

拷贝Linux文件系统

背景

操作步骤，分为三步：

- 一、使用rsync工具拷贝根文件系统
- 二、制作ext4的根文件系统
- 三、烧写测试

拷贝Linux文件系统

背景

一般的Linux发行版ubuntu和Debian，除了u-boot和kernel可以从源码构建，根文件系统是一个编译好的img文件。当我们在Linux安装好自己程序的运行或者开发环境，想把板子的系统移植到其他板子，就需要把板子的文件系统拷贝下来，重新打包为一个整包镜像或者根文件系统镜像（rootfs.img），烧录到其他板子

操作步骤，分为三步：

一、使用rsync工具拷贝根文件系统

准备工作：

- 1、开发板插上网线，板子和电脑处于同一个网段，并能连接到外网
- 2、通过USB转TTL串口设备连接板子的debug串口，进入板子系统（比如使用xshell工具，串口波特率为115200），执行一下操作

+ 通过ifconfig获取以太网的ip

```
root@linaro-alip:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.1.199  netmask 255.255.255.0  broadcast 192.168.1.255
    inet6 fe80::ee8d:ec48:ff2:78b5  prefixlen 64  scopeid 0x20<link>
    ether 3e:8a:6d:b3:2c:d7  txqueuelen 1000  (Ethernet)
    RX packets 6484  bytes 468943 (457.9 KiB)
    RX errors 0  dropped 1  overruns 0  frame 0
    TX packets 231  bytes 39905 (38.9 KiB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
    device interrupt 27
```

开始拷贝：

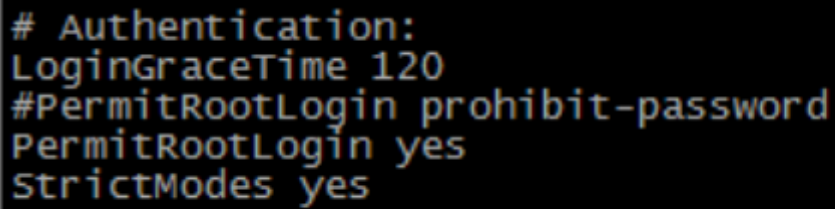
- 1、准备一台Ubuntu系统的实体机或虚拟机作为PC端
- 2、在开发板的 UBUNTU 系统上面安装软件 openssh 的服务端 openssh-server：

```
sudo apt-get update #密码为rpdzkj
sudo apt-get install openssh-server
```

3、修改开发板 root 登录权限:

```
sudo vi /etc/ssh/sshd_config
```

4、在 sshd_config 中修改 PermitRootLogin 选项如下, 用以确保 root 登录权限足够



```
# Authentication:
LoginGraceTime 120
#PermitRootLogin prohibit-password
PermitRootLogin yes
StrictModes yes
```

5、重启开发板或者重启 ssh 服务

```
/etc/init.d/ssh restart
```

6、PC 端安装 openssh 客户端 openssh-client

```
sudo apt-get install openssh-client
```

7、PC 端安装 rsync

```
sudo apt-get install rsync
```

8、PC 新建目录 ubuntu_make 并进入并创建Ubuntu目录:

```
mkdir ubuntu_make
cd ubuntu_make
mkdir ubuntu
```

9、同步数据:

```
sudo rsync -avx root@<开发板 IP>:/ ubuntu/
```

二、制作ext4的根文件系统

1、同步完成后，PC 进入 ubuntu 文件夹下面增加脚本文件 etc/onlyone.sh

```
cd ubuntu
sudo vim etc/onlyone.sh
```

onlyone.sh 内容为:

```
#!/bin/sh

read line < /proc/cmdline

for arg in $line; do

    if [ "5" -le "$(expr length $arg)" ]; then

        if [ "root=" = "$(expr substr $arg 1 5)" ]; then

            {

                *debug_arg=$(expr $arg : 'root=\.\\*\')

                resize2fs $debug_arg

            }

        fi

    fi

done
```

2、.PC 端 ubuntu 文件夹下更改该文件权限:

```
sudo chmod 777 etc/onlyone.sh
```

3、返回上级目录 (make_ubuntu) , 并制作一个make_ubuntu.sh 到当前文件夹:

```
cd ..
vim make_ubuntu.sh
```

