TCCXXX
HOW TO CHANGE LOGO
USER GUIDE

TCCxxxx_Android 4.4.2(Kitkat-mr1.1)_v1.00E_How to change logo

Rev. 1.00

Mar 28, 2014

Telechips

DISCLAIMER

All information and data contained in this material are without any commitment, are not to be considered as an offer for conclusion of a contract, nor shall they be construed as to create any liability. Any new issue of this material invalidates previous issues. Product availability and delivery are exclusively subject to our respective order confirmation form; the same applies to orders based on development samples delivered. By this publication, Telechips, Inc. does not assume responsibility for patent infringements or other rights of third parties that may result from its use.

Further, Telechips, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.
No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of Telechips, Inc.
This product is designed for general purpose, and accordingly customer be responsible for all or any of intellectual property licenses required for actual application. Telechips, Inc. does not provide any indemnification for any intellectual properties owned by third party.
Telechips, Inc. can not ensure that this application is the proper and sufficient one for any other purposes but the one explicitly expressed herein. Telechips, Inc. is not responsible for any special, indirect, incidental or consequential damage or loss whatsoever resulting from the use of this application for other purposes.

COPYRIGHT STATEMENT

Important Notice

For customers who use licensed Codec ICs and/or licensed codec firmware of mp3:
"Supply of this product does not convey a license nor imply any right to distribute content created with this product in revenue-generating broadcast systems (terrestrial. Satellite, cable and/or other distribution channels), streaming applications(via internet, intranets and/or other networks), other content distribution systems(pay-audio or audio-on-demand applications and the like) or on physical media(compact discs, digital versatile discs, semiconductor chips, hard drives, memory cards and the like). An independent license for such use is required. For details, please visit http://mp3licensing.com".

For customers who use other firmware of mp3:
"Supply of this product does not convey a license under the relevant intellectual property of Thomson and/or Fraunhofer Gesellschaft nor imply any right to use this product in any finished end user or ready-to-use final product. An independent license for such use is required. For details, please visit http://mp3licensing.com".

For customers who use Digital Wave DRA solution:
"Supply of this implementation of DRA technology does not convey a license nor imply any right to this implementation in any finished end-user or ready-to-use terminal product. An independent license for such use is required."

For customers who use DTS technology:
"This product made under license to certain U.S. patents and/or foreign counterparts."
"© 1996 – 2011 DTS, Inc. All rights reserved."

For customers who use Dolby technology:
"Supply of this Implementation of Dolby technology does not convey a license nor imply a right under any patent, or any other industrial or intellectual property right of Dolby Laboratories, to use this Implementation in any finished end-user or ready-to-use final product. It is hereby notified that a license for such use is required from Dolby Laboratories."

For customers who use MS technology:
"This product is subject to certain intellectual property rights of Microsoft and cannot be used or disctributed further without the appropriate license(s) from Microsoft."

Revision History

Date	Version	Description
2014-03-28	1.00	initial release

TABLE OF CONTENTS

Contents

1 Introduction 1-1
1.1 Logo Change List 1-1
2 Bootloader Logo Change 2-2
2.1 When set to 16bit display output (default) 2-2
2.1.1 Change Bootloader Logo by Changing Logo.h 2-2
2.1.2 Make *.img File to extract Hex Value 2-2
2.1.3 Extract Hex Value 2-2
2.2 When set to 32bit display output 2-3
2.2.1 Change Bootloader Logo by Changing Logo_24bit.h 2-3
2.2.2 Make *.img File to extract Hex Value 2-3
2.2.3 Extract Hex Value 2-3
2.3 How to Change and Apply Bootloader Logo Size 2-4
2.4 Change Bootloader Logo by Changing Splash.img 2-4
2.4.1 Change lk bootloader settings 2-4
2.4.2 How to make splash image 2-5
2.4.3 How to download splash image to splash partition 2-6
2.4.3.1 Use SD/MMC or NAND V8 FTL only for Splash partition. 2-6
2.4.3.2 Fastboot mode download 2-6
3 Change Kernel Logo 3-7
3.1 Set Host Environment 3-7
3.2 Make Logo Image 3-7
3.2.1 Make *.ppm File 3-7
3.2.2 Make and Change *.c File 3-7
3.3 Change Linux Kernel Source 3-7
3.3.1 Change kernel/drivers/video/logo/Kconfig 3-7
3.3.2 Change kernel/drivers/video/logo/Makefile 3-8
3.3.3 Change kernel/include/linux/linux_logo.h 3-8
3.3.4 Change kernel/drivers/video/logo/logo.c 3-8
3.4 Kernel Configuration 3-8
4 Android init logo 4-10
4.1 Make Image (*.rle) for Android init Logo 4-10
4.1.1 When set to 16bit display output 4-10
4.1.2 When set to 32bit display output 4-10
4.2 Set Android init Logo Path 4-11
4.3 Apply Android init Logo 4-11
5 Android animation logo 5-12
5.1 Change android animation logo by modify image which responsible for animation. 5-12
5.1.1 Control Android Animation Logo 5-12
5.1.2 Image to Control Animation 5-12
5.1.3 Apply Android Animation Logo 5-12
5.2 Change android animation logo by bootanimation.zip 5-13
6 Notes. 6-14
6.1.1 When there is no ‘rgb2565’ 6-14
6.1.2 When there is no ‘rgbto888’ 6-14

