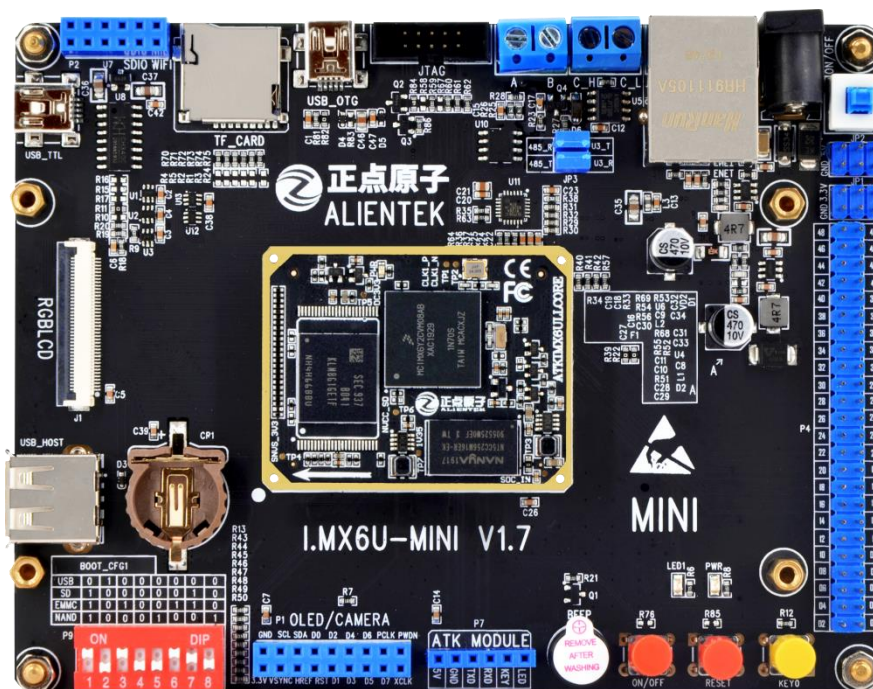
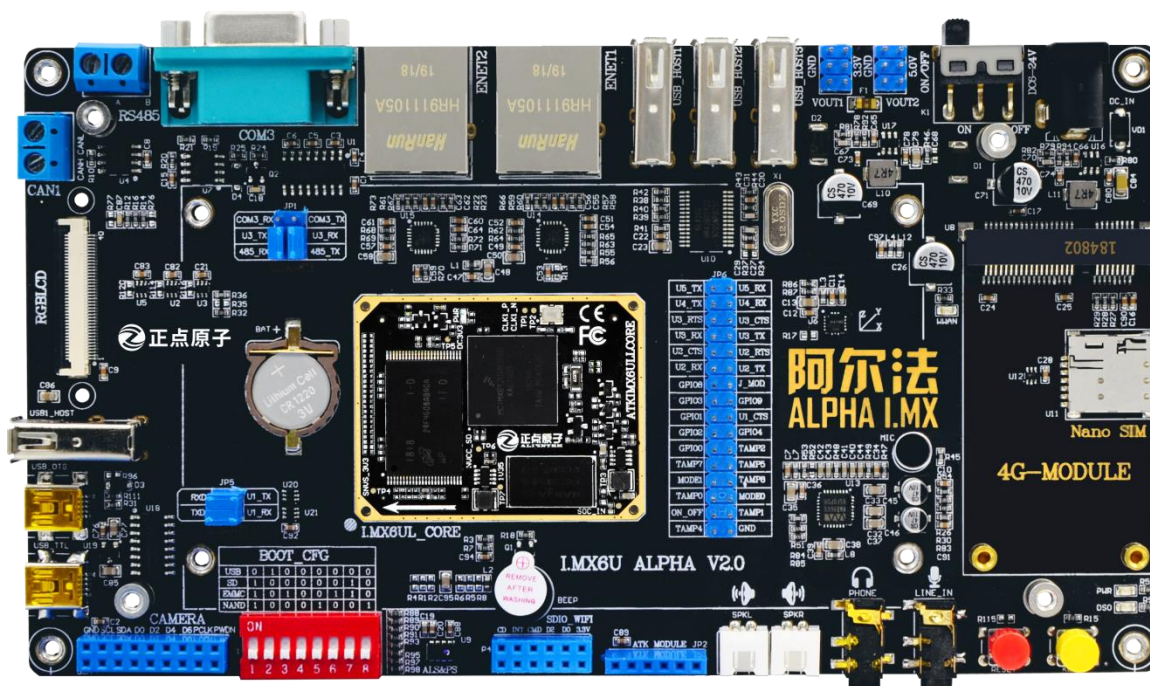


I.MX6U 虚拟机

使用参考手册 V1.2.1





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: 测试版本	正点原子 linux 团队	正点原子 linux 团队	2021.03.08
V1.1	添加网盘链接地址。 修复一些笔误。	正点原子 linux 团队	正点原子 linux 团队	2021.03.22
V1.2	镜像改进: 适配 Qt 教程。 1、修复部分源码问题。 2、在 Qt 启动脚本中添加使能环境变量。 3、添加 Qt creator 的出厂交叉编译器。 4、添加 Qt 远程调试。 5、修复 Qt 插件 ClangCodeModel 问题。 6、修复 ubuntu16 中 Qt 的 openssl 问题。 7、关闭 ubuntu 升级版本提醒。 8、添加搜狗拼音和五笔输入法。 9、添加出厂系统 opencv 库。 文档更新: 1、更新新的虚拟机适配的内容。 2、添加第九章 Qt 教程源码测试。	正点原子 linux 团队	正点原子 linux 团队	2021.06.16
V1.2.1	1、添加前言内容。 2、1.3.1 小节添加解压软件下载链接。	正点原子 linux 团队	正点原子 linux 团队	2021.09.25

