Table of Contents

介绍

这本书是介绍思维方法的，帮助我们理解现代技术的某些方面的细节。

可以利用插件，非常方便地输入数学公式。图片的插入使用了标准的Markdown语法。

$$\int_{-\infty}^{\infty} g(x)dx$$

Hello! this is the book.

技术应用笔记

一些和技术相关的笔记。

Intro

添加RK3568的boot启动分区，这样通过修改boot文件系统上的kernel,dtb文件，就可以达到升级系统的目的。

根据Rockchip的文档SDK/docs/common/Uboot/Rockchip_Developer_Guide_uboot_nextdev_CN.pdf, SDK中提供的u-boot默认支持distro模式，可以很好的支持文件系统启动。

RK3568 u-boot

u-boot使用CONFIG_BOOTCOMMAND，启动时会做设备检测,检测顺序为Android格式固件、RK FIT(Flat Image Tree)格式固件、RKP格式固件和文件系统格式固件。

```
#define RKIMG_BOOTCOMMAND            \
    "boot_android {% math_inline %}{devtype} {% endmath_inline %}{devnum};"    \
    "boot_fit;"                \
    "bootrkp;"                \
    "run distro_bootcmd;"
#endif
```

将boot格式化为一个256MB的分区，拷贝kernel文件，dtb文件,并提供extlinux/extlinux.conf配置文件供u-boot读取。这样uboot启动时运行distro_bootcmd会自动检测到文件系统分区，并根据extlinux.conf文件加载设备树,启动kernel,加载rootfs分区，完成启动。

生成boot分区

原来的packed update.img的分配格式如SDK/device/rockchip/rk356x/parameter.txt文件所示.

```
FIRMWARE_VER: 1.0
MACHINE_MODEL: RK3568
MACHINE_ID: 007
MANUFACTURER: RK3568
MAGIC: 0x5041524B
ATAG: 0x00200800
MACHINE: 0xffffffff
CHECK_MASK: 0x80
PWR_HLD: 0,0,A,0,1
TYPE: GPT
CMDLINE: mtdparts=rk29xxnand:0x00002000@0x00004000(uboot),\
0x00002000@0x00006000(misc),\
0x00040000@0x00008000(boot:bootable),\
0x00018000@0x00048000(recovery),\
0x00008000@0x00060000(backup),\
-@0x00068000(rootfs:grow)
uuid:rootfs=614e0000-0000-4b53-8000-1d28000054a9
```

其中, CMDLINE的格式为 size@offset。boot部分为0x40000, 单位为sector, 512Btyes, 因此0x40000=2^18=256K, 256K*512=128MByte。

这样预留给boot分区的空间就是128MB.

生成boot image

这里借鉴了firefly RK3568 SDK的脚本文件。建立一个目录extboot/, 在目录下生成extlinux/extlinux.conf文件，将SDK/kernel/arch/arm64/boot/Image文件,SDK/kernel/arch/arm64/boot/dts/*.dtb文件拷贝进行。

