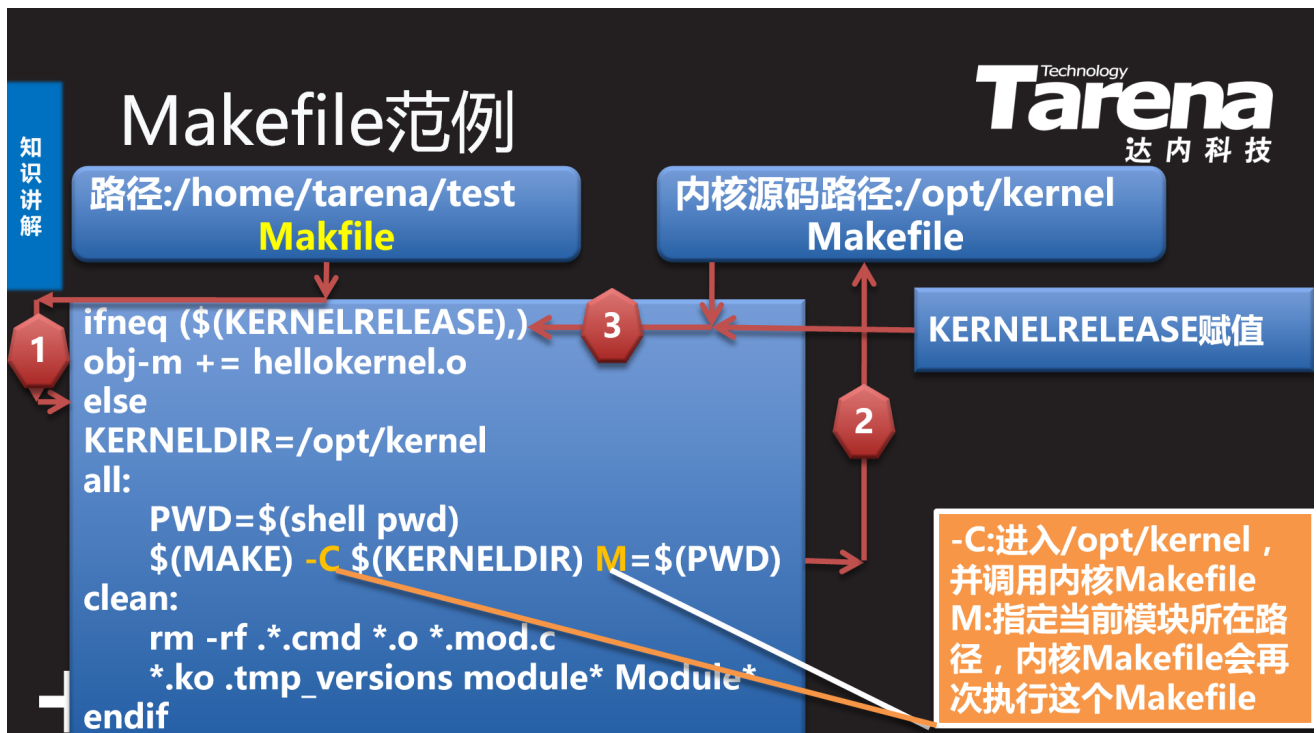


Makefile理解

- 内核Makefile编译modules过程
- 多文件的Makefile编写
- 内核模块的加载
- 内核模块参数的使用
- 内核打印printk
- 内核模块开发特点
- 内核打印参数

Makefile理解

内核Makefile编译modules过程



obj-m:表示编译为Modules模块；

obj-y:表示编译进内核

多文件的Makefile编写

多文件

hellokernel.c模块

file1.c

file2.c

file1_test()

file2_test()



多文件Makefile范例

 路径:/home/tarena/test
Makfile

 内核源码路径:/opt/kernel
Makefile

```

1 ifneq ($(KERNELRELEASE),)
  obj-m += hellokernel.o file1.o file2.o
else
  KERNELDIR=/opt/kernel
all:
  $(MAKE) -C $(KERNELDIR) M=$(PWD)
clean:
  rm -rf *.cmd *.o *.mod.c
  *.ko .tmp_versions module* Module*
endif
  