目录

前言	6
第一章 安装虚拟机.....	7
1.1 电脑配置要求.....	7
1.2 开发环境.....	7
1.3 虚拟机下载.....	7
1.3.1 解压镜像.....	7
1.4 安装 VMware 软件	8
1.5 安装 alientek 虚拟机.....	8
1.5.1 主机磁盘分区.....	8
1.5.2 创建虚拟机.....	8
1.5.3 安装 alientek.iso 镜像系统	17
1.6 配置网络环境.....	23
第二章 虚拟机常用目录汇总.....	24
2.1 概念	24
2.1.1 根目录 (/)	24
2.1.2 用户主目录 (~)	24
2.1.3 当前路径.....	24
2.1.4 绝对路径.....	24
2.1.5 相对路径.....	24
2.2 用户主目录文件.....	25
2.2.1 ~/alpha.....	25
2.2.2 ~/linux	26
第三章 文件互传.....	28
3.1 Ubuntu 和 Windows 主机互传	28
3.1.1 filezilla	28
3.1.2 samba 服务器	28
第四章 编译开发板出厂镜像.....	31
4.1 编译出厂 uboot	31
4.2 编译出厂内核源码及设备树.....	31
4.3 编译出厂 Qt GUI 综合 Demo.....	32
第五章 编译教程镜像.....	34
5.1 编译教程移植好的 uboot 源码.....	34
5.2 编译教程内核源码.....	35
第六章 网络挂载教程镜像.....	37
6.1 配置开发板 uboot 的网络信息.....	37
6.2 TFTP 加载 zImage 和设备树.....	37
6.3 NFS 挂载教程文件系统	38
第七章 网络挂载出厂镜像.....	40
7.1 TFTP 挂载出厂内核和设备树	40
7.2 NFS 挂载出厂文件系统	40
第八章 Qt 相关	42

8.1 Qt Creator.....	42
8.2 虚拟机上运行出厂系统 Qt GUI 界面.....	42
8.2.1 导入出厂 Qt 工程.....	42
8.2.2 显示图片.....	45
8.2.3 播放音频.....	46
第九章 在出厂系统上运行 Qt 教程源码.....	48
9.1 关闭出厂系统桌面.....	48
9.2 使能出厂交叉编译器环境变量.....	48
9.3 编译运行 Qt 摄像头例程.....	49
9.3.1 Qt 程序运行在 Ubuntu 上.....	50
9.3.2 Qt 程序运行在出厂系统上.....	53
9.3.3 使用网络进行 Qt 远程调试.....	56
9.4 重新选择构建套件.....	58

前言

正点原子 Linux 开发板发布已有一年多的时间了, 用户众多, 有想入门学习驱动的学生, 有想转行做嵌入式开发的工程师, 也有想深入学习嵌入式的发烧友等等。所谓众口难调, 每个人的需求都不一样, 之前就有用户反馈学习周期长, 想要快速体验或者开发, 这就需要一个较为完善的操作系统。正点原子的驱动开发指南所使用的是纯净的 Ubuntu16.04 平台, 用户可以按照驱动开发指南的学习路线, 一步一步地亲手把 Ubuntu 平台搭建完成。但是有些用户不想自己搭建平台, 只需要在平台上开发, 因此我们花了一些时间, 搭建了一个虚拟机平台, 提供给有需要的用户使用。

【用户基础】

用户使用的不是自己搭建的虚拟机, 对虚拟机的环境、操作等不太熟悉, 需要一个简单的文档来指导一下, 因此推出了这份文档, 同时这就需要用户有**一定的基础能力**, 如熟练使用 VI 编辑器、熟悉 Linux 指令、熟悉正点原子的资料分布等, 不然甚至会是事倍功半的效果。

建议初学者还是按照驱动开发指南的路线来学习, 打好基础。再次声明, 此平台只使用于有 Linux 开发经验的用户。

【平台内容】

Ubuntu 系统: Ubuntu16.04

GCC 版本: 5.4

教程通用交叉编译器: gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabihf

出厂系统专用交叉编译器: fsl-imx-x11-glibc-x86_64-meta-toolchain-qt5-cortexa7hf-neon-toolchain-4.1.15-2.1

QT 环境: 开源版本 QT5.12.9、开源版本 QT Creator4.12.2

编辑器软件: Visual Studio Code、vim、gedit

网络工具: TFTP、NFS、Samba

以上软件都是开源版本, 仅提供给有需要的用户学习使用, 如有需要请自行购买商业版本软件。

【参考文档】

系统搭建、驱动学习部分请参考《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》

QT 环境搭建部分请参考《【正点原子】I.MX6U 出厂系统 Qt 交叉编译环境搭建 V1.6》

网络搭建部分请参考《【正点原子】I.MX6U 网络环境 TFTP&NFS 搭建手册》

【路径问题】

本平台使用的用户名、具体路径与驱动开发指南、QT 移植等文档里的路径不同, 用户在使用时以实际路径为准。初学者很容易因为文件路径出现问题, 建议先学习好 Linux 文件管理和路径的相关知识, 可以参考驱动开发指南的 Ubuntu 基础入门或者网上的学习资料, 如 <https://www.runoob.com/linux/linux-file-content-manage.html>