1 Introduction

This document is intended to help to easily and quickly change various image logos used for blootloader, kernel, Android init and Android animation of Telechips. This document is for all platforms supported by Telechips.

※ Note that TCC893X is used as standard.

※ bmp image file saved in this document is 24bit.

1.1 Logo Change List

- Bootloader logo (telechips logo)
- Kernel logo (linux penguin image)
- Android init logo (androboy + telechips logo)
- Android animation logo

2 Bootloader Logo Change

2.1 When set to 16bit display output (default)

2.1.1 Change Bootloader Logo by Changing Logo.h

If you want to change your logo by changing the alignment of telechips_logo[] of Logo.h file, refer to the followings.

2.1.2 Make *.img File to extract Hex Value

Prepare 480x272 sized bmp image file in the following path, change it into *.img(raw) format as follows.

```
$ cd ~/mydroid/android/bootloader/bootable/lk
$ convert -depth 8 telechips_logo.bmp rgb:boot_logo.img  // bmp → img(raw)
$ rgb2565 <boot_logo.img> boot_logo_480X272.img          // RGB888 → RGB565
```

2.1.3 Extract Hex Value

In order to extract hex value from the *.img(raw) made in 2.1.2, a command called **hexdump** is used.

```
$ hexdump -v -e '/1 "0x%02x,"' boot_logo_480X272.img > boot_logo.h
```

If the above procedure is executed, hex values are arrayed in boot_logo.h file. Then, these values should be applied to the alignment of telechips_logo[].

```
…  /* omit */ …

unsigned char telechips_logo[]={
0x77,0xde,0x57,0xd6,0x57,0xd6,0x57,0xde,0x57,0xde,0x57,0xde,0x57,0xde,0x77,0xde,
0x57,0xde,0x57,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xde,
0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xde,0x77,0xde,0x77,0xde,0x77,0xde,
…  /* omit */ …
0x57,0xd6,0x57,0xd6,0x57,0xd6,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xde,
0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,
0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xd6,0x77,0xde,0x77,0xde,0x77,0xde,
}
…  /* omit */ …
```

2.2 When set to 32bit display output

If the user is set to 32bit display output setting in bootloader, must be set to 32bit all of part of which are described in the back (android init logo)

2.2.1 Change Bootloader Logo by Changing Logo_24bit.h

If you want to change your logo by changing the alignment of telechips_logo[] of Logo_24bit.h file, refer to the followings.

2.2.2 Make *.img File to extract Hex Value

Prepare 480x272 sized bmp image file in the following path, change it into *.img(raw) format as follows.

```
$ cd ~/mydroid/android/bootloader/bootable/lk
$ convert -depth 8 telechips_logo.bmp rgb:boot_logo.img  // bmp → img(raw)
$ rgbto888 <boot_logo.img> boot_logo_480X272.img         // RGB888 align
```

2.2.3 Extract Hex Value

In order to extract hex value from the *.img(raw) made in 2.2.2, a command called **hexdump** is used.

```
$ hexdump -v -e '/1 "0x%02x,"' boot_logo_480X272.img > boot_logo.h
```

