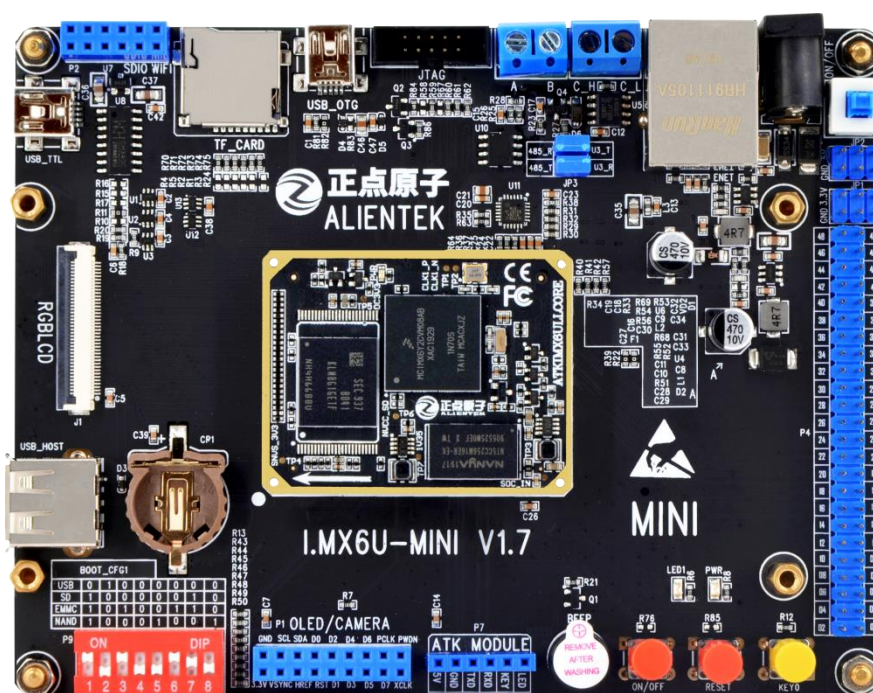
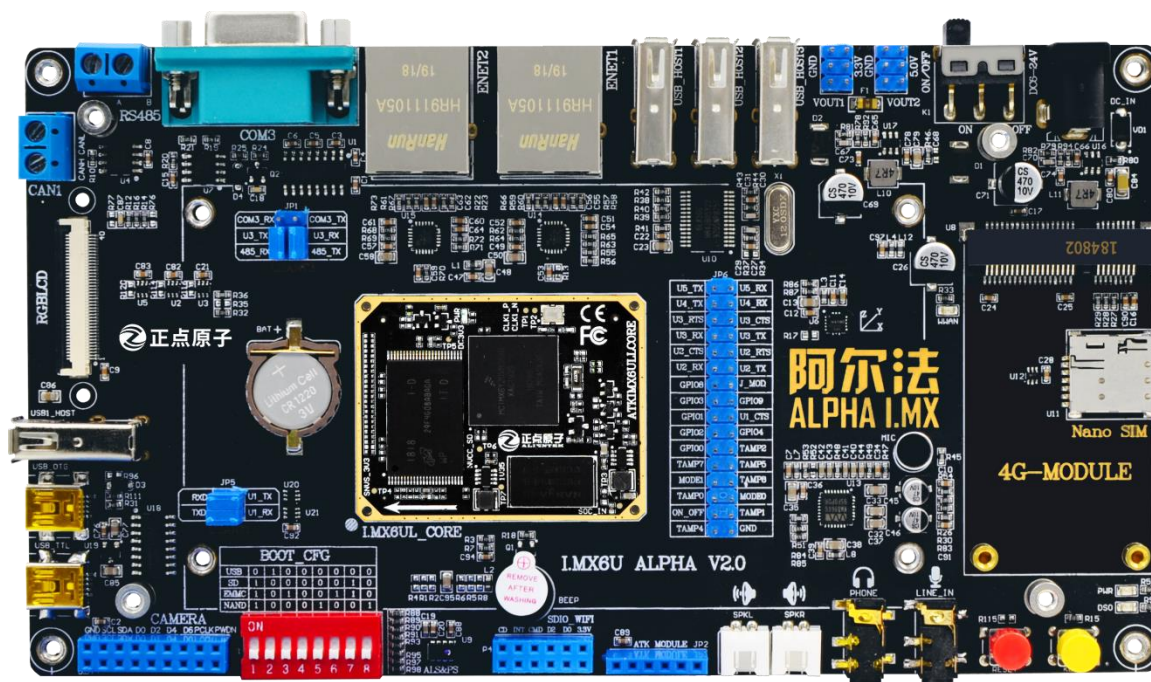


I.MX6U 修改开机进度条及 内核 logo 参考手册 V1.1





正点原子公司名称：广州市星翼电子科技有限公司

原子哥在线教学平台：www.yuanzige.com

开源电子网 / 论坛：<http://www.openedv.com/forum.php>

正点原子淘宝店铺：<https://openedv.taobao.com>

正点原子官方网站：www.alientek.com

正点原子 B 站视频：<https://space.bilibili.com/394620890>

电话：020-38271790 传真：020-36773971

请关注正点原子公众号，资料发布更新我们会通知。

请下载原子哥 APP，数千讲视频免费学习，更快更流畅。



扫码关注正点原子公众号



扫码下载“原子哥”APP

文档更新说明

版本	版本更新说明	负责人	校审	发布日期
V1.0	初稿: 添加修改内核 logo 和进度条的内容, 提供修改好的 psplash 源码	正点原子 linux 团队	正点原子 linux 团队	2021.01.04
V1.1	添加第三章 设置内核 logo 全屏显示。 添加 1.4 小节删除出厂系统进度条	正点原子 linux 团队	正点原子 linux 团队	2022.6.15

目录

第一章 修改进度条及其 logo	5
1.1 下载 psplash 源码.....	5
1.2 制作进度条和 logo	7
1.3 替换出厂系统进度条和 logo.....	11
1.4 删除出厂系统进度条.....	12
第二章 修改内核 logo	16
2.1 图片准备.....	16
2.2 源码准备.....	16
2.3 图片格式转换.....	17
2.4 配置内核 logo 显示	19
2.5 编译内核.....	21
第三章 设置内核 logo 全屏	22
3.1 图片准备.....	22
3.2 修改内核 logo 全屏	23
3.3 验证测试.....	24

第一章 修改进度条及其 logo

1.1 下载 psplash 源码

可以在网上下载 psplash 源码，正点原子也提供了修改好的 psplash 源码在光盘中。

光盘路径：开发板光盘 A-基础资料\1、例程源码\7、第三方库源码\psplash\psplash.tar.gz

在 Ubuntu 中创建一个 psplash 文件夹用于存放本文档的源码，将 psplash.tar.gz 源码放到此目录下解压。

```
tar -vxf psplash.tar.gz
```

```
alientek@ubuntu16:~/psplash$ ls
psplash.tar.gz
alientek@ubuntu16:~/psplash$ tar -vxf psplash.tar.gz
.gitignore
AUTHORS
COPYING
ChangeLog
Makefile.am
NEWS
README
base-images/
base-images/poky-logo.png
base-images/poky-logo2.png
configure.ac
make-image-header.sh
psplash-bar-img.h
psplash-colors.h
psplash-config.h
psplash-console.c
psplash-console.h
psplash-fb.c
psplash-fb.h
psplash-hand-img.h
psplash-poky-img.h
psplash-write.c
psplash.c
psplash.doap
psplash.h
radeon-font.h
```