第一章 安装虚拟机

1.1 电脑配置要求

电脑主机的配置要求不是必须的, 为的是提高开发体验和避免因性能过低产生的编译出错, 可以根据自己实际情况分配虚拟机性能, 以下只是建议:

分配虚拟机处理器: i5 处理器四核以上。

分配虚拟机内存: 4GB 以上。

分配虚拟机硬盘: 建议单独分区一个 100GB 以上的硬盘空间给虚拟机。

1.2 开发环境

软件: VMware Workstation Pro15.5 版本。

主机操作系统: WIN10。

虚拟机系统: Ubuntu16.04 版本

1.3 虚拟机下载

网盘地址链接: <https://pan.baidu.com/s/1CYUheJNsCdGov0XW5vvGwQ>

提取码: atkq

在网盘里有四个 alientek_Qt.iso.xz p 文件, 分别以 001、002、003、004 为后缀, 这里要下载这四个文件。

如果链接失效了, 可以在正点原子下载中心下载:

<http://www.openedv.com/docs/boards/arm-linux/zdyz-i.mx6ull.html>

1.3.1 解压镜像

因为打包出来的虚拟机太大, 百度云限制单个文件最大为 4G, 因此这里拆成了四个文件上传到百度云。下载完成后, 确保这四个文件都在同一文件夹下, 选中这四个文件解压就可以了。

可以下载解压软件 7-Zip 来进行解压, 软件下载地址: <https://www.7-zip.org/>。笔者也将这个软件放在网盘资料盘里, 路径: 开发板光盘 A-基础资料\03、软件\7z2102-x64.exe, 没有的话可以去更新下网盘资料。双击安装, 安装完后, 将四个压缩文件全部选中右键, 使用 7-Zip 提取到当前位置。

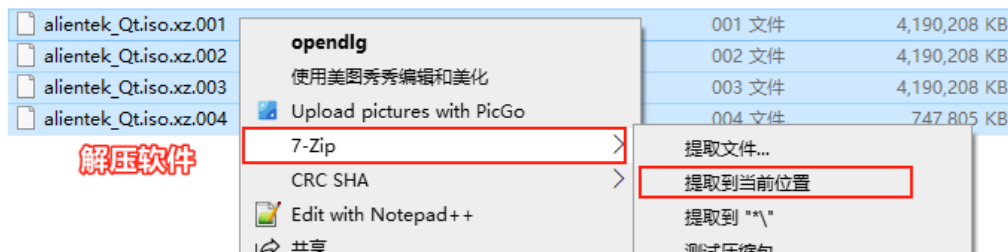


图 1.3-1 解压镜像

最终得到了一个 alientek_Qt.iso 镜像文件 (大约 13G), 这个就是我们后面要导入到虚拟机的镜像文件。iso 文件可以不用和安装的 ubuntu 在同一个硬盘分区, 也不建议放在移动磁盘或者 U 盘。

1.4 安装 VMware 软件

参考《【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南》第一章的 1.1 小节即可，方法比较简单，这里不再赘述。