If the above procedure is executed, hex values are arrayed in boot_logo.h file. Then, these values should be applied to the alignment of telechips_logo[].

```
…  /* omit */ …

unsigned char telechips_logo[]={
0x77,0xde,0x57,0xd6,0x57,0xd6,0x57,0xde,0x57,0xde,0x57,0xde,0x57,0xde,0x77,0xde,
0x57,0xde,0x57,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xde,
0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xde,0x77,0xde,0x77,0xde,0x77,0xde,
…  /* omit */ …
0x57,0xd6,0x57,0xd6,0x57,0xd6,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xde,
0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,
0x77,0xde,0x77,0xde,0x77,0xde,0x77,0xde,0x57,0xd6,0x77,0xde,0x77,0xde,0x77,0xde,
}
…  /* omit */ …
```

For 32bit output , set as follows.

```
$ cd ~/mydroid/android/bootloader/bootable/lk
$ vi platform/tcc893x/lcdc.c

//#define _LCD_32BPP_          //LCD output format setting
#define _LCD_32BPP_          //LCD output format setting
#if defined(_LCD_32BPP_)
#define LCDC_FB_BPP           32
#else
#define LCDC_FB_BPP           16
#endif
```

2.3 How to Change and Apply Bootloader Logo Size

You can find that 480X272 is set as a default for a bootloader logo size. If you want larger size of logo, you may 1024x600 sized bmp image in 2.1.1 and change 480 to 1024 and 272 to 600 in the following variables.

```
$ cd ~/mydroid/android/bootloader/bootable/lk
$ vi platform/tcc893x/lcdc.c

#define LCDC_FB_WIDTH   1024  //480
#define LCDC_FB_HEIGHT  600  //272
```

Then, if you have made a bootloader image and download it to a board, you can find the logo has been changed.
For more details on bootloader image creation and download, refer to Quick Start Guide in the Manual path.

2.4 Change Bootloader Logo by Changing Splash.img

If you want to change your Bootloader logo by making splash.img, refer to the followings.
(For more information on splash image display, refer **TCCxxxx-Android_4.4.2(Kitkat-mr1.1)-V0.01E-How to use splash partition image display**

2.4.1 Change lk bootloader settings

The following is how to set bootloader for a boot logo by using a splash image. The bootloader image made in the following procedure should be downloaded to a board.

```
$ cd ~/mydroid/android
$ vi bootable/bootloader/lk/target/tcc893x_evm/rules.mk
```

```
# Define Default Splash
ifeq ($(DISP_DEFINES), DISPLAY_DUAL)
DEFINES += DISPLAY_SPLASH_SCREEN_DIRECT=1
else
DEFINES += DISPLAY_SPLASH_SCREEN=1
#DEFINES += DISPLAY_SPLASH_SCREEN_DIRECT=0
endif
```

2.4.2 How to make splash image

The splash partition patch include the script to make splash image. before using this script, the convert utile is have to installed on your build system. the steps as below.

1. Configure the splash image in device/telechips/tcc893x/Boardconfig.mk

```
#Splash Image generate
TARGET_BOARD_SPLASH_USE := true
```

2. Build Android system than you can get the mkspalshimg script for making splash image. the mksplashimg is below (device/Telechips/common/splash)

```
# !bin/sh

for ((idx=0; idx<$NIMG; idx++));
do
#IMG_ORI[$idx]=${PARAMS[$idx+2]}
IMG_ORI[$idx]=${PARAMS[$idx+3]}
IMG_EXT[$idx]=${IMG_ORI[$idx]%%.*}.tmp
IMG_FN[$idx]=${IMG_ORI[$idx]%%.*}.img

IMG_RSL[$idx]=$(identify "${IMG_ORI[$idx]}" | cut -f 3 -d' ')

IMG_BITS[$idx]=$(file "${IMG_ORI[$idx]}" | cut -f 11 -d' ')

#convert image
if [ "$FMTSIZE" -eq "32" ];  then
convert -depth 8 ${IMG_ORI[$idx]} rgb:${IMG_EXT[$idx]}
rgbto888 <${IMG_EXT[$idx]}> ${IMG_FN[$idx]}
else if [ "$FMTSIZE" -eq "16" ] ; then
convert -depth 8 ${IMG_ORI[$idx]} rgb:${IMG_EXT[$idx]}
rgb2565 <${IMG_EXT[$idx]}> ${IMG_FN[$idx]}

else
echo "ERROR : [ choose fmt 16 or 32] "
ussage
fi
fi

done

echo ${IMG_RSL[*]}
echo ${IMG_BITS[*]}

MK_SPLASH="mksplash $PAGESIZE $NIMG ${IMG_FN[*]} ${IMG_RSL[*]} $FILENAME"
echo $MK_SPLASH
$MK_SPLASH
```

