:FI Development Exploration 93–Using the YIE002 Random Nun Generator under UEFI

luobing4365 Posted on 2021-08-16 16:50:38 Read 458 times collect Likes copy
Category Column: UEFI Development Article Tags: uefi bios Low-level application development USB UEFI Programming Practic

UEFI Development This column includes this content 503 Subscribe 104 articles Subscribe
our colum

›e keep it-> Author: Luo Bing https://blog.csdn.net/luobing4365)

g the YIE002 random number generator under UEFI

:FI development exploration 72, the development board of YIE002 was planned. The main goal at that time was to use 02 as a two-way communication device for USB HID and implement the host computer access program under Wind ‹ and UEFI.

y, these goals have been achieved, especially the Windows test program UsbHID, which has now become my dedic ool.

ɘ was one item in the development plan that has not yet been completed, which is to enable the YIE002 development ɟ to provide the function of generating random numbers so that the generated random numbers can be obtained throug ISB channel in various systems, including the UEFI system.

nspiration for this goal comes from ChaosKey, developed by foreign engineers. I came across it by chance when I was :hing for information online while studying UEFI development. It is a small device (the size of a USB flash drive) that rates random numbers. The author spent a lot of time to make it, which I find very interesting. Therefore, when designii 02, a mechanism for generating random numbers was reserved.

nt some time on the weekend and finally made this random number generator . For the specific process, please see ' blog: YIE002 Development Exploration 10-Random Number Generator.

use this small device under UEFI today.

:ode Writing

:code writing process is actually very simple. We borrowed the HelloHID project from UEFI Development Exploration 88 :ly modified the functions to achieve the required functions.

ge HelloHID to RngUEFI, including the INF file and source file name. Then in RobinPkg.dsc, add compilation support f
ngUEFI project to complete the framework adjustment.

the YIE002 random number generator provides three USB HID communication capabilities, you can use any of them
nunicate with it. Therefore, I directly wrote the code using the Feature Report communication method.

the function of getting random numbers as follows:

```
//Name: GetRNG_YIE002
//Input: index
//Output:  RNG
UINT16 GetRNG_YIE002(IN INT16 index)
{
  EFI_STATUS Status;
  UINT8   ReportId;
  UINT8 myBuffer[16];
  UINT16 random_value;

  gBS->SetMem(myBuffer,16,0xA0);
  ReportId = 0;
  Status = UsbSetReportRequest(
    gUsbIO[index],
    0,               //interface,
    ReportId,
    HID_FEATURE_REPORT,
    16,
    myBuffer
  );
  if(EFI_ERROR(Status))
  {
    Print(L"UsbSetReportRequest Error!\n");
    return FALSE;
  }
  gBS->SetMem(myBuffer,16,0x00);
  Status = UsbGetReportRequest(
    gUsbIO[index],
    0,               //interface,
    ReportId,
    HID_FEATURE_REPORT,
    16,
    myBuffer
  );
  if(EFI_ERROR(Status))
  {
    Print(L"UsbGetReportRequest Error!\n");
    return FALSE;
  }
  random_value = myBuffer[1];
  random_value = (random_value<<8);
  random_value += myBuffer[0];

  return random_value;
}
```

the statement to access the device in the main program:

```
//测试YIE002制作的随机数生成器
  if(retVal)
  {
    for(i=0; i<20; i++)
    {
      wRandomValue = GetRNG_YIE002(myDevice);
      Print(L"------ %02d Get Random number from YIE002:%04x\n",i,wRandomValue);
      Delayms(1000);
    }
  }
```

code for obtaining random numbers is completed.