1.5 安装 alientek 虚拟机

1.5.1 主机磁盘分区

首先自行在主机上分配出一个 100G 以上的磁盘空间，这里我分配了一个 250G 的磁盘作为虚拟机空间。（不熟悉此操作的话可以参考下百度）



图 1.5-1 磁盘空间

1.5.2 创建虚拟机

打开 VMware Workstation Pro，选择 新建虚拟机 选项。

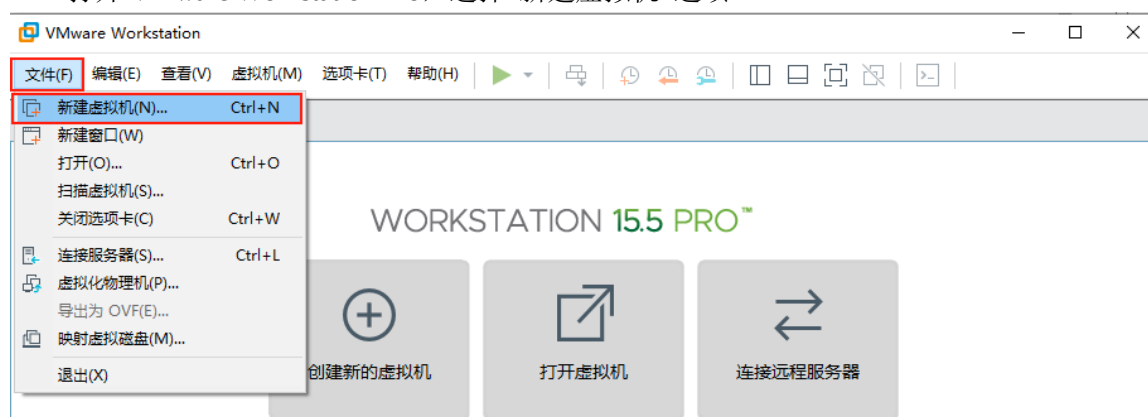


图 1.5-2 新建虚拟机

弹出向导界面，配置类型选择“自定义”。

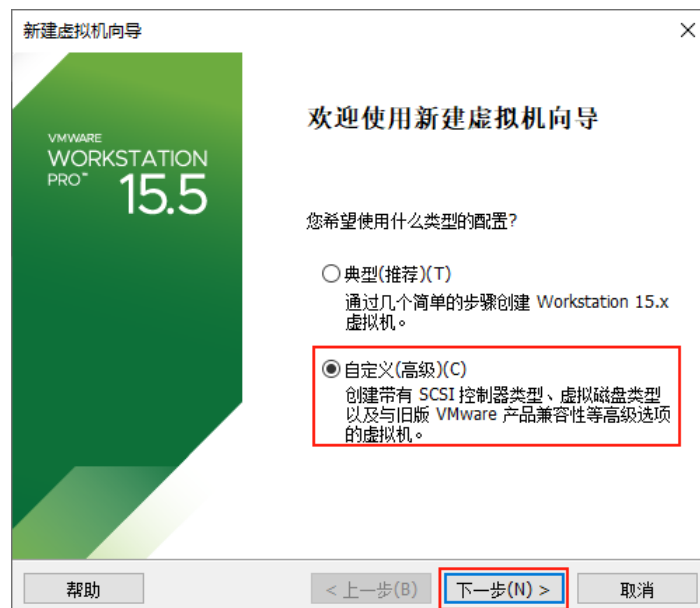


图 1.5-3 配置类型自定义

硬件功能默认即可，点击下一步。

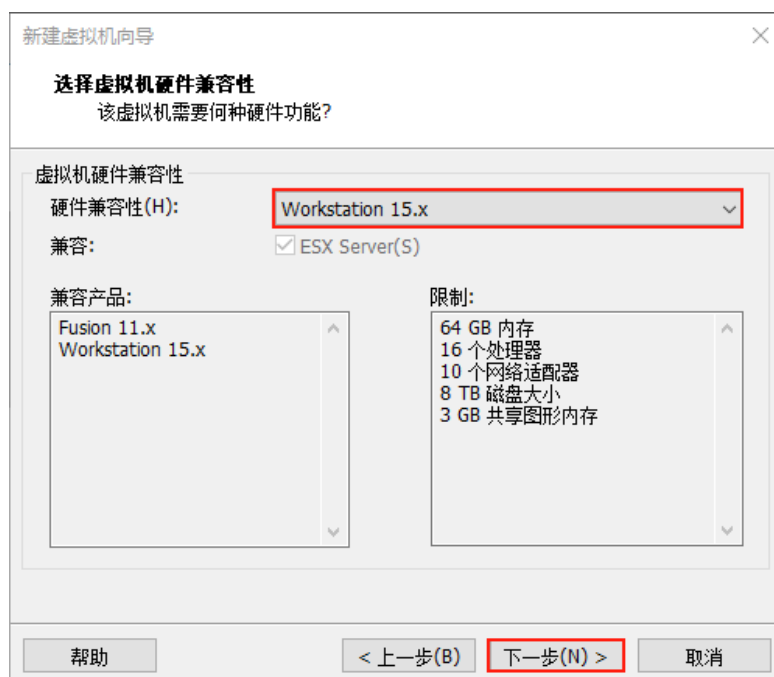