make_ubuntu.sh的内容如下

```
#!/bin/bash

#ubuntu(ubuntu-core) build already annotation
#not often compiled .....too slow and need root
MAKE_THEARD=`cat /proc/cpuinfo| grep "processor"| wc -l`
```

```
RESULT="ubuntu-rootfs"
```

```
function creat_result_dir()
```

```
{  
    if [ ! -d $RESULT ];  
        then  
            {  
                mkdir Image-rk3288-ubuntu  
            }  
        fi  
}
```

```
function ubuntu_core_build()
```

```
{  
    dd if=/dev/zero of=linux-rootfs-core.img bs=1M count=1000  
    sudo mkfs.ext4 -F -L linuxroot linux-rootfs-core.img  
  
    if [ ! -d mount ];  
        then  
            {  
                mkdir mount  
            }  
        fi  
    sudo mount linux-rootfs-core.img mount  
    sudo cp -a ubuntu_core/* mount  
    sudo umount mount  
    e2fsck -p -f linux-rootfs-core.img  
    resize2fs -M linux-rootfs-core.img  
    rm -rf mount  
    mv linux-rootfs-core.img $RESULT  
}
```

```
function ubuntu_build()
```

```
{  
    dd if=/dev/zero of=linux-rootfs.img bs=1M count=5000  
    sudo mkfs.ext4 -F -L linuxroot linux-rootfs.img  
  
    if [ ! -d mount ];  
        then  
            {  
                mkdir mount  
            }  
        fi  
    sudo mount linux-rootfs.img mount  
    sudo cp -a ubuntu/* mount  
    sudo umount mount  
    e2fsck -p -f linux-rootfs.img  
    resize2fs -M linux-rootfs.img  
    rm -rf mount  
    mv linux-rootfs.img $RESULT  
}
```

```
function ubuntu_clean()
```

```
{  
    rm $RESULT/linux-rootfs.img  
}
```

```
function ubuntu_core_clean()
```

```
{
```

```

rm $RESULT/linux-rootfs-core.img
}
function result_clean()
{
    rm -rf $RESULT
}

# creat_result_dir

if [ $1 == "clean" ]
then
{
    if [ ! -n $2 ]
    then
    {
        ubuntu_core_clean
        ubuntu_clean
        result_clean
        echo clean Img ok
    }
    elif [ $2 == "ubuntu" -o $2 == "ubuntu/" ]
    then
    {
        ubuntu_clean
    }
    elif [ $2 == "ubuntu_core" -o $2 == "ubuntu_core/" -o $2 == "ubuntu-
core" -o $2 == "ubuntu-core/" ]
    then
    {
        ubuntu_core_clean
    }
    else
    {
        ubuntu_core_clean
        ubuntu_clean
        result_clean
        echo clean Img ok
    }
    fi
}
elif [ $1 == "ubuntu_core" -o $1 == "ubuntu_core/" -o $1 == "ubuntu-core" -o $1
== "ubuntu-core/" ]
then
{
    ubuntu_core_build
}
elif [ $1 == "ubuntu" -o $1 == "ubuntu/" ]
then
{
    ubuntu_build
}
else
{
    ubuntu_core_build
    ubuntu_build
}
fi

```

并赋予该脚本777权限

```
sudo chmod 777 make_ubuntu.sh
```

4、运行 make_ubuntu.sh 制作 ubuntu 固件:

```
sudo ./make_ubuntu.sh ubuntu
```

5、或制作 ubuntu_core: (只有你本来的文件系统是Ubuntu_core才需要这步)

```
sudo ./make_ubuntu.sh ubuntu_core
```

脚本运行完成, 在ubuntu-rootfs 目录下生成 ubuntu_core、ubuntu 固件

```
rpdkj@ubuntu:~/debug/ubuntu_make/ubuntu$ ls
linux-rootfs-core.img  linux-rootfs.umd
rpdkj@ubuntu:~/debug/ubuntu_make/ubuntu$
```

三、烧写测试

1、做好的img镜像重新命名为rootfs.img,可以将其分区烧写, 验证该文件系统是OK的再打包为整包镜像来批量烧写

2、下载网盘SDK下的Linux源码, 解压到PC端的一个空白目录

3、将上述 linux-rootfs.img/linux-rootfs-core.img 命名为ubuntu.img, 替换源码ubuntu目录下的相同文件 (没有ubuntu目录就建立一个)

,返回源码根目录, 运行编译脚本生成**整包固件**:

```
rpdkj@rpdkj:~/first/linux/px30-linux$ ./build.sh init #第一次编译要选择板型和系统,
这里以rp-px30为例子
processing option: init
=====You're building on Linux=====
Please choose BoardConfig
 1. BoardConfig_rp_px30_buildroot
 2. BoardConfig_rp_px30_ubuntu
 3. BoardConfig_rp_px30_ubuntu_core
 4. BoardConfig_rp_px30_debian
 5. BoardConfig_nano_px30_buildroot
 6. BoardConfig_nano_px30_ubuntu
 7. BoardConfig_nano_px30_ubuntu_core
 8. BoardConfig_nano_px30_debian
```

```
9. BoardConfig_dr4_px30_buildroot
10. BoardConfig_dr4_px30_ubuntu
11. BoardConfig_dr4_px30_ubuntu_core
12. BoardConfig_dr4_px30_debian
Please input num:

rpdzkj@rpdzkj:~/first/linux/px30-linux$ ./build.sh #先整包编译一遍，也就是先编译u-
boot和kernel，如果你不                是第一次编译，
这一步可以不执行
rpdzkj@rpdzkj:~/first/linux/px30-linux$ ./build.sh firmware
rpdzkj@rpdzkj:~/first/linux/px30-linux$ ./build.sh updateimg
```

编译完成后，即可在 rockdev/下生成 update-*.img