图 1.1-1 解压 psplash.tar

解压完后会看到一个 make-image-header.sh 的脚本，等下会使用这个脚本来将图片变转化成头文件。

```
alientek@ubuntu16:~/psplash$ ls
AUTHORS      make-image-header.sh  psplash-console.c  psplash-hand-img.h
base-images  NEWS                  psplash-console.h  psplash-poky-img.h
ChangeLog    psplash-bar-img.h     psplash.doap        psplash.tar.gz
configure.ac psplash.c              psplash-fb.c        psplash-write.c
COPYING      psplash-colors.h      psplash-fb.h        radeon-font.h
Makefile.am  psplash-config.h      psplash.h           README
alientek@ubuntu16:~/psplash$
```

图 1.1-2 查看 psplash 里的文件

在使用脚本之前，需要安装库，否则会报错。

```
sudo apt-get install libgdk-pixbuf2.0-dev
```

```
alientek@ubuntu16:~/psplash$ sudo apt-get install libgdk-pixbuf2.0-dev
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列【新】软件包将被安装:
  libgdk-pixbuf2.0-dev
升级了 0 个软件包, 新安装了 1 个软件包, 要卸载 0 个软件包, 有 99 个软件包未被升级。
需要下载 0 B/44.3 kB 的归档。
解压缩后会消耗 342 kB 的额外空间。
正在选中未选择的软件包 libgdk-pixbuf2.0-dev。
(正在读取数据库 ... 系统当前共安装有 232975 个文件和目录。)
正准备解包 .../libgdk-pixbuf2.0-dev_2.32.2-1ubuntu1.6_amd64.deb ...
正在解包 libgdk-pixbuf2.0-dev (2.32.2-1ubuntu1.6) ...
正在处理用于 man-db (2.7.5-1) 的触发器 ...
正在设置 libgdk-pixbuf2.0-dev (2.32.2-1ubuntu1.6) ...
alientek@ubuntu16:~/psplash$
```

图 1.1-3 安装库

将光盘提供的如下图片复制到 psplash 源码中。

光盘路径: 开发板光盘 A-基础资料\1、例程源码\7、第三方库源码\psplash\logo-alientek.png
主图片 logo-alientek.png, 即正点原子的 logo。



图 1.1-4 logo-alientek.png

进度条的外框 logo-psplash-bar.png。

光盘路径: 开发板光盘 A-基础资料\1、例程源码\7、第三方库源码\psplash\logo-psplash-bar.png



图 1.1-5 logo-psplash-bar.png

将这两张图片拷贝到 psplash 源码目录中, 如下图所示。

```
alientek@ubuntu16:~/psplash$ ls
AUTHORS          logo-alientek.png  psplash-bar-img.h  psplash-console.h  psplash-hand-img.h  README
base-images      logo-psplash-bar.png psplash.c          psplash.doap       psplash-poky-img.h
ChangeLog        Makefile.am        psplash-colors.h  psplash-fb.c       psplash-write.c
configure.ac     make-image-header.sh psplash-config.h  psplash-fb.h       radeon-font.h
COPYING          NEWS               psplash-console.c psplash.h
```

图 1.1-6 将图片放入 psplash 源码

1.2 制作进度条和 logo

在 psplash 源码中执行以下指令来生成 logo-alientek-img.h

```
./make-image-header.sh logo-alientek.png POKY
```

```
ls
```

```
alientek@ubuntu16:~/psplash$ ./make-image-header.sh logo-alientek.png POKY
alientek@ubuntu16:~/psplash$ ls
AUTHORS          logo-alientek.png      psplash.c          psplash-fb.c        psplash-write.c
base-images      logo-psplash-bar.png  psplash-colors.h   psplash-fb.h        radeon-font.h
ChangeLog        Makefile.am           psplash-config.h   psplash.h           README
configure.ac     make-image-header.sh  psplash-console.c  psplash-hand-img.h
COPYING          NEWS                 psplash-console.h  psplash-poky-img.h
logo-alientek-img.h psplash-bar-img.h     psplash.doap       psplash.tar.gz
alientek@ubuntu16:~/psplash$
```

图 1.2-1 生成 logo-alientek-img.h

POKY 参数只是一个变量，它会传入到 logo-alientek-img.h 文件里面，可以打开查看。

```
vi logo-alientek-img.h
```

```
alientek@ubuntu16: ~/psplash
1 /* GdkPixbuf RGBA C-Source image dump 1-byte-run-length-encoded */
2
3 #define POKY_IMG_ROWSTRIDE (1708)
4 #define POKY_IMG_WIDTH (427)
5 #define POKY_IMG_HEIGHT (214)
6 #define POKY_IMG_BYTES_PER_PIXEL (4) /* 3:RGB, 4:RGBA */
7 #define POKY_IMG_RLE_PIXEL_DATA ((uint8*) \
8   "\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377" \
9   "\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377" \
10  "\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377" \
11  "\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377" \
12  "\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377" \
13  "\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377" \
14  "\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377\377\0\0\0\377" \
```

篇幅过长
以下省略

图 1.2-2 打开 logo-alientek-img.h

可以看到它必须是要加 POKY 参数的，因为其他地方要用到这样的宏定义。

返回到 psplash 源码下，执行以下指令生成 psplash-bar-img.h

```
./make-image-header.sh logo-psplash-bar.png BAR
```

```
ls
```

```
alientek@ubuntu16:~/psplash$ ./make-image-header.sh logo-psplash-bar.png BAR
alientek@ubuntu16:~/psplash$ ls
AUTHORS          logo-alientek.png      psplash-bar-img.h  psplash.doap        psplash.tar.gz
base-images      logo-psplash-bar.png  psplash.c          psplash-fb.c        psplash-write.c
ChangeLog        Makefile.am           psplash-colors.h   psplash-fb.h        radeon-font.h
configure.ac     make-image-header.sh  psplash-config.h   psplash.h           README
COPYING          NEWS                 psplash-console.c  psplash-hand-img.h
logo-alientek-img.h psplash-bar-img.h     psplash-console.h  psplash-poky-img.h
alientek@ubuntu16:~/psplash$
```

图 1.2-3 生成 psplash-bar-img.h

同样我们打开 logo-psplash-bar-img.h 这个文件。它必须是加 BAR 参数的，因为其他地方会用到这样的宏定义。

```
vi logo-psplash-bar-img.h
```

```

alientek@ubuntu16: ~/psplash
1 /* GdkPixbuf RGBA C-Source image dump 1-byte-run-length-encoded */
2
3 #define BAR_IMG_ROWSTRIDE (920)
4 #define BAR_IMG_WIDTH (230)
5 #define BAR_IMG_HEIGHT (28)
6 #define BAR_IMG_BYTES_PER_PIXEL (4) /* 3:RGB, 4:RGBA */
7 #define BAR_IMG_RLE_PIXEL_DATA ((uint8*) \
8  "\377\376\377\0\377\377\376\377\0\377\377\376\377\0\377\321\376\377\0\377" \
9  "\377\0\0\0\0\343\0\0\0\0\204\376\377\0\377\377\0\0\0\0\343\0\0\0\0\204" \
10 "\376\377\0\377\377\0\0\0\0\343\0\0\0\0\204\376\377\0\377\377\0\0\0\0" \
11 "\343\0\0\0\0\204\376\377\0\377\377\0\0\0\0\343\0\0\0\0\204\376\377\0" \
12 "\377\377\0\0\0\0\343\0\0\0\0\204\376\377\0\377\377\0\0\0\0\343\0\0\0" \
13 "\0\204\376\377\0\377\377\0\0\0\0\343\0\0\0\0\204\376\377\0\377\377\0" \
14 "\0\0\0\343\0\0\0\0\204\376\377\0\377\377\0\0\0\0\343\0\0\0\0\204\376" \