图 1.5-4 硬件功能默认

在“安装客户机操作系统”界面，选择“安装程序光盘映像文件”，使用下载好的 alientek.iso 文件安装系统。

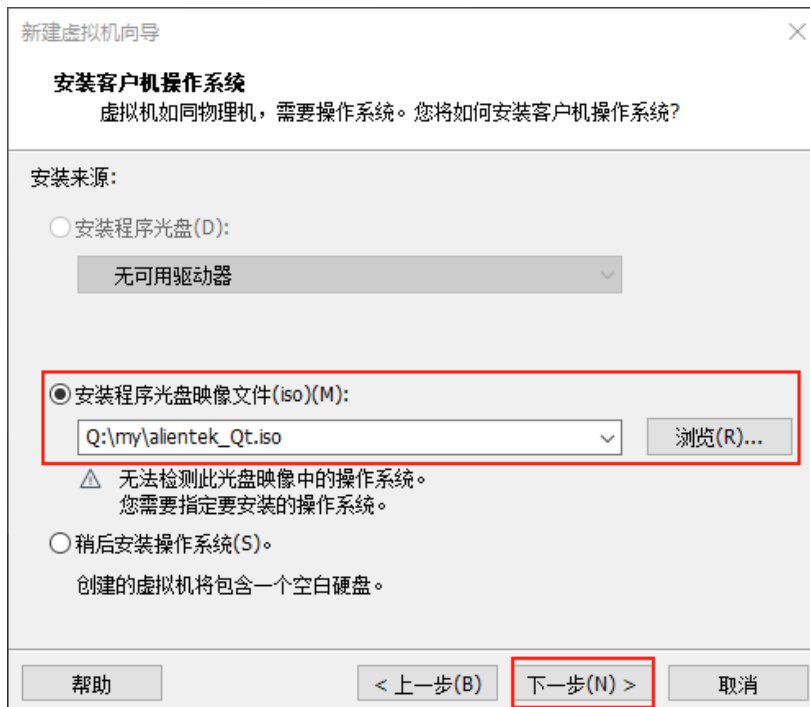


图 1.5-5 安装 alientek_Qt.iso

在“客户机操作系统”界面选择 Linux 操作系统，版本为 Ubuntu64 位。

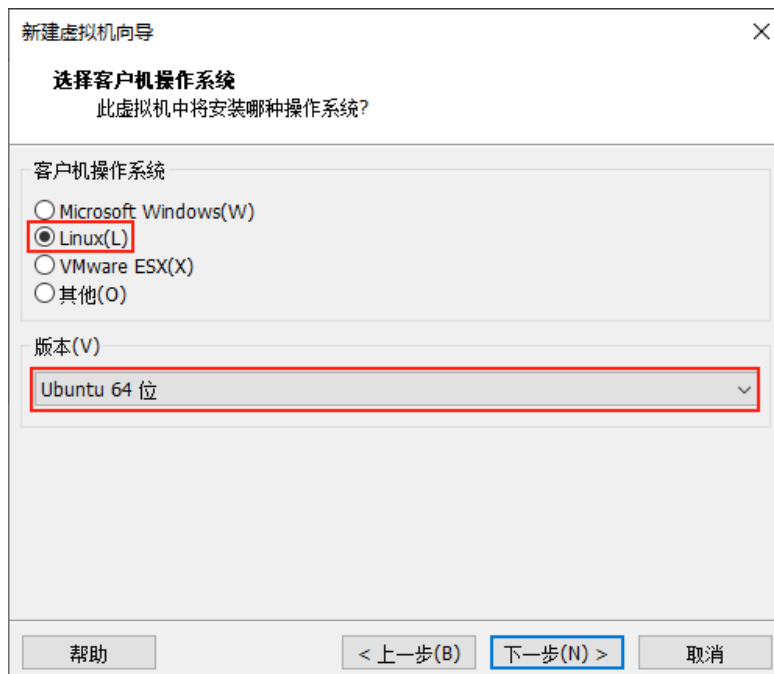


图 1.5-6 选择操作系统版本

在“命名虚拟机”界面，可以自定义虚拟机名称，这里笔者设置为 alientek。“位置”要选择刚刚分配的磁盘空间，比如笔者这里就是 M 盘。

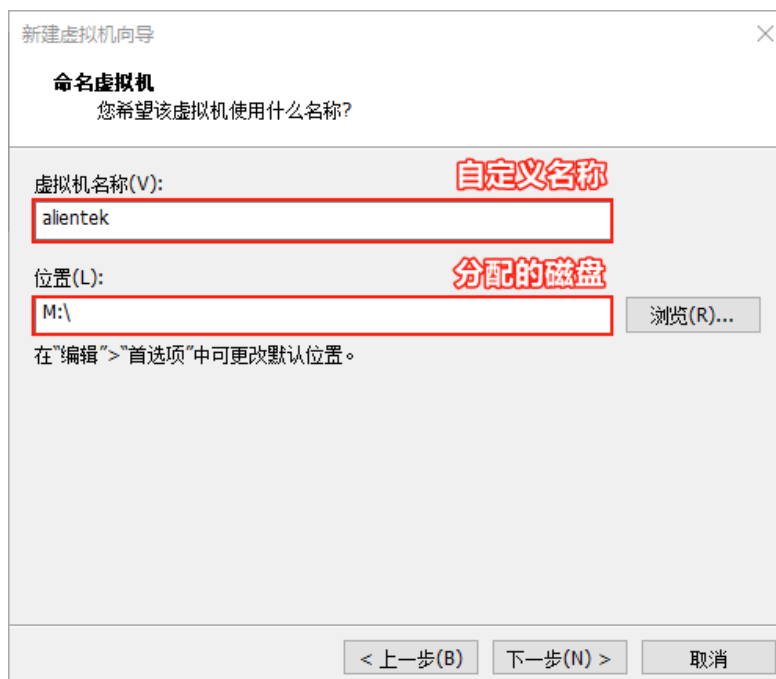


图 1.5-7 命名虚拟机及位置