使用mkfs.ext4生成image,其中包含了extboot/目录下的内容。

上述过程在build.sh脚本里添加函数模块，自动生成。

```
rm -rf {% math_inline %}{EXTBOOT_DIR} && mkdir -p {% endmath_inline %}{EXTBOOT_DIR}/

KERNEL_VERSION={% math_inline %}(cat {% endmath_inline %}TOP_DIR/kernel/include/conf
echo "label rk-kernel.dtb linux-{% math_inline %}KERNEL_VERSION" > {% endmath_inline

cp {% math_inline %}{TOP_DIR}/{% endmath_inline %}RK_KERNEL_IMG {% math_inline %}EXT
echo -e "\tkernel /Image-{% math_inline %}KERNEL_VERSION" >> {% endmath_inline %}EXT

cp {% math_inline %}{TOP_DIR}/kernel/arch/{% endmath_inline %}{RK_ARCH}/boot/dts/roc

ln -sf {% math_inline %}{RK_KERNEL_DTS}.dtb {% endmath_inline %}EXTBOOT_DIR/rk-kerne

echo -e "\tfdt /rk-kernel.dtb" >> {% math_inline %}EXTBOOT_DIR/extlinux/extlinux.con

cp {% endmath_inline %}{TOP_DIR}/kernel/.config {% math_inline %}EXTBOOT_DIR/config-
cp {% math_inline %}{TOP_DIR}/kernel/System.map {% endmath_inline %}EXTBOOT_DIR/Syst

make ARCH={% endmath_inline %}RK_ARCH INSTALL_MOD_STRIP=1 INSTALL_MOD_PATH={% math_i

EXTBOOT_IMG_SIZE=128M

rm -rf {% endmath_inline %}EXTBOOT_IMG && truncate -s {% math_inline %}EXTBOOT_IMG_S
fakeroot {% math_inline %}{TOP_DIR}/device/rockchip/common/mkfs.ext4 -Fq -L "boot" -
```

将生成的extboot.img作为新的boot.img打包进update.img

下载测试

通过启动时的debug console看到uboot检测分区格式的过程，最终uboot通过distro command找到emmc0的boot分区，并加载kernel文件，进入正常启动。耗时<1s。

```
U-Boot 2017.09(u-boot commit id: 0f621b0e434b6944de865b517687e096930885b9)(sdk versi

Model: Hummingbird Board
PreSerial: 2, raw, 0xfe660000
DRAM:  4 GiB
Sysmem: init
Relocation Offset: ed249000
Relocation fdt: eb9f86a0 - eb9fecd8
CR: M/C/I

...


Hit key to stop autoboot('CTRL+C'):  0
ANDROID: reboot reason: "(none)"
optee api revision: 2.0
TEEC: Waring: Could not find security partition
Not AVB images, AVB skip
No valid android hdr
Android image load failed
Android boot failed, error -1.
## Booting FIT Image FIT: No fit blob
FIT: No FIT image

## Booting Rockchip Format Image
Could not find kernel partition, ret=-1
Card did not respond to voltage select!
mmc_init: -95, time 10
switch to partitions #0, OK
mmc0(part 0) is current device
Scanning mmc 0:3...
Found /extlinux/extlinux.conf
Retrieving file: /extlinux/extlinux.conf
=================begin===================
79 bytes read in 3 ms (25.4 KiB/s)
1:    rk-kernel.dtb linux-4.19.219
Retrieving file: /Image-4.19.219
=================begin===================
28135432 bytes read in 159 ms (168.8 MiB/s)
Retrieving file: /rk-kernel.dtb
=================begin===================
121632 bytes read in 5 ms (23.2 MiB/s)
Fdt Ramdisk skip relocation
## Flattened Device Tree blob at 0x08300000
   Booting using the fdt blob at 0x08300000
  'reserved-memory' ramoops@110000: addr=110000 size=f0000
   Using Device Tree in place at 0000000008300000, end 0000000008320b1f
No resource partition
No file: logo_kernel.bmp
** File not found logo.bmp **
```

```
Adding bank: 0x00200000 - 0x08400000 (size: 0x08200000)
Adding bank: 0x09400000 - 0xf0000000 (size: 0xe6c00000)
Adding bank: 0x1f0000000 - 0x200000000 (size: 0x10000000)
Total: 524.4 ms

Starting kernel ...
```

