UEFI Development Exploration 69- YIE001PCIe Development Board (05 UEFI Driver)

原创 luobing4365 Posted on 2021-01-16 21:45:44 Views: 828 Collection 1 Likes copyright

Category Column: UEFI Development Article Tags: UEFI BIOS Low-level application development UEFI Drivers BlankDrv

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

In the previous article, after the driver BlankDrv completed the core EFI_DRIVER_BINDING_PROTOCOL and obtained the driver name Protocol, the entire UEFI driver framework still has two tasks to be completed:

- Install the Protocol provided by the UEFI driver;
- After the driver is loaded, it resides in the memory. The UEFI driver uninstall function should be provided to support the function of removing the driver from the system.

1 Complete the driver framework

To install the Protocol, you can use the InstallMultipleProtocolInterfaces() function . Since there are three protocols to be installed in BlankDrv, the library function EfiLibInstallDriverBindingComponentName2() is used to install all the protocols, as shown in Example 1.

[Example 1] Install the Protocol provided by BlankDrv

```
EFI_STATUS EFIAPI UefiMain(
  IN EFI_HANDLE ImageHandle,
  IN EFI_SYSTEM_TABLE *SystemTable
  )
{   EFI_STATUS Status;   ….. //Three Protocols required for installation   Status = EfiLibInstallDriverBindingComponentName2 (             ImageHandle,
SystemTable,            &gBlankDrvDriverBinding,            ImageHandle,            &gBlankDrvComponentName,            &gBlankDrvComponentName2
);   return Status; }
```

The driver's unload function is called when the UEFI Shell command unload is used. The entry point of this function is added in the INF file of the UEFI project and is given by the UNLOAD_IMAGE field in the [Defines] Section. In BlankDrv.inf of the sample project BlankDrv, it is defined as follows:

```
[Defines]
  ...
ENTRY_POINT = UefiMain //Entry point of the driver
UNLOAD_IMAGE = DefaultUnload //Unload function of the driver
```

It can be seen that the unloading function of BlanDrv is DefaultUnload(), which is defined in the source file BlankDrv.c. In the function, it mainly implements the following functions:

- Traverse the system's handle database and store all device handles in a buffer array;
- The handles in the buffer array, that is, all device handles in the system, are judged. If the driver is used, it is disconnected from the controller;
- Uninstall all protocols installed by the driver, including EFI_DRIVER_BINDING_PROTOCOL, EFI_COMPONENT_NAME_PROTOCOL and EFI_COMPONENT_NAME2_PROTOCOL.

The code of the DefaultUnload() function is long, so I will not list it here. You can go directly to the source file to view its implementation.

At this point, the framework of the UEFI driver model is complete. Copy the BlankDrv project file to the folder RobinPkg\Drivers, and add the compilation path in the [Components] Section of RobinPkg.dsc:

```
RobinPkg/Drivers/BlankDrv/BlankDrv.inf
```

Start the UEFI compilation environment and run the following command to compile the IA32 UEFI driver.

```
C:\UEFIWorkspace>build -t VS2015x86 -p RobinPkg\RobinPkg.dsc \
-m RobinPkg\Drivers\BlankDrv\BlankDrv.inf -a IA32
```

2. Test Drive

Different from service drivers, BlankDrv provides the function of uninstalling drivers. Therefore, you can use the unload command to uninstall the driver.

Since this framework does not specify a hardware device, it can still be tested in the EDK2 simulator.

In addition to the UEFI Shell commands load and dh introduced in UEFI Development Exploration 67, the commands drivers and unload are also used. The usage of these two commands is introduced as follows.

- **The drivers** command is used to list the system's drivers. Its syntax is:

```
drivers [-l XXX] [-sfo]
```

The parameter -l is used to specify the language code used, such as ISO 639-2; the parameter -sfo uses the standard format to output the displayed information. The drivers command prints out the driver information that conforms to the UEFI driver model in the form of a list. The printed information includes the following items:

The DRV item indicates the driver handle number;
the VERSION item indicates the driver version number;
the TYPE item indicates the driver type, B is a bus driver, and D is a color palette driver;
the CFG item indicates whether the driver supports the Driver Configuration Protocol;
the DIAG item indicates whether the driver supports the Driver Diagnostics Protocol;
the #D item shows the number of devices controlled by the driver;
the #C item indicates the number of sub-devices;
the DRIVER NAME item indicates the driver name;
and the IMAGE PATH indicates the driver source.

- **The unload** command is used to uninstall the UEFI driver. Its syntax format is:

```
unload [-n] [-v|-verbose] Handle
```

This command is used to clear the driver from the memory . The driver to be cleared is represented by its loaded handle. The unload command provides two parameters. "-n" means skipping all prompt information during execution; "-v" lists the information related to the specified handle during execution, including the driver file name, the symbol file of the driver file, the image loading base address, etc.

Start the EDK2 simulator, use the load command to load the driver, then use the drivers or dh command to check the driver loading status, and finally use the unload command to unload the driver. The operation is shown in Example 2.

[Example 2] Test BlankDrv in EDK2 simulator

```
FS0:\> load BlankDrv.efi    //Load driver
Image 'FS0:\BlankDrv.efi' loaded at 6211000 – Success
FS0:\> drviers           //List all drivers in the system
…… //Other driver information
87 0000000A ? – – – – My Blank Driver \BlankDrv.efi
FS0:\> dh 87             //Display device handle information
87: ComponentName2 ComponentName DriverBinding ImageDevicePath
(..8881,00000000)\BlankDrv.efi) LoadImage(\BlankDrv.efi)
FS0:\> unload 87        //Unload driver
Unload – Handle [52E0290]. [y/n]?
y
Unload – Handle [52E0290] Result Success
```

So far, the introduction to the UEFI driver model is complete. BlankDrv does not implement any actual functions, but mainly provides a relatively complete driver framework. In the next article, we will take an open source UEFI driver GopRotate as an example to build a test UEFI application for it and demonstrate the interesting screen change function.

To facilitate everyone's experiments, the source code of BlankDrv is placed in the gitee repository.

Gitee address: https://gitee.com/luobing4365/uefi-explorer
Project code is located at: / 69 BlankDrv/BlankDrv.zip