在“处理器配置”界面，根据自己的实际情况给虚拟机分配内核数即可，这里笔者电脑是 8 核的，因此分配了 4 核给虚拟机。

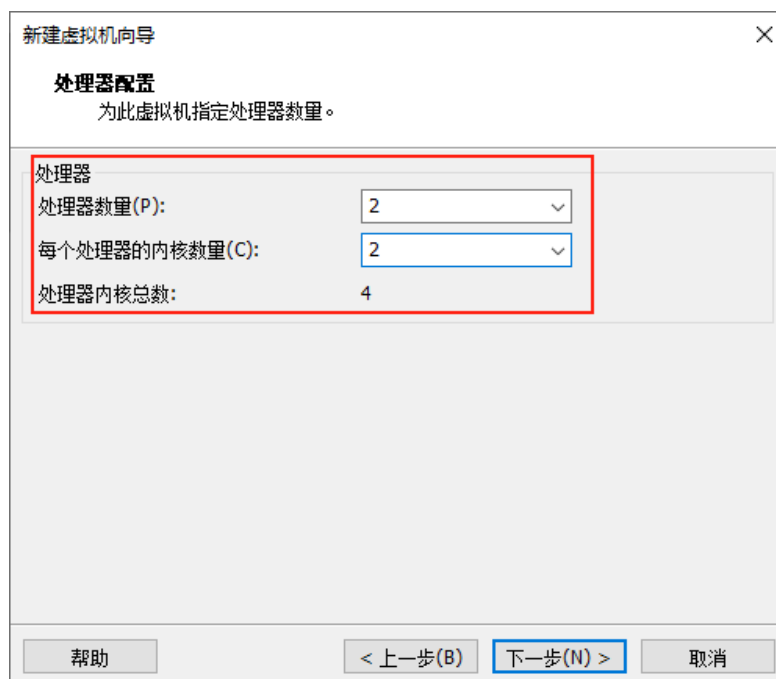


图 1.5-8 处理器配置

同理，根据自己的情况选择虚拟机的内存大小，这里笔者设置为 4GB。

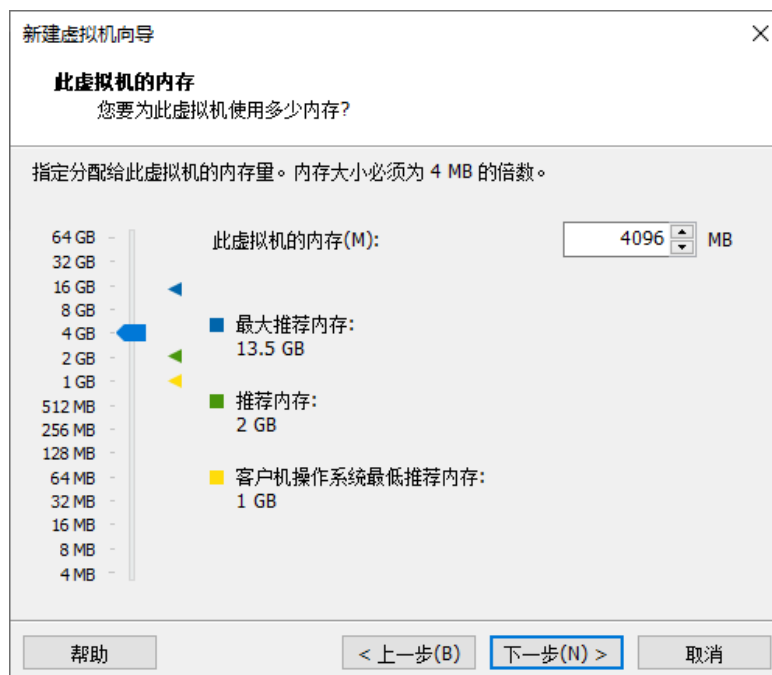


图 1.5-9 设置虚拟机内存

“网络类型”选择“使用桥接网络”。

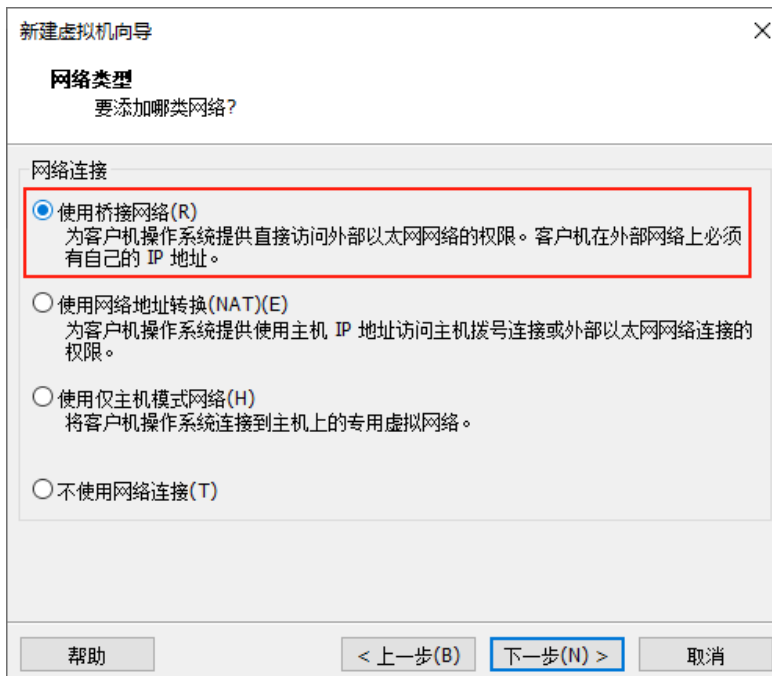


图 1.5-10 使用桥接网络

I/O 控制器类型选择默认。



图 1.5-11 控制器类型

“磁盘类型”也是选择默认的。



图 1.5-12 磁盘类型