```

篇幅过长
以下省略

图 1.2-4 查看 logo-psplash-bar-img.h

打开 psplash.c, 将里面的 `#include "psplash-poky-img.h"` 修改成 `#include "logo-alientek-img.h"`。将里面的 `#include "psplash-bar-img.h"` 修改成 `#include "logo-psplash-bar-img.h"`。修改完后如下图所示。

vi psplash.c

```

alientek@ubuntu16: ~/psplash
1 /*
2  * psplash - a lightweight framebuffer splashscreen for embedded devices.
3  *
4  * Copyright (c) 2006 Matthew Allum <mallum@o-hand.com>
5  *
6  * Parts of this file ( fifo handling ) based on 'usplash' copyright
7  * Matthew Garret.
8  *
9  * This program is free software; you can redistribute it and/or modify
10  * it under the terms of the GNU General Public License as published by
11  * the Free Software Foundation; either version 2, or (at your option)
12  * any later version.
13  *
14  * This program is distributed in the hope that it will be useful,
15  * but WITHOUT ANY WARRANTY; without even the implied warranty of
16  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
17  * GNU General Public License for more details.
18  *
19  */
20
21 #include "psplash.h"
22 #include "psplash-config.h"
23 #include "psplash-colors.h"
24 // #include "psplash-poky-img.h"
25 // #include "psplash-bar-img.h"
26 #include "logo-alientek-img.h"
27 #include "logo-psplash-bar-img.h"
28 #include "radeon-font.h"
29
30 #define SPLIT_LINE_POS(fb) \
31     ( (fb)->height \
32     - (( PSPLASH_IMG_SPLIT_DENOMINATOR \

```

图 1.2-5 修改 psplash.c

修改完成后保存退出 psplash.c, 返回到源码目录下。

制作 autogen.sh 脚本, 用于生成 Makefile。

vi autogen.sh

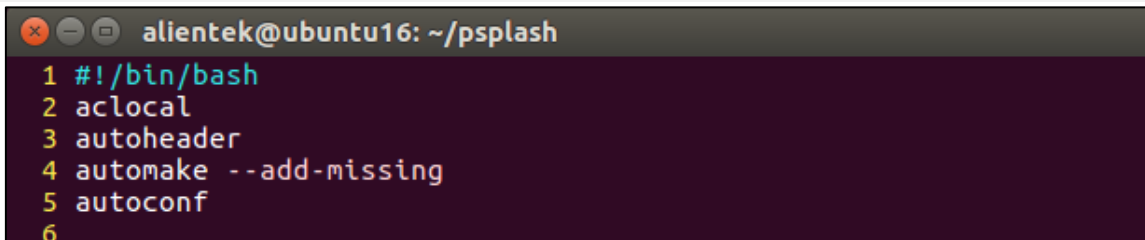
脚本内容如下。

```

#!/bin/bash
aclocal
autoheader
automake --add-missing

```

```
autoconf
```



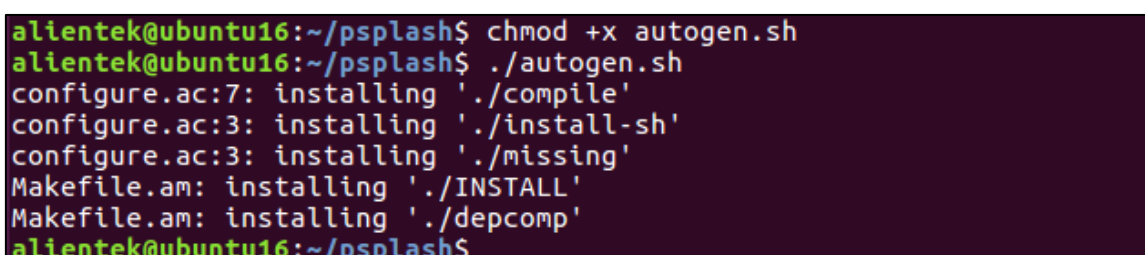
```
alientek@ubuntu16: ~/psplash
1 #!/bin/bash
2 aclocal
3 autoheader
4 automake --add-missing
5 autoconf
6
```

图 1.2-6 autogen.sh 脚本内容

制作完脚本后, 赋予它可执行权限, 再执行这个脚本。

```
chmod +x autogen.sh
```

```
./autogen.sh
```



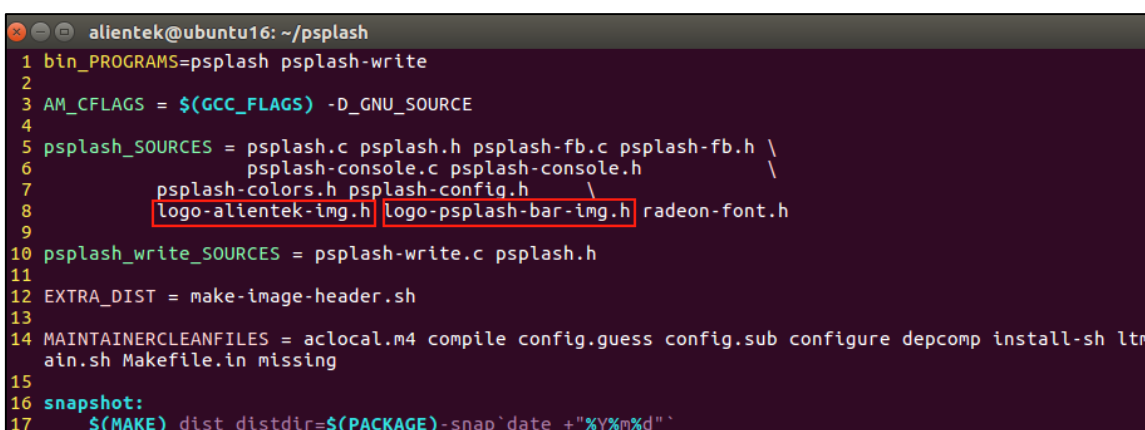
```
alientek@ubuntu16:~/psplash$ chmod +x autogen.sh
alientek@ubuntu16:~/psplash$ ./autogen.sh
configure.ac:7: installing './compile'
configure.ac:3: installing './install-sh'
configure.ac:3: installing './missing'
Makefile.am: installing './INSTALL'
Makefile.am: installing './depcomp'
alientek@ubuntu16:~/psplash$
```

图 1.2-7 赋予脚本可执行权限并执行脚本

修改 Makefile.am, 将 psplash-poky-img.h 修改成 logo-alientek-img.h, 将 psplash-bar-img.h 修改成 logo-psplash-bar-img.h

```
vi Makefile.am
```

修改完后的 Makefile.am 如下图所示。



```
alientek@ubuntu16: ~/psplash
1 bin_PROGRAMS=psplash psplash-write
2
3 AM_CFLAGS = $(GCC_FLAGS) -D_GNU_SOURCE
4
5 psplash_SOURCES = psplash.c psplash.h psplash-fb.c psplash-fb.h \
6                  psplash-console.c psplash-console.h \
7                  psplash-colors.h psplash-config.h \
8                  logo-alientek-img.h logo-psplash-bar-img.h radeon-font.h
9
10 psplash_write_SOURCES = psplash-write.c psplash.h
11
12 EXTRA_DIST = make-image-header.sh
13
14 MAINTAINERCLEANFILES = aclocal.m4 compile config.guess config.sub configure depcomp install-sh ltm
15 ain.sh Makefile.in missing
16
17 snapshot:
18     $(MAKE) dist distdir=$(PACKAGE)-snap`date +%Y%m%d`
```

图 1.2-8 修改 Makefile.am

修改完 Makefile.am 后保存退出, 回到源码目录下。

执行以下指令, 使能环境变量。

```
source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
```

```

alientek@ubuntu16:~/psplash$
alientek@ubuntu16:~/psplash$ source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
alientek@ubuntu16:~/psplash$

```

图 1.2-9 使能环境变量

环境变量使能后, 就可以执行以下指令配置交叉编译工具链。

```
./configure --host=arm-linux
```

```

alientek@ubuntu16:~/psplash$
alientek@ubuntu16:~/psplash$ ./configure --host=arm-linux
configure: loading site script /opt/fsl-imx-x11/4.1.15-2.1.0/site-config-cortexa7hf-neon-poky-linux-gnueabi
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for arm-linux-strip... arm-poky-linux-gnueabi-strip
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for style of include used by make... GNU
checking for arm-linux-gcc... arm-poky-linux-gnueabi-gcc -march=armv7ve -mfloat-abi=hard -mcpu=cortex-a7 --sysroot=/opt/fsl-imx-x11/4.1.15-2.1.0/sysroots/cortexa7hf-neon-poky-linux-gnueabi
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...

