00 00 00 00

```

Comparing with the previous blog on making USB HID devices, it can be seen that the three communication modes have been implemented.

So far, all the codes we planned in the introduction of the YIE002 development board have been completed, from the production of the USB HID device of the development board to the [Windows software](#) , UEFI software and Linux software of the host computer.

The study of UEFI related to the YIE002 development board ends here. For other embedded programming of this development board, please move to my "[Embedded Development](#) " column, which I will update from time to time.

Gitee address: <https://gitee.com/luobing4365/uefi-explorer>

Project code is located under: /90 hidlibusb

about  
Us

Careers

Business  
Cooperation

Seeking  
coverage

400-660-  
0108

kefu@csdn.net

Online  
Customer  
Service

Working hours  
8:30-22:00

Public Security Registration Number 11010502030143 Beijing ICP No. 19004658 Beijing Internet Publishing House [2020] No. 1039-165

Commercial website registration information Beijing Internet Illegal and Harmful Information Reporting Center Parental Control

Online 110 Alarm Service China Internet Reporting Center Chrome Store Download Account Management Specifications

Copyright and Disclaimer Copyright Complaints Publication License Business license

©1999-2025 Beijing Innovation Lezhi Network Technology Co., Ltd.