“选择磁盘”里选择“创建新虚拟磁盘”。

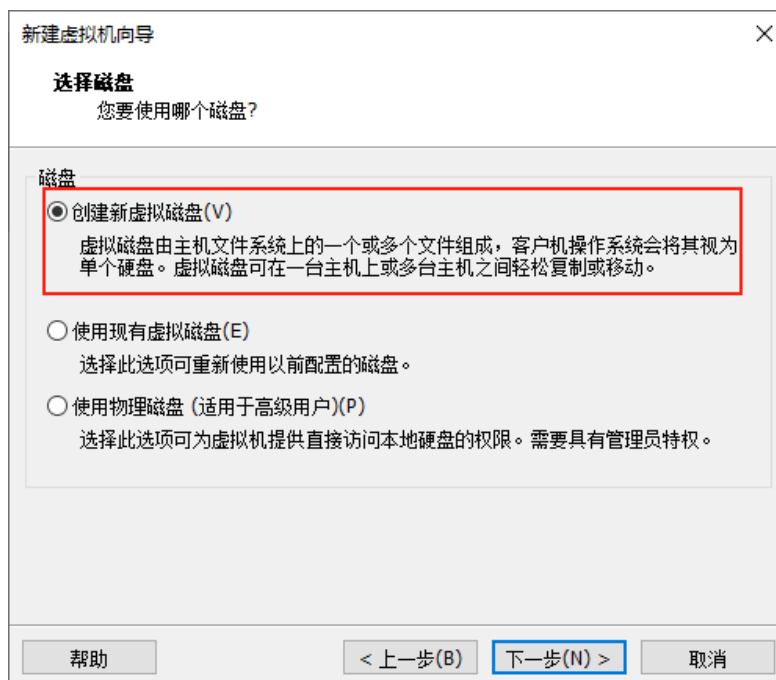


图 1.5-13 选择磁盘

“指定磁盘容量”选择之前自己分配的磁盘空间大小即可，如果是 200GB 就填写 200GB，笔者之前创建的是 250GB，这里就写 250GB。



图 1.5-14 磁盘大小

“指定磁盘文件”使用默认的即可，这里笔者的默认为 alientek.vmdk。

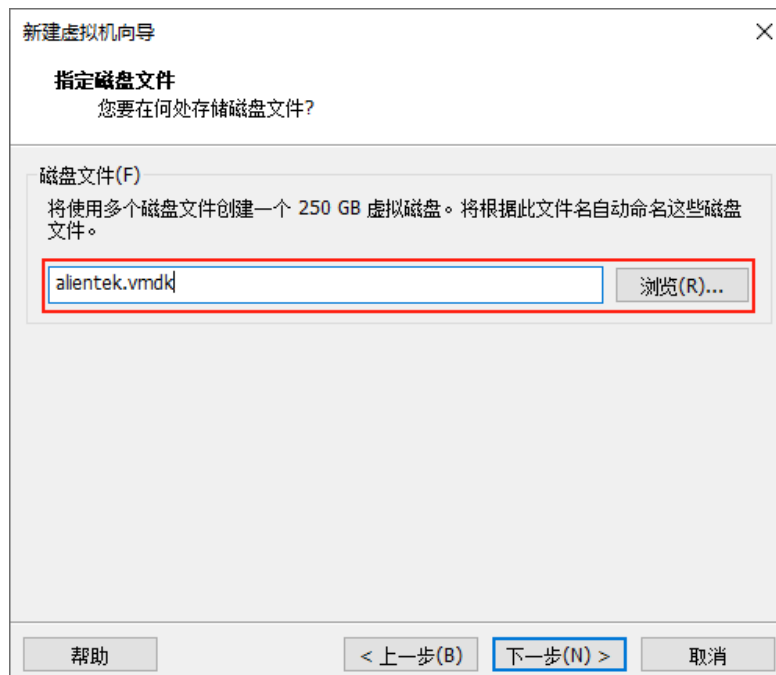


图 1.5-15 默认磁盘文件

完成创建虚拟机。



图 1.5-16 完成

创建完成后, 我们再点击菜单栏的“虚拟机”->“设置”, 将 USB 控制器的兼容性修改为 USB3.0, 不然后面使用读卡器或者其他 USB 设备的时候可能会出问题。

