在前一篇文章中，我们介绍了如何在 Ubuntu 上为 Android 系统编写 Linux 内核驱动程序。在这个名为 hello 的 Linux 内核驱动程序中，创建三个不同的文件节点来供用户空间访问，分别是传统的设备文件/dev/hello、proc 系统文件/proc/hello 和 devfs 系统属性文件/sys/class/hello/hello/val。进一步，还通过 cat 命令来直接访问/proc/hello 和/sys/class/hello/hello/val 文件来，以验证驱动程序的正确性。在这一篇文章里，我们将通过自己编写的 C 可执行程序来访问设备文件/dev/hello。可能读者会觉得奇怪，怎么能在 Android 系统中用 C 语言来编写应用程序呢？Android 系统上的应用程序不都是 Java 应用程序吗？其实是可以的，读者不妨用 adb shell 命令连上 Android 模拟器，在/system/bin 目录下可以看到很多 C 可执行程序，如 cat 命令。今天，我们就来学习一下怎么在 Android 系统中添加用 C 语言编写的可执行程序吧。

一. 参照在 Ubuntu 上为 Android 系统编写 Linux 内核驱动程序一文，准备好 Linux 驱动程序。使用 Android 模拟器加载包含这个 Linux 驱动程序的内核文件，并且使用 adb shell 命令连接上模拟，验证在/dev 目录中存在设备文件 hello。

二. 进入到 Android 源代码工程的 external 目录，创建 hello 目录:

USER-NAME@MACHINE-NAME:~/Android$ cd external

USER-NAME@MACHINE-NAME:~/Android/external$ mkdir hello

三. 在 hello 目录中新建 hello.c 文件：

view plain

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.  #include <fcntl.h>
4.  #define DEVICE_NAME "/dev/hello"
5.  int main(int argc, char** argv)
6.  {
7.      int fd = -1;
8.      int val = 0;
9.      fd = open(DEVICE_NAME, O_RDWR);
10.     if(fd == -1) {
11.         printf("Failed to open device %s./n", DEVICE_NAME);
12.         return -1;
13.     }
14.
15.     printf("Read original value:/n");
```

```
    read(fd, &val, sizeof(val));
    printf("%d./n/n", val);
    val = 5;
    printf("Write value %d to %s./n/n", val, DEVICE_NAME);
        write(fd, &val, sizeof(val));

    printf("Read the value again:/n");
        read(fd, &val, sizeof(val));
        printf("%d./n/n", val);
    close(fd);
    return 0;
}
```

这个程序的作用中，打开/dev/hello 文件，然后先读出/dev/hello 文件中的值，接着写入值 5 到/dev/hello 中去，最后再次读出/dev/hello 文件中的值，看看是否是我们刚才写入的值 5。从/dev/hello 文件读写的值实际上就是我们虚拟的硬件的寄存器 val 的值。

四. 在 hello 目录中新建 Android.mk 文件：

LOCAL_PATH := $(call my-dir)

include $(CLEAR_VARS)

LOCAL_MODULE_TAGS := optional

LOCAL_MODULE := hello

LOCAL_SRC_FILES := $(call all-subdir-c-files)

include $(BUILD_EXECUTABLE)

注意，BUILD_EXECUTABLE 表示我们要编译的是可执行程序。

五. 参照如何单独编译 Android 源代码中的模块一文，使用 mmm 命令进行编译：

USER-NAME@MACHINE-NAME:~/Android$ mmm ./external/hello

编译成功后，就可以在 out/target/product/gernerc/system/bin 目录下，看到可执行文件 hello 了。

六. 重新打包 Android 系统文件 system.img：

USER-NAME@MACHINE-NAME:~/Android$ make snod

这样，重新打包后的 system.img 文件就包含刚才编译好的 hello 可执行文件了。

七. 运行模拟器，使用/system/bin/hello 可执行程序来访问 Linux 内核驱动程序：

USER-NAME@MACHINE-NAME:~/Android$ emulator -kernel ./kernel/common/arch/arm/boot/zImage &

USER-NAME@MACHINE-NAME:~/Android$ adb shell

root@android:/ # cd system/bin

root@android:/system/bin # ./hello

Read the original value:

0.

Write value 5 to /dev/hello.

Read the value again:

5.

看到这个结果，就说我们编写的 C 可执行程序可以访问我们编写的 Linux 内核驱动程序了。

介绍完了如何使用 C 语言编写的可执行程序来访问我们的 Linux 内核驱动程序，读者可能会问，能不能在 Android 的 Application Frameworks 提供 Java 接口来访问 Linux 内核驱动程序呢？可以的，接下来的几篇文章中，我们将介绍如何在 Android 的 Application Frameworks 中，增加 Java 接口来访问 Linux 内核驱动程序，敬请期待。