```

篇幅过长
以下省略

图 1.2-10 配置交叉编译工具链

执行 make 指令进行编译。

```
make
```

```

alientek@ubuntu16:~/psplash$ make
cd . && /bin/bash /home/alientek/psplash/missing automake-1.15 --gnu Makefile
perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
    LANGUAGE = "zh_CN:zh",
    LC_ALL = (unset),
    LANG = "zh_CN.UTF-8"
are supported and installed on your system.
perl: warning: Falling back to the standard locale ("C").
perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
    LANGUAGE = "zh_CN:zh",
    LC_ALL = (unset),
    LANG = "zh_CN.UTF-8"
are supported and installed on your system.
perl: warning: Falling back to the standard locale ("C").
Unescaped left brace in regex is deprecated, passed through in regex; marked by <-- HERE in m/\${ <--

```

篇幅过长
以下省略

图 1.2-11 编译源码

编译完成后会在源码目录下生成 psplash 和 psplash-write 文件。

```

alientek@ubuntu16:~/psplash$
alientek@ubuntu16:~/psplash$ ls
aclocal.m4      config.status      logo-psplash-bar.png  psplash-colors.h    psplash-hand-img.h
AUTHORS         config             Makefile              psplash-config.h    psplash.o
autogen.sh      configure.ac        Makefile.am           psplash-console.c   psplash-poky-img.h
autom4te.cache  COPYING            Makefile.in           psplash-console.h   psplash.tar.gz
base-images     depcomp            make-image-header.sh  psplash-console.o   psplash-write.c
ChangeLog       INSTALL           missing               psplash.doap        psplash-write.o
compile         install-sh         NEWS                 psplash.fb.c        psplash-font.h
config.h        logo-alientek.png  psplash-bar-img.h     psplash.fb.o        README
config.h.in     logo-psplash-bar-img.h psplash.c             psplash.h            stamp-h1
alientek@ubuntu16:~/psplash$

```

图 1.2-12 生成的 psplash 和 psplash-write 文件

1.3 替换出厂系统进度条和 logo

将编译生成的 psplash 和 psplash-write 文件拷贝到开发板出厂文件系统/usr/bin 目录下。

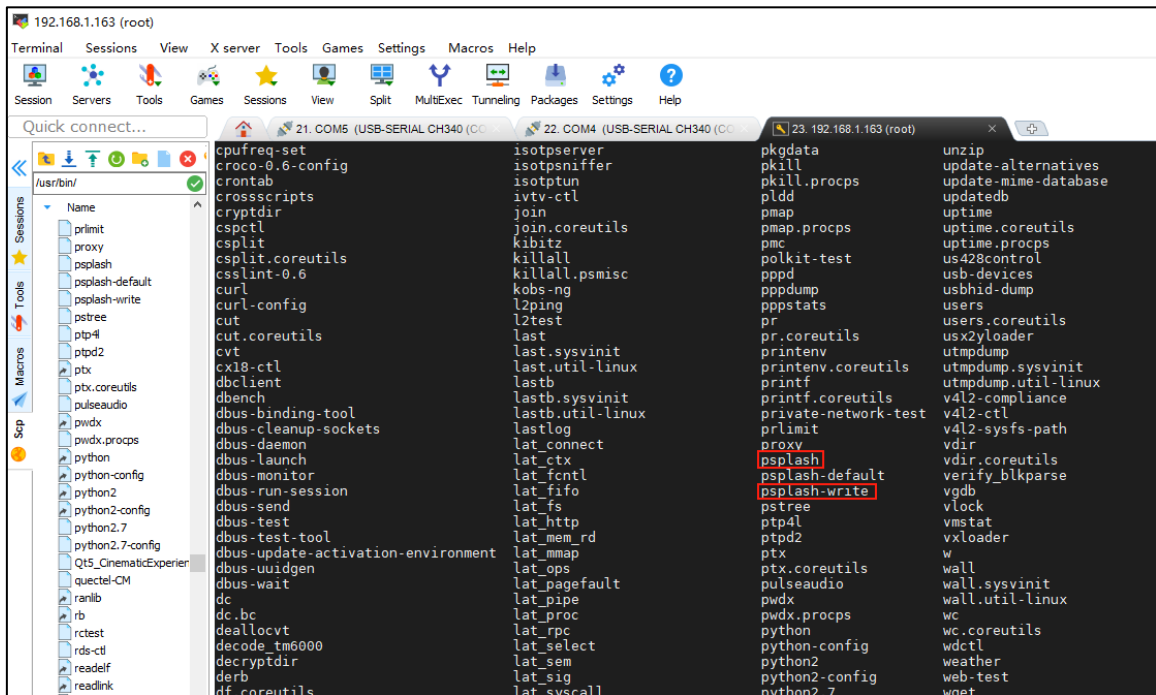


图 1.3-1 拷贝 psplash 文件到开发板/usr/bin

在开发板/etc/init.d 中已存在 psplash.sh 脚本。

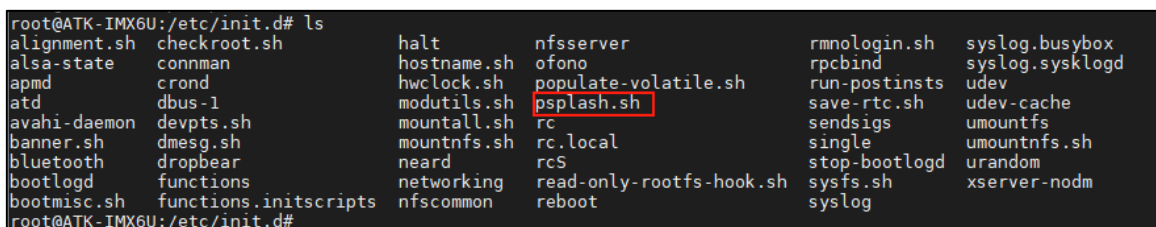


图 1.3-2 开发板/etc/init.d 中的 psplash.sh 脚本

在开发板/etc/rcS.d 目录下已存在 psplash.sh 的链接，用于开机启动。

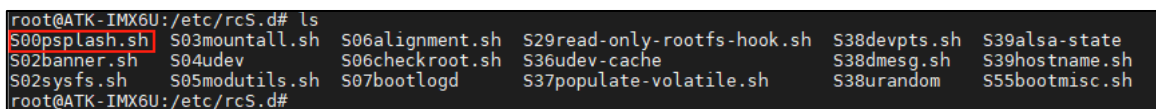


图 1.3-3 开发板/etc/rcS.d 下的 psplash.sh 链接

完成后启动开发板，可以看到如下界面。（注：作者这里使用手机拍照，所以拍下来会有一些阴影。）



图 1.3-4 开机启动画面

1.4 删除出厂系统进度条

打开出厂系统里的/etc/init.d/psplash.sh 文件, 添加终止 psplash.sh 的指令。如下所示, 添加 exit 0

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          psplash
# Required-Start:
# Required-Stop:
# Default-Start:     S
# Default-Stop:
### END INIT INFO

exit 0

read CMDLINE < /proc/cmdline
for x in $CMDLINE; do
    case $x in
        psplash=false)
            echo "Boot splashscreen disabled"
            exit 0;
            ;;
    esac
done

export TMPDIR=/mnt/.psplash
mount tmpfs -t tmpfs $TMPDIR -o,size=40k

rotation=0
if [ -e /etc/rotation ]; then
    read rotation < /etc/rotation
fi
```

图 1.4-1 终止 psplash.sh

注释掉出厂系统里的/etc/inittab 里的打印映射。