```

KERNELRELEASE赋值

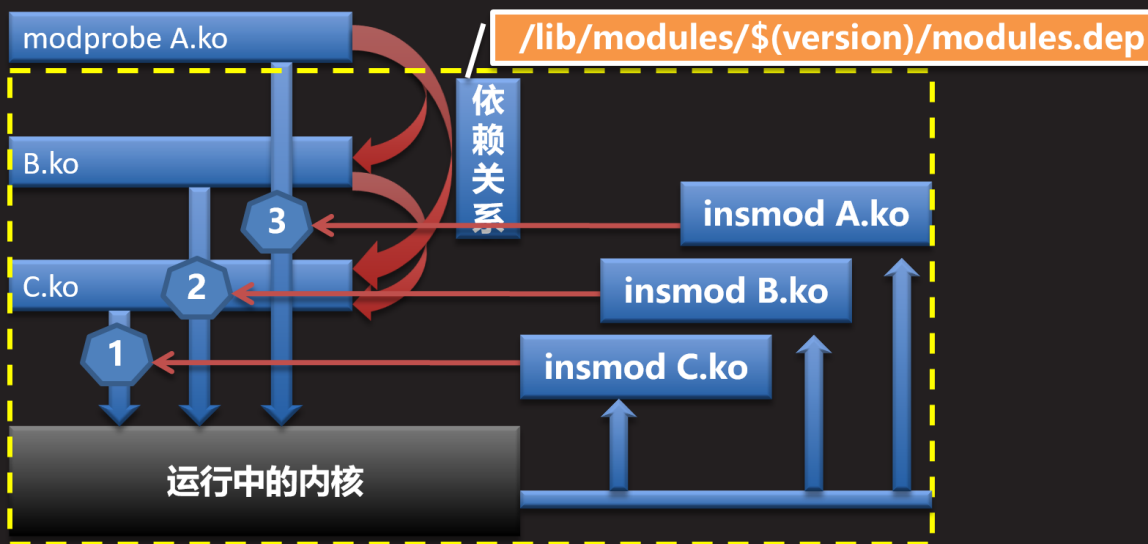
-C:进入/opt/kernel，并调用内核Makefile。
M:指定当前模块所在路径，内核Makefile会再次执行这个Makefile。
file1.c和file2.c可以有module_init和module_exit

注意：如果使用obj-m +=1.o 2.o 3.o，那么2.c和3.c可以有module_init()和module_exit()文件

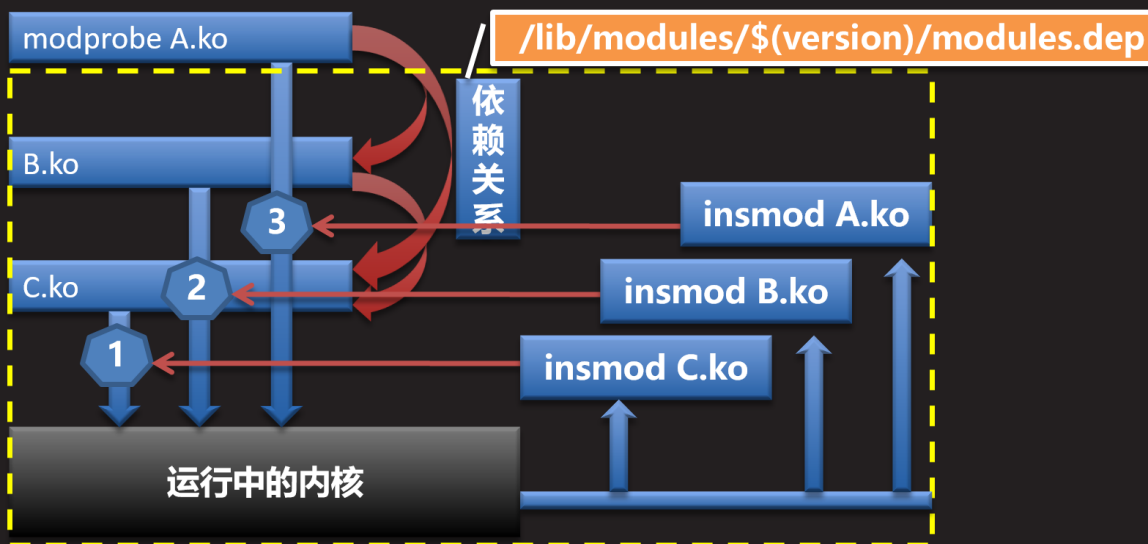
内核模块的加载

除了熟悉的insmod命令之外，对于依赖特别多，而且又不知道具体的依赖关系的模块，可以用modprobe命令来加载内核模块：

modprobe

Technology
Tarena
达内科技

modprobe

Technology
Tarena
达内科技

modules.dep

P
C
平
台

make modules_install

安装到 : /lib/下

A
R
M
平
台

make modules_install INSTALL_MOD_PATH=/home/tarena/

1.在/home/tarena/下生成lib目录
2.将lib下所有内容拷贝到开发板根文件系统的lib目录中



内核模块参数的使用

内核模块参数

`module_param(name,type,perm);`
功能：指定模块参数，用于在加载模块时或者模块加载以后传递参数给模块。
name：模块参数的名称
type：模块参数的数据类型
perm：模块参数的访问权限

权限一般用8进制表示即可,如0664.