3. The mkspalshimg script need the argument for making splash image

   A. Pagesize – for nand flash memory. (MTD : pagesize , **SD/MMC:512** , **FTL(NAND_v8):512**)
   B. Image format – determinated based on the display output setting(**16 or 32**)
   C. Number of image – the number of image that include splash image. it can contains *maximum 10 images.*
   D. Target bitmap images - the image format have to bitmatp(bmp) and the resolution is same as you want to display.
   E. Target out file – the final splash partition image.

Examples.

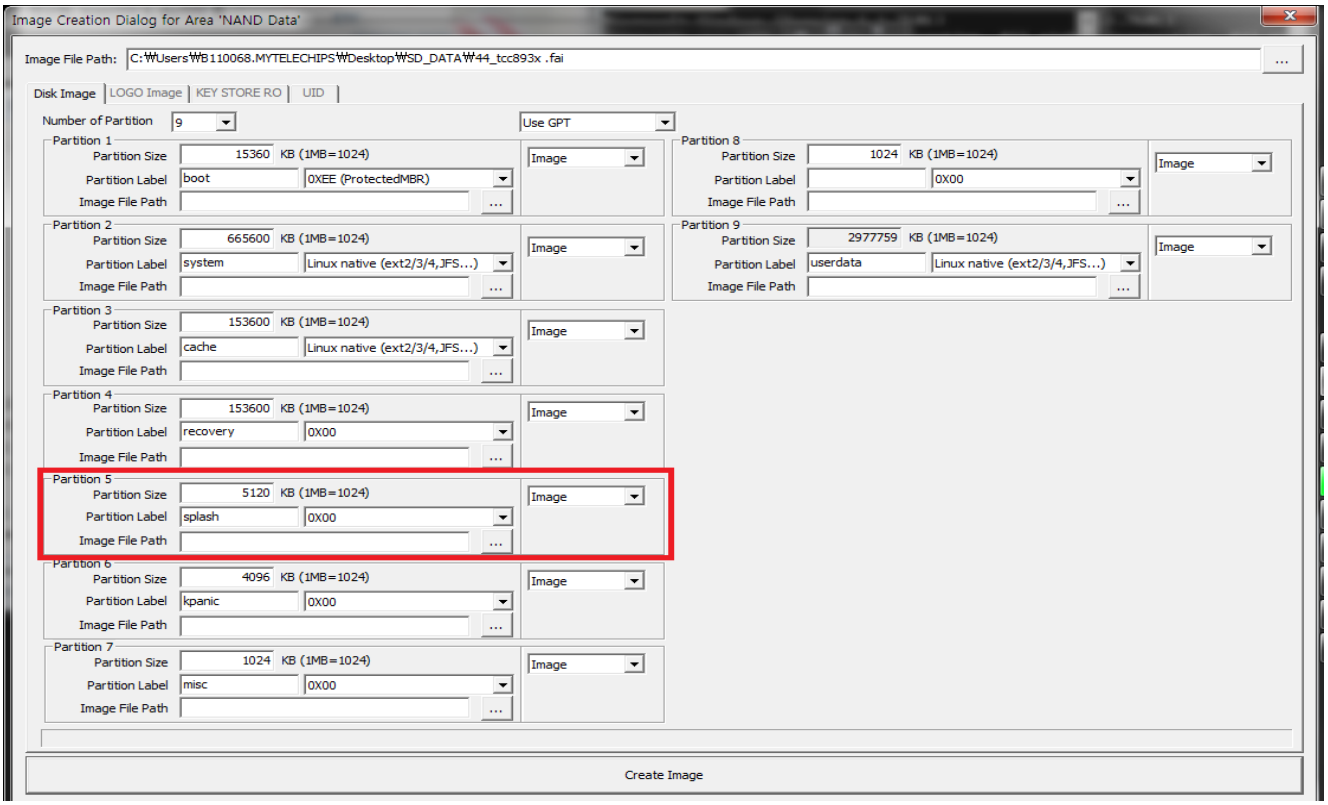
```
# when set to 32bit display output

$mksplashimg 8192 32 2 bootlogo.bmp test.bmp splash.img //nand
$mksplashimg 512 32 2 bootlogo.bmp test.bmp splash.img //EMMC

# when set to 16bit display output

$mksplashimg 8192 16 2 bootlogo.bmp test.bmp splash.img //nand
$mksplashimg 512 16 2 bootlogo.bmp test.bmp splash.img //EMMC
```