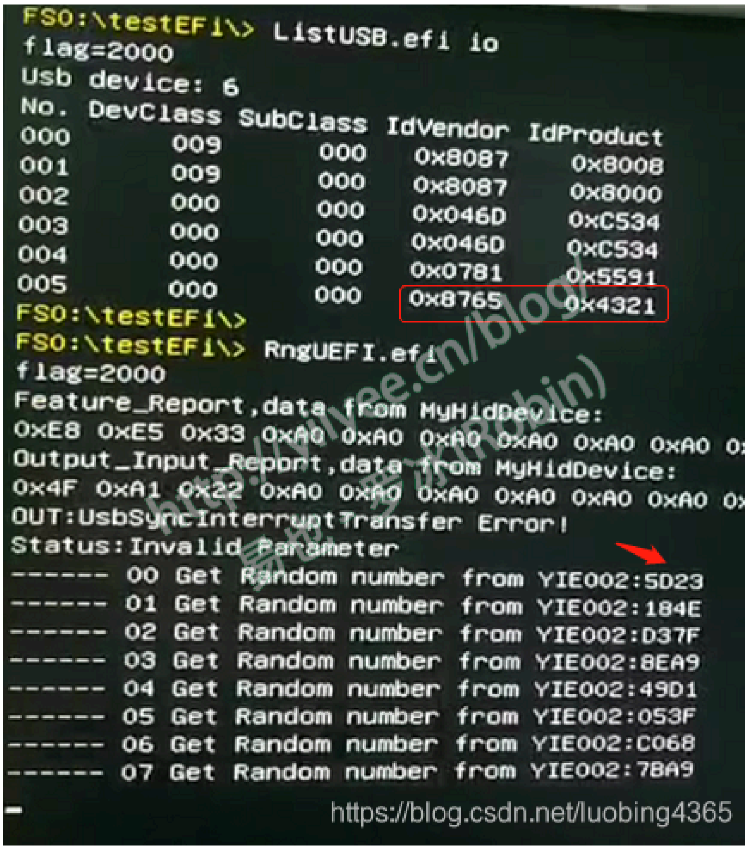
actual code is at the end of the article as usual, you can read it yourself if you are interested.

sting

pile the code using the following command:

```
C:\vUDK2018\edk2>build -p RobinPkg\RobinPkg.dsc -m RobinPkg\Applications\RngUEFI\RngUEFI.inf -a X
```

the generated file RngUEFI.efi to the UEFI boot USB disk for testing, connect YIE002 to the testing machine, and star Shell. The test results are shown in Figure 1.



```
FS0:\testEFI\> ListUSB.efi io
flag=2000
Usb device: 6
No. DevClass SubClass IdVendor IdProduct
000       009      000    0x8087    0x8008
001       009      000    0x8087    0x8000
002       000      000    0x046D    0xC534
003       000      000    0x046D    0xC534
004       000      000    0x0781    0x5591
005       000      000    0x8765    0x4321
FS0:\testEFI\>
FS0:\testEFI\> RngUEFI.efi
flag=2000
Feature_Report,data from MyHidDevice:
0xE8 0xE5 0x33 0xA0 0xA0 0xA0 0xA0 0xA0 0xA0 0
Output_Input_Report,data from MyHidDevice:
0x4F 0xA1 0x22 0xA0 0xA0 0xA0 0xA0 0xA0 0xA0 0x
OUT:UsbSyncInterruptTransfer Error!
Status:Invalid Parameter
------ 00 Get Random number from YIE002:5D23
------ 01 Get Random number from YIE002:184E
------ 02 Get Random number from YIE002:D37F
------ 03 Get Random number from YIE002:8EA9
------ 04 Get Random number from YIE002:49D1
------ 05 Get Random number from YIE002:053F
------ 06 Get Random number from YIE002:C068
------ 07 Get Random number from YIE002:7BA9
_
```

e 1 Testing the random number generator of YIE002

g the test, plug the YIE002 into the USB port of the test machine, and use the previous enumeration tool ListUsb.efi in Shell to see whether the device is plugged in (the PID and VID we use are 0x4321 and 0x8765 respectively).

n running RngUEFI.efi, random numbers will continue to be obtained from YIE002. As can be seen in Figure 1, all obta es are 16-bit random values.

re testing the random number acquisition, three USB HID communication methods were run. The host computer acces od through endpoint 1 (that is, the access method of reading and writing files, YIE002 uses endpoint 1 to read and writ spond to this method) is not supported on this test machine, so an error is returned.

is point, the programming of the YIE002 development board that was set at that time was almost completed.

e address: https://gitee.com/luobing4365/uefi-explorer
ct code is located in: /FF RobinPkg/RobinPkg/Applications/RngUEFI