加载并查看boot分区

```
Disklabel type: gpt
Disk identifier: 1124E418-9008-41F8-9E3D-8D872216C8A1

Device          Start      End  Sectors  Size Type
/dev/mmcblk0p1  16384    24575     8192    4M unknown
/dev/mmcblk0p2  24576    32767     8192    4M unknown
/dev/mmcblk0p3  32768   557055   524288  256M unknown
/dev/mmcblk0p4 557056   655359    98304   48M unknown
/dev/mmcblk0p5 655360   688127    32768   16M unknown
/dev/mmcblk0p6 688128 61071295 60383168 28.8G unknown
pi@hummingbird:~$ sudo mount /dev/mmcblk0p3 /boot
pi@hummingbird:~$ ls /boot
Image-4.19.219       config-4.19.219  ido-rk3568-ctb3516.dtb  lost+found
System.map-4.19.219  extlinux         lib                     rk-kernel.dtb
pi@hummingbird:~$ df
Filesystem     1K-blocks    Used Available Use% Mounted on
/dev/root       29700044  400600  28076088   2% /
devtmpfs         1987112       0   1987112   0% /dev
tmpfs            1996168       0   1996168   0% /dev/shm
tmpfs             399236    1056    398180   1% /run
tmpfs               5120       0      5120   0% /run/lock
tmpfs            1996168       0   1996168   0% /sys/fs/cgroup
tmpfs             399232       0    399232   0% /run/user/1000
/dev/mmcblk0p3    229344   34044    176956  17% /boot
pi@hummingbird:~$
```

结束

What is libsrt?

Problem facing

Gstreamer 1.14.4 doesnt support streamid property in its srtserversink plugin.

libsrt way of handling streamid

It seems to pass it to srt_setsockopt() function.

SRT doesn't do URI parsing at all, only describe the socket option, doesn't mention the URI.

It is a valid case to set a socket option of string type to a string value containing printable character. It is a valid case to call srt_setsockflag (or srt_setsockopt) function using SRT_STREAMID, as well as extract this option's value from the socket.

The srt socket option,

srt-live-transmit application,

socket-groups.md,

srt-test-multiplex,

GStreamer application is using a different approach,

```
srtserversink uri=srt://xx.xx.xx.xx:8080 latency=100
```

VLC and FFMPEG both support the use case, support setting streamid by option.

SRT API Socket Options

Support int32, int64, bool, string, linger,

SRT C API srt.h, is based on the legacy UDT API,

srt-live-transmit, srt-file-transmit,

1. setup and teardown, srt_startup(), srt-cleanup(),
2. Creating and Destroying a Socket, SRT socket, srt_create_socket(), srt_close(STRSocket s) Use UDP socket underlying,
3. Binding and Connecting srt_bind(SRTSOCKET u, struct sockaddr* name, int namelen) srt_bind_acquire(SRTSOCKET u, UDPSOCKET updsock) srt_listen() srt_accept() srt_connect() srt_connect_debug()
4. Rendezvous SRTO_RENDEZVOUS flag, connect to rendezvous counterpart lsa, local ip/port, rsa, remote ip/port, srt_setsockopt(m_sock,0,SRTO_RENDEZVOUS, &yes, sizeof yes) srt_bind() srt_connect() HandleConnection(sock)
5. Sending and receiving, srt_send(), srt_recv(), srt_sendmsg(), srt_recvmsg(),
6. srt epoll srt_epoll_wait() srt_epoll_uwait()

Some interesting points from Haivision github repo

```
Accept URIs with standard encoding - in this case, e.g.
srt://example.com:9000?streamid=%23%21%3A%3Au=admin,r=foo [EDIT: fixed]
```

srt source code, version 1.14.4

https://gitlab.freedesktop.org/gstreamer/gst-plugins-bad/-/tree/1.14.4

gstsrtclientsink.c

- gst_srt_client_sink_set_property()
  - priv->poll_timeout
  - priv->bind_address
  -
- gst_srt_client_sink_start
  - gst_srt_client_connect_full

srt source code, version 1.19

srt_params

srt_options[]

```
gst_srt_object_set_common_params()
    call srt_setsockopt()
    gst_srt_object_apply_socket_option()
        call function below,
        srt_setsockopt()

gst_srt_object_set_socket_option()
    gst_srt_object_set_string_value()
```

How to change the plugin code?

Materials

SRT and RIST are recent streaming protocols, a growing list of alternatives to RTMP(among wich one can find webrtc, warp/quit etc.)

SRT can recover from packet loss up to range of 15%, RIST up to 50% packet loss.

glibc GIO