2.4.3 How to download splash image to splash partition

if you make splash image completely, then you can download the image to target splash partition.

2.4.3.1 Use SD/MMC or NAND V8 FTL only for Splash partition.

If you use the sd/mmc or nand version 8 driver for splash partition, you can download splash image using fwdn.
The red box is splash image partition. Add the splash image when you make partition using fwdn.



2.4.3.2 Fastboot mode download

The fastboot mode download is are same both of MTD and SD/MMC

```
$ fastboot flash splash {splsh image}
```

3 Change Kernel Logo

3.1 Set Host Environment

If a bootup logo is intended to be used in Android booting, change to .ppm format is required.
(For more information on .ppm extention, refer to http://www.fileinfo.com/extension/ppm.)

※ **This method is one which has been validated only when the 16 bit display output only.**

3.2 Make Logo Image

3.2.1 Make *.ppm File

Prepare 1024x600 bmp image in the following path. For easy understanding, telechip_logo.bmp image will be used. Then, the bmp image should be changed to *.ppm format as can be seen in the following procedure.

```
$ cd ~/mydroid/android/
$ cd kernel/drivers/video/logo

# bmptopnm telechips_logo.bmp | pnmtoplainpnm > image1024x600.ppm
# pnmquant -fs 223 image1024x600.ppm > image1024x600_256.ppm
# pnmnoraw image1024x600_256.ppm > logo_telechips_clut224.ppm
```

3.2.2 Make and Change *.c File

Make `logo_telechips_clut224.c` corresponding to `logo_telechips_clut224.ppm` file made in 3.2.1.

```
$ cd ~/mydroid/android/
$ cd kernel/drivers/video/logo
$ cp logo_linux_clut224.c logo_telechips_clut224.c
```

The file should be changed as follows.

```
$ cd ~/mydroid/android/
$ vi kernel/drivers/video/logo/logo_telechips_clut224.c

static unsigned char logo_telechips_clut224_data[] __initdata = {
…  /* omit */ …
static unsigned char logo_telechips_clut224_clut[] __initdata = {
…  /* omit */ …
const struct linux_logo logo_telechips_clut224 __initconst = {
        .type           = LINUX_LOGO_CLUT224,
        .width          = 1024,
        .height         = 600,
        .clutsize       = 160,
        .clut           = logo_telechips_clut224_clut,
        .data           = logo_telechips_clut224_data
  };
```

3.3 Change Linux Kernel Source

3.3.1 Change kernel/drivers/video/logo/Kconfig

In order to make options selected by the user in “make menuconfig”, move to the following path.

```
$ cd ~/mydroid/android/kernel/drivers/video/logo
$ vi Kconfig
```

Then, add the following procedure.

```
# add this code
    config LOGO_TELECHIPS_CLUT224
    bool "Telechips 224-color logo"
    default y
```

3.3.2 Change kernel/drivers/video/logo/Makefile

```
obj-$(CONFIG_LOGO)                          += logo.o
obj-$(CONFIG_LOGO_LINUX_MONO)               += logo_linux_mono.o
obj-$(CONFIG_LOGO_LINUX_VGA16)              += logo_linux_vga16.o
obj-$(CONFIG_LOGO_LINUX_CLUT224)            += logo_linux_clut224.o
obj-$(CONFIG_LOGO_TELECHIPS_CLUT224)        += logo_telechips_clut224.o
```