```
# /etc/inittab: init(8) configuration.
# $Id: inittab,v 1.91 2002/01/25 13:35:21 miquels Exp $

# The default runlevel.
id:5:initdefault:

# Boot-time system configuration/initialization script.
# This is run first except when booting in emergency (-b) mode.
si::sysinit:/etc/init.d/rcS

# What to do in single-user mode.
~~:S:wait:/sbin/sulogin

# /etc/init.d executes the S and K scripts upon change
# of runlevel.
#
# Runlevel 0 is halt.
# Runlevel 1 is single-user.
# Runlevels 2-5 are multi-user.
# Runlevel 6 is reboot.

l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
# Normally not reached, but fallthrough in case of emergency.
z6:6:respawn:/sbin/sulogin
#mxc0:12345:respawn:/bin/start_getty 115200 ttymxc0
mxc0:12345:respawn:/sbin/getty -l /bin/autologin -n -L 115200 ttymxc0
# /sbin/getty invocations for the runlevels.
#
# The "id" field MUST be the same as the last
# characters of the device (after "tty").
#
# Format:
# <id>:<runlevels>:<action>:<process>
#
#l1:12345:respawn:/sbin/getty 38400 ttyl
```

图 1.4-2 注释掉打印映射

注释掉/etc/init.d/banner.sh 里的打印信息。

```
#!/bin/sh
### BEGIN INIT INFO
# Provides: banner
# Required-Start:
# Required-Stop:
# Default-Start:    S
# Default-Stop:
### END INIT INFO

if [ ! -e /dev/tty ]; then
    /bin/mknod -m 0666 /dev/tty c 5 0
fi

if ( > /dev/tty0 ) 2>/dev/null; then
    vtmaster=/dev/tty0
elif ( > /dev/vc/0 ) 2>/dev/null; then
    vtmaster=/dev/vc/0
elif ( > /dev/console ) 2>/dev/null; then
    vtmaster=/dev/console
else
    vtmaster=/dev/null
fi
#echo > $vtmaster
#echo "Please wait: booting..." > $vtmaster
~
~
```

图 1.4-3 注释掉打印信息

保存好修改的文件后，重启开发板即可，不再显示进度条。

第二章 修改内核 logo

修改完出厂系统的进度条后,我们发现开机后仍后正点原子的出厂 logo 存在,这并不是刚刚的修改失败,因为这个是内核的 logo,我们刚刚修改的是进度条界面的 logo。接下来我们就来修改内核的 logo。

2.1 图片准备

图片要求:

格式: BMP 或者 PNG 格式。如果是别的格式的图片,可以通过 Windows 自带的画图软件来另存为 BMP 或者 PNG 格式。

分辨率: 不能高于屏幕 LCD 的分辨率。可以用画图软件,可以对图片右键选择属性,在详细信息中可以看到图片的分辨率。

这里以光盘源码中的 logo-linux.png 为例。

光盘路径: 开发板光盘 A-基础资料\1、例程源码\7、第三方库源码\psplash

2.2 源码准备

本文档以出厂系统为例。在 Ubuntu 中新建一个 linux-alientek 的文件夹用于存放出厂的内核源码。示例:将光盘中的出厂内核源码 `linux-imx-4.1.15-2.1.0-gb78e551-v1.4.tar.xz` 拷贝到 linux-alientek 目录中。请用网盘上最新的出厂内核源码编译。

光盘路径: 开发板光盘 A-基础资料\1、例程源码\3、正点原子 Uboot 和 Linux 出厂源码解压内核源码。

```
tar vxvf 内核源码包
```

```
ls
```

```
./net/decnet/netfilter/Makefile
./net/decnet/netfilter/dn_rtmsg.c
./net/decnet/dn_table.c
./net/decnet/dn_nsp_out.c
./Kbuild
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ ls
arch      Documentation  ipc            MAINTAINERS   samples       virt
block     drivers        Kbuild        Makefile      scripts
build.sh  firmware      Kconfig       mm            security
COPYING   fs            kernel        net           sound
CREDITS   include       lib           README        tools
crypto    init          linux-imx-4.1.15-2.1.0-gb78e551-v1.4.tar.xz  REPORTING-BUGS  usr
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
```

图 2.2-1 解压完的内核源码

在文件管理器中进入内核源码的 `drivers/video/logo` 目录下,可以看到启动内核时的 logo。这里可以看到正点原子出厂系统内核启动的 logo。

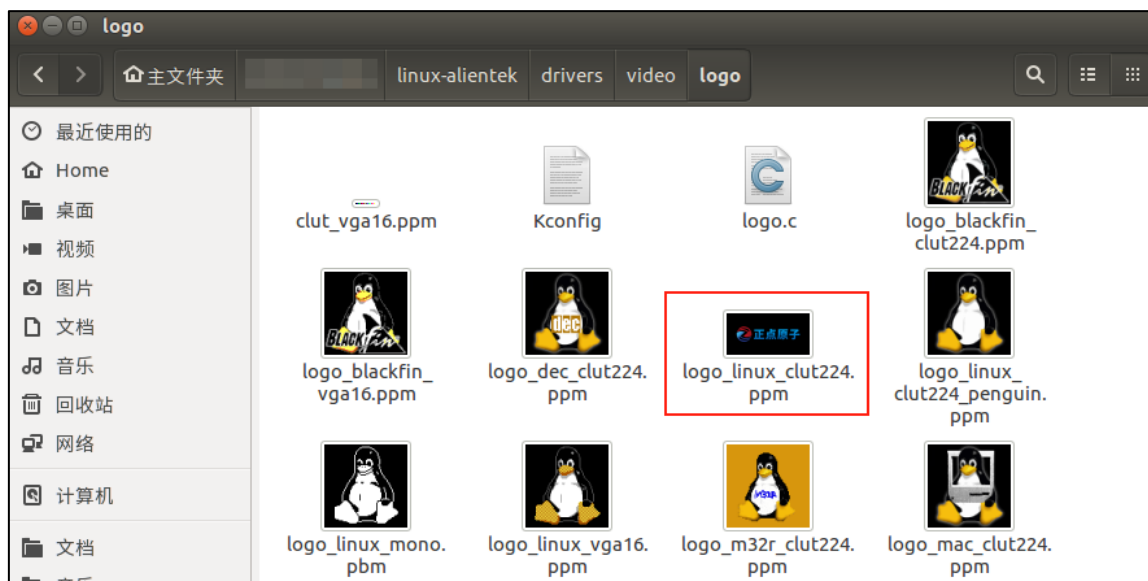


图 2.2-2 内核源码里的 logo

将我们准备好的 logo-linux.png 图片拷贝到这个目录中。拷贝完后的目录如下所示。