模块文件:hellokernel.c
int irq;
char *pstr;
module_param(irq, int, 0664);
module_param(pstr, charp, 0);

irq

/sys/module/hellokernel/parameters/irq

pstr

/sys/module/hellokernel/parameters/没有



内核模块参数数组

```
module_param(name,type,nump,perm);
```

功能：指定模块参数，用于在加载模块时或者模块加载以后传递参数给模块。
name：模块参数的名称
type：模块参数的数据类型
nump：数组元素个数指针
perm：模块参数的访问权限

```
static int fish[10];
static int nr_fish;
module_param_array(fish,int,&nr_fish,0664);
```

nr_fish:保存最终传递数组元素个数，不能大于10个



内核模块参数使用

模块文件:hellokernel.c

```
static int irq;
static char *pstr;
static int fish[10];
static int nr_fish;
module_param(pstr, charp, 0);
module_param(irq, int, 0664);
module_param_array(fish,int,&nr_fish,0664);
```



```
insmod hellokernel.ko irq=100 pstr=china fish=1,2,3
echo 200 > /sys/module/hellokernel/parameters/irq
echo 10,20,30 > /sys/module/hellokernel/parameters/fish
```

适应方法：insmod hellokernel.ko irq=100 pstr=tarena fish=1,2,3,4,5

查看打印信息

cat /sys/module/hellokernel/parameters/irq //查看irq文件内容

cat /sys/module/hellokernel/parameters/fish //查看fish文件内容

修改以上两个文件内容

echo 2000 > /sys/module/hellokernel/parameters/irq

echo 10,20,30,40,50 > /sys/module/hellokernel/parameters/fish 注意：修改文件等于修改变量的内容

rmmod hellokernel //查看变量的内容是否修改 切记：一旦权限不为0，那么就会生成跟变量名同名的文件，会占用内存空间；为0，不占用内存；所有要考虑好是否真的需要指定一个非0权限！



内核默认的printk的输出级别的设置方法：

1. 通过修改配置文件/proc/sys/kernel/printk

```
cat /proc/sys/kernel/printk
```

7 4 1 7 //关注第一个数字7即可，它就是默认的输出级别，

只要是小于7的printk信息都可以输出，大于等于7的不输出

echo 8 > /proc/sys/kernel/printk 降低输出级别，这样的输出信息都可以打印

insmod hellokernel.ko //查看一下输出的信息

2.第一种方法无法解决内核启动的输出信息，只能通过设置uboot的启动参数bootargs来指定输出级别

```
setenv bootargs root=/dev/nfs nfsroot=192.168.1.8:/opt/rootfs
ip=192.168.1.6:192.168.1.8:192.168.1.1:255.255.255.0::eth0:on init=/linuxrc
console=ttySAC0,115200 debug (设置默认的输出级别为10)
```

```
setenv bootargs root=/dev/nfs nfsroot=192.168.1.8:/opt/rootfs
ip=192.168.1.6:192.168.1.8:192.168.1.1:255.255.255.0::eth0:on init=/linuxrc
console=ttySAC0,115200 quiet (设置默认的输出级别为4)
```

```
setenv bootargs root=/dev/nfs nfsroot=192.168.1.8:/opt/rootfs
ip=192.168.1.6:192.168.1.8:192.168.1.1:255.255.255.0::eth0:on init=/linuxrc
console=ttySAC0,115200 loglevel=8
```

内核模块开发特点

1. 不能访问C库也不能访问标准的C头文件
2. 必须使用GNU C
3. 没有内存保护机制
4. 难以执行浮点运算
5. 内核给每一个进程只有一个很小的定长堆栈（8KB）
6. 内核支持中断，抢占和SMP，时刻注意同步和并发
7. 要考虑可移植性的重要性

内核打印参数

```
void test_module(void)
{
    printk("%s, %s, %d\n",
           __FILE__, __func__, __LINE__);
}
//__FILE__:函数所在的文件名

//__func__:函数的名字

//__LINE__:函数所定义的行数
```