3.3.3 Change kernel/include/linux/linux_logo.h

```
$ cd ~/mydroid/android/
$ vi kernel/include/linux/linux_logo.h
```

```
extern const struct linux_logo logo_linux_clut224;
extern const struct linux_logo logo_telechips_clut224;
```

3.3.4 Change kernel/drivers/video/logo/logo.c

```
$ cd ~/mydroid/android/
$ cd kernel/ drivers/video/
$ vi logo.c
```

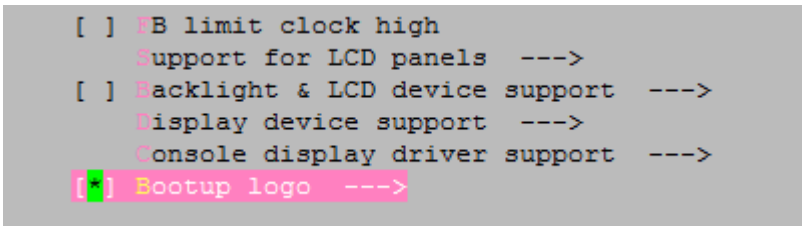
```
#ifdef CONFIG_LOGO_LINUX_CLUT224
      /* Generic Linux logo */
      logo = &logo_linux_clut224;
#endif

 /*   add    */
#ifdef CONFIG_LOGO_GPH_CLUT224
       /* Telechips Linux logo */
       logo = &logo_telechips_clut224;
#endif
```

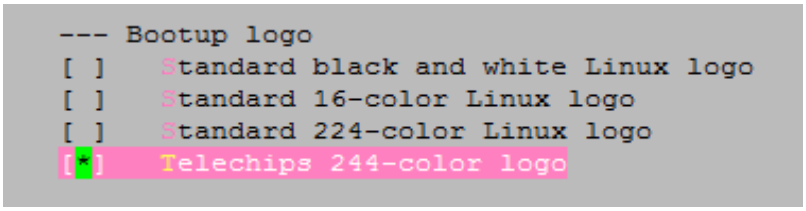
3.4 Kernel Configuration

By executing "make menuconfig", execute kernel configuration.

```
$ cd ~/mydroid/android/kernel
$ make menuconfig
```

Move to Device Drivers->Graphic support -> bootup logo and check as follows.





Then, execute save & exit and rebuild a kernel. If you execute booting with this kernel image, you can check the kernel logo which has been set previously.

4 Android init logo

4.1 Make Image (*.rle) for Android init Logo

In order to make an image(*.rle) with Android init logo used, move to the following path.

```
$ cd ~/mydroid/android/
$ cd device/telechips/common/
```

Prepare 1024x600 sized bmp image file. then, change it to *.img(raw) format with the following commands and then change the image into *.rle file.

4.1.1 When set to 16bit display output

```
$ convert -depth 8 initlogo_1024_600.bmp rgb:initlogo1024x600.raw
$ rgb2565 -rle <initlogo1024x600.raw> initlogo1024x600.rle
```

4.1.2 When set to 32bit display output

```
$ convert -depth 8 initlogo_1024_600.bmp rgb:initlogo1024x600.raw
$ rgbto888 -rle <initlogo1024x600.raw> initlogo1024x600.rle
```

For 32bit output , set as follows.

```
$ cd ~/mydroid/android/system/core/init
$ vi logo.c


#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <sys/types.h>

#include <linux/fb.h>
#include <linux/kd.h>

#include "log.h"

//#define INIT_RLE_32BIT
#define INIT_RLE_32BIT
```

4.2 Set Android init Logo Path

In Android, You may set a path of Android init logo.

```
$ cd ~/mydroid/android/
$ vi device/telechips/tcc893x/device.mk
```

You may select a certain file and this can be used for Android init logo.

```
PRODUCT_COPY_FILES += \
      device/telechips/common/initlogo1024x600.rle:root/initlogo.rle
```

4.3 Apply Android init Logo

According to 4.2, if you properly set a path of *.rle file which has been made in 4.1 you can use your wanted Android init logo.

5 Android animation logo

5.1 Change android animation logo by modify image which responsible for animation.

5.1.1 Control Android Animation Logo

The Android animation logo in this part has two image files and source code which handles control. Source code is composed of the part setting a path of image files and the part giving an effect to each image.