```
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ cd drivers/video/logo/
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ ls
clut_vga16.ppm      logo_linux_clut224_penguin.ppm  logo_mac_clut224.ppm  logo_superh_mono.pbm
Kconfig            logo_linux_clut224.ppm          logo_parisc_clut224.ppm  logo_superh_vga16.ppm
logo_blackfin_clut224.ppm  logo_linux_mono.pbm            logo_sgi_clut224.ppm    Makefile
logo_blackfin_vga16.ppm  logo-linux.png                  logo_spe_clut224.ppm
logo.c              logo_linux_vga16.ppm           logo_sun_clut224.ppm
logo_dec_clut224.ppm  logo_m32r_clut224.ppm          logo_superh_clut224.ppm
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$
```

图 2.2-3 已将准备的 logo 拷贝到内核目录中

将正点原子出厂系统 logo 重命名为 logo_linux_clut224_V1.ppm。因为我们等下要替换掉 logo_linux_clut224.ppm。

```
mv logo_linux_clut224.ppm logo_linux_clut224_V1.ppm
```

```
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ mv logo_linux_clut224.ppm logo_linux_clut224_V1.ppm
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ ls
clut_vga16.ppm      logo_linux_clut224_penguin.ppm  logo_mac_clut224.ppm  logo_superh_mono.pbm
Kconfig            logo_linux_clut224_V1.ppm       logo_parisc_clut224.ppm  logo_superh_vga16.ppm
logo_blackfin_clut224.ppm  logo_linux_mono.pbm            logo_sgi_clut224.ppm    Makefile
logo_blackfin_vga16.ppm  logo-linux.png                  logo_spe_clut224.ppm
logo.c              logo_linux_vga16.ppm           logo_sun_clut224.ppm
logo_dec_clut224.ppm  logo_m32r_clut224.ppm          logo_superh_clut224.ppm
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$
```

图 2.2-4 重命名原来的 logo

2.3 图片格式转换

我们需要把 png 格式的图片转换成 ppm 格式。

安装图片转换工具 netpdm, 执行以下命令。

```
sudo apt-get install netpbm
```

```

alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ sudo apt-get install netpbm
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列【新】软件包将被安装:
  netpbm
升级了 0 个软件包, 新安装了 1 个软件包, 要卸载 0 个软件包, 有 99 个软件包未被升级。
需要下载 997 kB 的归档。
解压后会消耗 4,330 kB 的额外空间。
获取:1 http://mirrors.aliyun.com/ubuntu xenial/main amd64 netpbm amd64 2:10.0-15.3 [997 kB]
已下载 997 kB, 耗时 0 秒 (2,062 kB/s)
正在选中未选择的软件包 netpbm。
(正在读取数据库 ... 系统当前共安装有 232522 个文件和目录。)
正准备解包 .../netpbm_2%3a10.0-15.3_amd64.deb ...
正在解包 netpbm (2:10.0-15.3) ...
正在处理用于 man-db (2.7.5-1) 的触发器 ...
正在设置 netpbm (2:10.0-15.3) ...
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$

```

图 2.3-1 安装 netpbm 工具

使用 netpbm 工具转换图片, 执行以下命令。

```
pngtopnm logo-linux.png > logo-linux.pnm
```

```
pnmquant 224 logo-linux.pnm > logo-linux224.pnm
```

```
pnmtoplainpnm logo-linux224.pnm > logo_linux_clut224.ppm
```

```

alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ pngtopnm logo-linux.png > logo-linux.pnm
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ pnmquant 224 logo-linux.pnm > logo-linux224.pnm
pnmcolormap: making histogram...
pnmcolormap: 256 colors found
pnmcolormap: choosing 224 colors...
pnmremap: 224 colors found in colormap
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ pnmtoplainpnm logo-linux224.pnm > logo_linux_clut224.ppm
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$

```

图 2.3-2 转换图片格式

分析:

```
pngtopnm logo-linux.png > logo-linux.pnm
```

这一步是将准备好的 logo-linux.png 转换成 logo-linux.pnm, 即转换成 pnm 格式。

```
pnmquant 224 logo-linux.pnm > logo-linux224.pnm
```

这一步是将 logo-linux.pnm 图片的最大颜色数量改为 224。

注意, 这里前后名字不能一样, 可以理解为要另存为。否则会报错, 如下图所示。

```

alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$ pnmquant 224 logo-linux.pnm > logo-linux.pnm
pnmcolormap: EOF / read error reading magic number 报错信息
pnmcolormap failed, rc=256
alientek@ubuntu16:~/alientek-alpha/linux-alientek/drivers/video/logo$

```

图 2.3-3 报错信息

```
pnmtoplainpnm logo-linux224.pnm > logo_linux_clut224.ppm
```

这一步是将 pnm 格式图片转换成 ppm 格式, 顺便将其重命名为 logo_linux_clut224.ppm, 我们是在内核源码的 drivers/video/logo 目录下直接进行的, 所以此步就直接替换内核原来的 logo (之前的 logo_linux_clut224.ppm 我们已经重命名了 logo_linux_clut224_V1.ppm)

如果不是在 drivers/video/logo 目录下进行的转换, 需要将转换好的 logo_linux_clut224.ppm 拷贝到 drivers/video/logo 目录下替换掉原先的 logo_linux_clut224.ppm。

2.4 配置内核 logo 显示

出厂系统的内核已经配置好 logo 显示了, 可以跳过此步。但为了文档的完整性, 这里还是要说明一下。(注: 这里使用的还是出厂的内核源码)

执行以下命令安装编译内核要用到的库和显示 menuconfig 菜单时要用到的库。

```
sudo apt-get install lzop
sudo apt-get install libncurses*
```

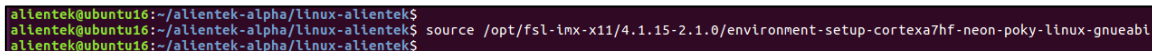


```
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ sudo apt-get install lzop
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
lzop 已经是最新版 (1.03-3.2)。
升级了 0 个软件包, 新安装了 0 个软件包, 要卸载 0 个软件包, 有 99 个软件包未被升级。
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ sudo apt-get install libncurses*
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
注意, 根据Glob 'libncurses*' 选中了 'libncursesw5-dbg'
注意, 根据Glob 'libncurses*' 选中了 'libncursesw5-dev'
```

图 2.4-1 安装所需要用到的库

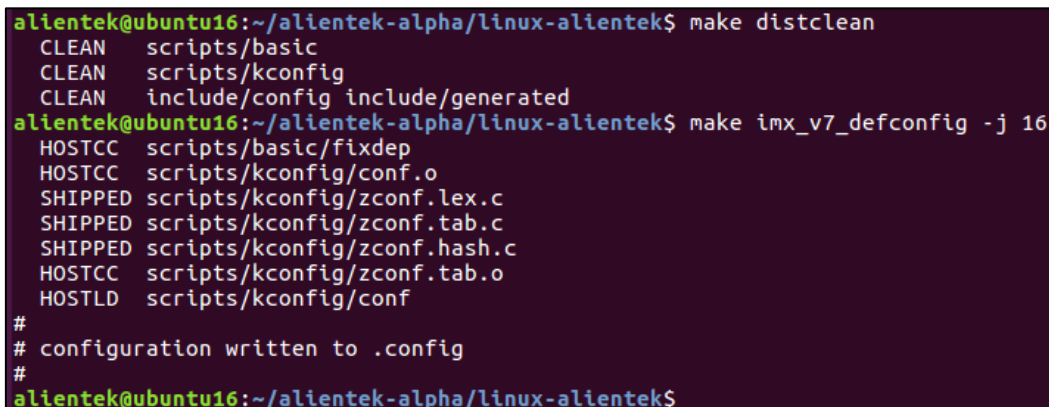
返回到内核源码的顶层目录下, 执行以下指令使能交叉编译器, 清理并配置内核。(注意: 如果没有安装文档对应的交叉编译器, 请按照《I.MX6U用户快速体验》4.2小节搭建交叉编译环境来安装)