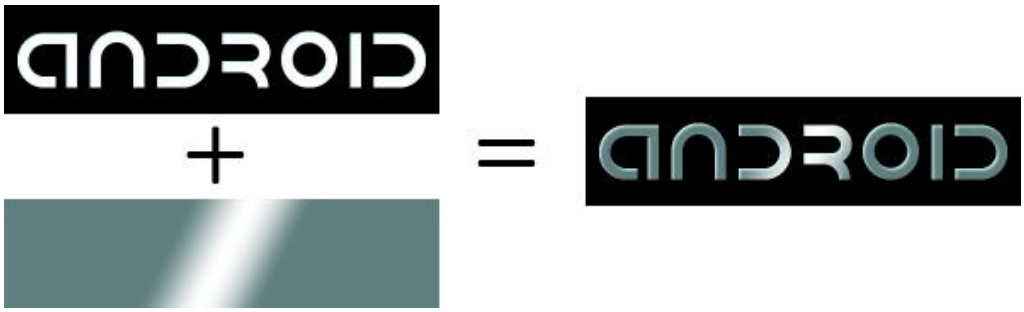
```
$ cd ~/mydroid/android/
$ vi frameworks/base/cmds/bootanimation/BootAnimation.cpp
```

```
bool BootAnimation::android()
{
    initTexture(&mAndroid[0], mAssets, "images/android-logo-mask.png");    // main mask
    initTexture(&mAndroid[1], mAssets, "images/android-logo-shine.png");
    // logo which flows behind mask

    … omit …
```

5.1.2 Image to Control Animation

```
$ cd ~/mydroid/android/
$ find . -name android-logo-*.png

./frameworks/base/core/res/assets/images/android-logo-mask.png
./frameworks/base/core/res/assets/images/android-logo-shine.png
```

The Android animation logo to be changed has two png files and are configured as follows.



As can be seen above, the upper one is android-logo-mask.png and the lower one is android-logo-shine.png.
In a Mask, letters are transparent. (Because it is a PNG file, alpha value exists.)

5.1.3 Apply Android Animation Logo

```
$ cd ~/mydroid/android/
$ cd frameworks/base/core/res/assets/images
```

As 5.2, if you make two images, save them to the above path and set a path of the images according to 5.1, you can use Android animation logo as you want.

5.2 Change android animation logo by bootanimation.zip

The android boot animation has been changed to enable easy replacement or customization. There is a file called bootanimation.zip stored in **out/targetproduct/tcc893x/system/media/** in the root file system.

```
$ cd ~/mydroid/android/
$ cd out/target/product/tcc893x/system/media
```

This file contains two things.

1) A description file (desc.txt) that outlines how the animation progresses, what images to use, image size etc.
2) Folder(s) that contain the images for the animation.

The basic structure of the bootanimation.zip file is as follows:

```
bootanimation.zip
   desc.txt
   android
   part1
```

android and part1 are directories that contain a series of images for example, in android there is:

```
   320_480_001.png
   320_480_002.png
     ··· omit ···
   320_480_023.png
   320_480_024.png
```

These images from the 'android' and 'part1' animations that are combined as outlined in the 'desc.txt' file to form the overall startup animation. The images are ordered by number and run in sequence.

```
   240 320 10
   p 1 0 android
   p 0 0 part1
```

The 'desc.txt' file outlines how the animation progresses and a sample is as follows:

command	description
240	width of the animation
320	height of the animation
10	desired fps of the animation
p	defines a animation part
1	how many times this animation. Part loop
0	defines a pause (max = 10)
android	the folder name where the animation images are
p	defined another animation part
0	defines that it loops forever (until android start)
0	defines a pause
part1	folder for the second animation part

< table 1. Descript of command in the “desc.txt’ >

6 Notes.

If there is no command (rgb2565 or rgbto888) following this guide, please be set up as follow.

6.1.1 When there is no ‘rgb2565’ command

```
$ cd ~/mydroid/android/build/tools/rgb2565
mm //make
```

6.1.2 When there is no ‘rgbto888’ command

```
$ cd ~/mydroid/android/device/telechips/common/splash
mm //make
```

When going thought the above process, it will be possible to find executable file corresponding to the path below.

```
$ cd ~/mydroid/android/out/host/linux-x86/bin
rgb2565        rgbto888              …
```