```
source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
make distclean
make imx_v7_defconfig -j 16
```



```
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ make distclean
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ make imx_v7_defconfig -j 16
```

图 2.4-2 使能交叉编译器



```
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ make distclean
CLEAN    scripts/basic
CLEAN    scripts/kconfig
CLEAN    include/config include/generated
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ make imx_v7_defconfig -j 16
HOSTCC   scripts/basic/fixdep
HOSTCC   scripts/kconfig/conf.o
SHIPPED  scripts/kconfig/zconf.lex.c
SHIPPED  scripts/kconfig/zconf.tab.c
SHIPPED  scripts/kconfig/zconf.hash.c
HOSTCC   scripts/kconfig/zconf.tab.o
HOSTLD   scripts/kconfig/conf
#
# configuration written to .config
#
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
```

图 2.4-3 清理并配置内核

执行以下命令打开配置菜单。

```
make menuconfig
```

```
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ ls
arch      CREDITS  firmware ipc      lib      mm      net      samples  tools
block     crypto   fs       Kbuild  linux-imx-4.1.15-2.1.0-gb78e551-v1.4.tar.xz  README  scripts  usr
build.sh  Documentation include Kconfig MAINTAINERS  REPORTING-BUGS  security  virt
COPYING   drivers  init     kernel  Makefile
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ make menuconfig
```

图 2.4-4 打开配置菜单

打开后的界面如下所示。

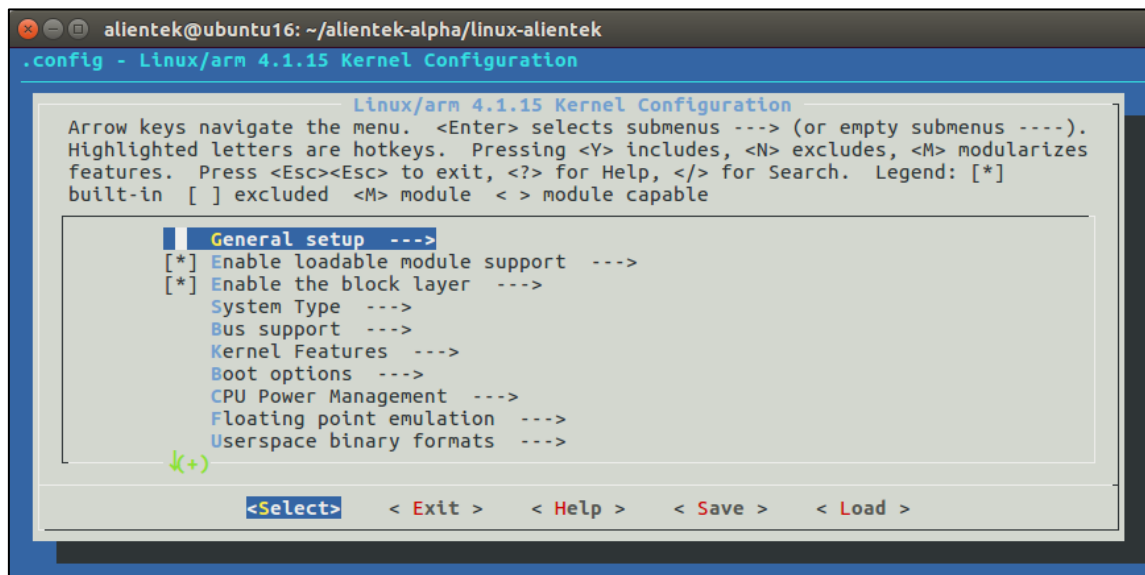


图 2.4-5 Kernel Configuration 界面

注意，为了菜单显示完整，Ubuntu 终端要足够宽，如果终端宽度不够，就会报如下错误。

```
alientek@ubuntu16:~/alientek-alpha/linux-alientek
alientek@ubuntu16:~/alientek-alpha/linux-alientek$ make menuconfig
scripts/kconfig/mconf Kconfig

*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

alientek@ubuntu16:~/alientek-alpha/linux-alientek$ make menuconfig
scripts/kconfig/mconf Kconfig
Your display is too small to run Menuconfig!
It must be at least 19 lines by 80 columns.
scripts/kconfig/Makefile:28: recipe for target 'menuconfig' failed
make[1]: *** [menuconfig] Error 1
Makefile:541: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
alientek@ubuntu16:~/alientek-alpha/linux-alientek$
```

图 2.4-6 因终端过小而引起的报错

配置内核 logo 显示。

Kernel Configuration 的基本用法：方向键上下选中目录/配置项，按回车键进入选中目录，按 y 键设置配置项，按 n 将取消配置项。按 Esc 键返回上一级目录，在顶层目录按两下 Esc 则退出。

要进入的目录:

- Device Drivers
 - Graphics support
 - Bootup logo
 - [*] Standard 224-color Linux logo

将 Standard 224-color Linux logo 选项选中, 也可以直接将 Bootup logo 整个选中。这样就配置了 224 色彩的 logo。连续按 Esc 键直至退出 Kernel Configuration, 如果修改过配置, 则会提示是否保存, 选中 yes 选项即可保存并退出。如果 Bootup logo 下还有别的 logo 选项, 要屏蔽掉 (按 N 键), 只保留 Standard 224-color Linux logo 这一项。

2.5 编译内核

执行以下指令编译内核 zImage 文件。

```
make zImage -j8
```

编译成功就会提示 zImage is ready。

```
AS      arch/arm/boot/compressed/hyp-stub.o
AS      arch/arm/boot/compressed/lib1funcs.o
AS      arch/arm/boot/compressed/ashldi3.o
AS      arch/arm/boot/compressed/bswapsdi2.o
AS      arch/arm/boot/compressed/piggy.lzo.o
LD      arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
allientek@ubuntu16:~/allientek-alpha/linux-allientek$
```

图 2.5-1 编译成功

将 arch/arm/boot 目录下的 zImage 烧录到开发板或者使用 TFTP 挂载到开发板上, 启动开发板即可显示 logo。



图 2.5-2 显示效果

第三章 设置内核 logo 全屏

V2.3 版本的出厂内核上修复了 logo 不能全屏的问题,如果是使用 V2.2 及更老版本的内核,可以根据本章进行内核修改。

正点原子出厂内核logo使用了一张428*214的图片,背景也是黑色的,如果使用样大小的图片就不需要进行居中操作。如果想要使用全屏的图片。那么可以修改以下文件。删除428*214的居中代码,后面用户想要全屏,经过研究代码发现:内核里有个代码判断logo的大小与屏的大小对比可能欠妥。对比logo的高度用了DIV_ROUND_UP取整操作。这样会导致误差,所以1024*600的屏幕没法全屏。

要设置内核logo全屏的话,需要删除掉源码中对图片居中的设置判断。

注意图片的尺寸大小要符合屏幕分辨率大小。

3.1 图片准备

这里笔者使用的是7寸1024*600的屏幕,准备了一张1024*600尺寸的图片。图片的背景色最好不要渐变色,效果不好。建议背景色使用纯色的。



图 3.1-1 准备图片

参考第二章的方法,生成图片的 ppm 文件。将 ppm 文件重命名为 logo_linux_clut224.ppm 并替换到内核源码 drivers/video/logo 目录里。

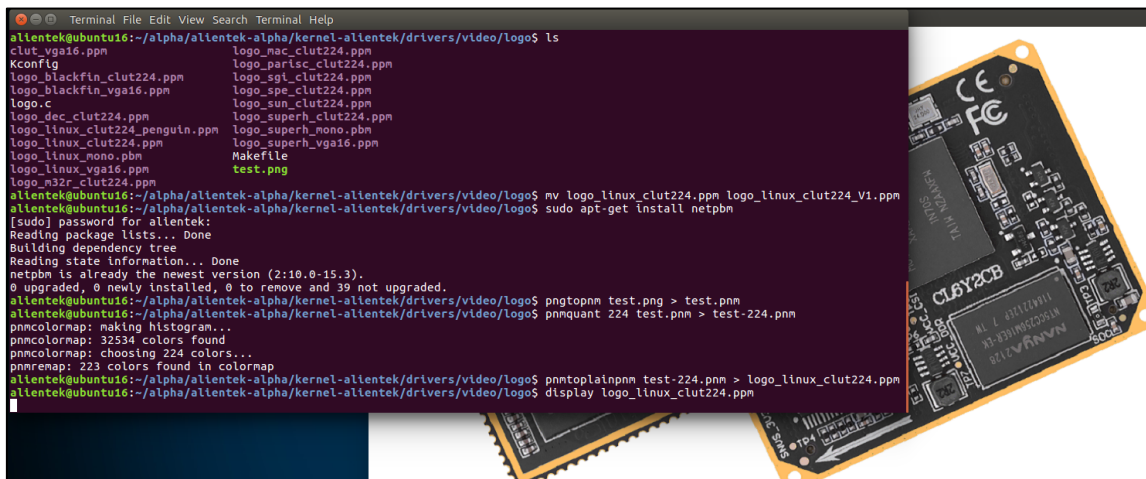


图 3.1-2 生成 ppm 文件

3.2 修改内核 logo 全屏

打开出厂内核源码 drivers/video/console/fbcon.c 文件。

搜索 fbcon_prepare_logo 函数，注释掉 logo 居中设置。

添加 logo_lines = logo_height / vc->vc_font.height;

如下图所示：

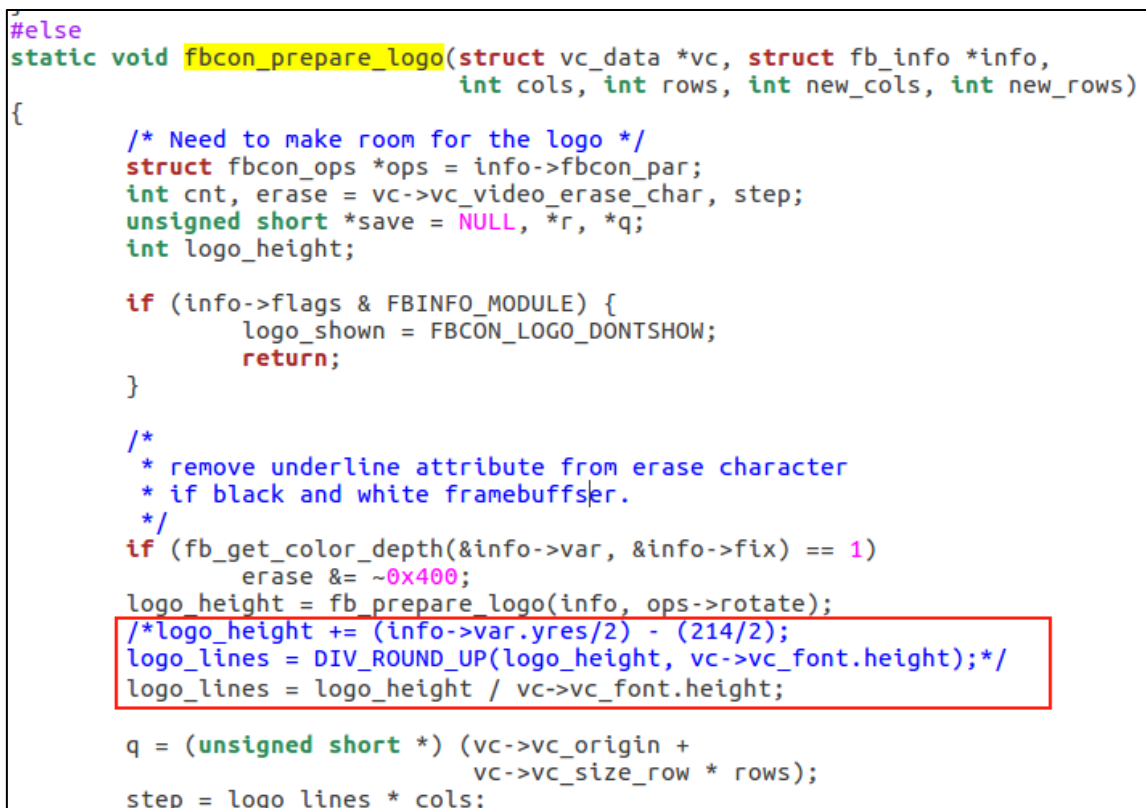


图 3.2-1 修改 fbcon.c

打开出厂内核源码 drivers/video/fbdev/core/fbmem.c。

搜索 static int fb_show_logo_line 函数, 注释掉 logo 居中设置。

如下图所示。

```
image.dx = 0;
image.dy = y;
//image.dx = (info->var.xres/2) - (428/2);
//image.dy = (info->var.yres/2) - (214/2);
image.width = logo->width;
image.height = logo->height;

if (rotate) {
    logo_rotate = kmalloc(logo->width *
                          logo->height, GFP_KERNEL);
    if (logo_rotate)
```

图 3.2-2 修改 fbmem.c

3.3 验证测试

修改完这些文件后, 编译出厂内核和设备树, 替换到开发板上启动。进度条 psplash 里也有 logo 图片, 参考 1.4 小节删除出厂系统进度条的方法, 就可以实现开机显示全屏 logo 直到进入 Qt 界面。

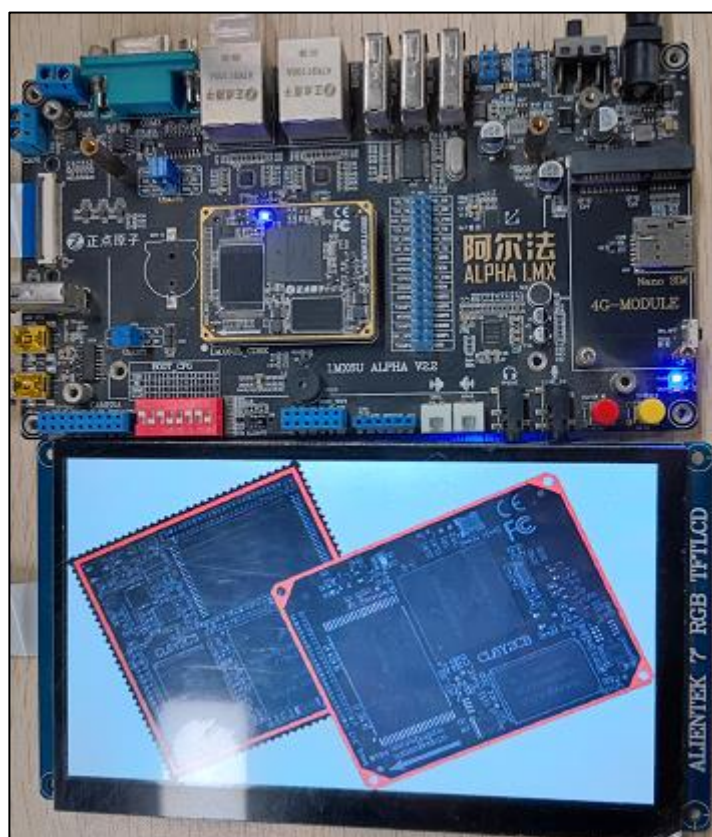


图 3.3-1 内核 logo 全屏