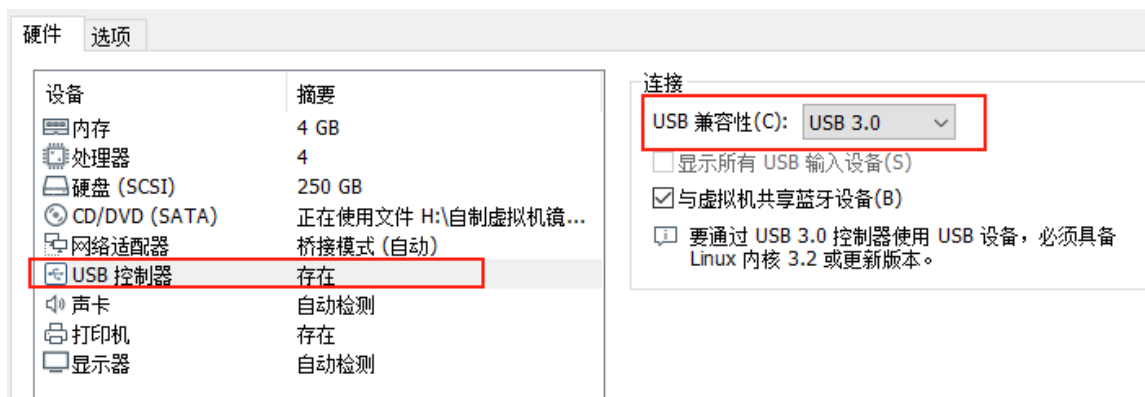


图 1.5-17 设置 USB 兼容性为 3.0

1.5.3 安装 alientek.iso 镜像系统

创建完虚拟机后, 点击启动虚拟机的按键。

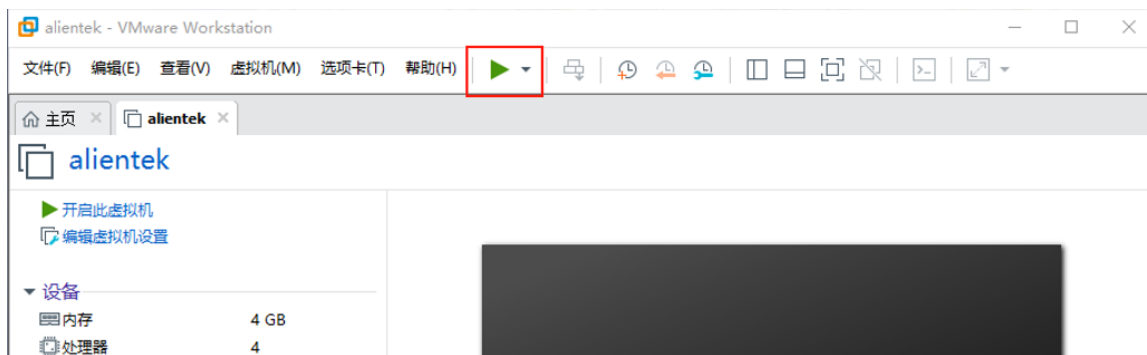


图 1.5-18

图 1.5-19 启动虚拟机

启动完后, 系统有 10 秒的时间选择启动系统还是引导安装系统。这里我们需要在此界面, 用方向键↓键, 将光标选择第二项来安装系统, 如下图所示。

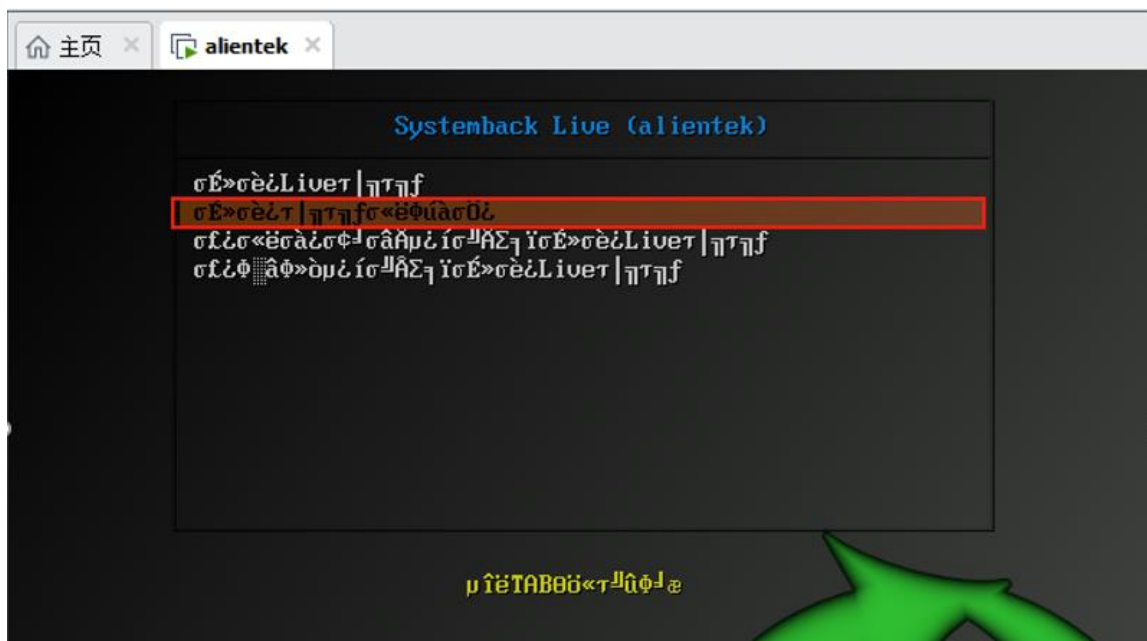


图 1.5-20 选择第二项进行安装系统

如果没在 10 秒内选择, 系统会自动根据光标所在位置进行下一步操作, 即自动启动系统, 进入这个默认的系统, 是不会保存文件的, 千万不要进这个第一个选项的系统。

如果不是使用第二项启动的, 等系统启动完成, 重启虚拟机 (关机时会提示按下 Enter 键继续, 此时只要按下 Enter 回车键即可关机), 重新进入到系统引导界面, 再重新选择第二项启动即可。

启动后会进入系统安装界面，“新用户全名”（首字母默认大写）和“新用户登录名”必须设置为 **alientek**，因为安装的镜像系统中的用户名为这个，必须保持一致，不然后面会有路径问题和权限问题。密码设置为数字 0，主机名称设置为 **ubuntu16** 即可。



系统安装

新用户全名：
Alientek ✓

新用户登录名：必须设置为alientek
alientek ✓

新用户密码：
0 ✓

新管理员密码（可选，Ubuntu里不建议使用）：
[Empty field]

新主机名：
ubuntu16 ✓

» 下一步

图 1.5-21 系统安装

点击下一步，进入“分区设置”界面，默认配置如下图所示。



图 1.5-22 默认分区设置界面

这里我们需要选中当前的分区，将其删除，如下图所示。



图 1.5-23 删除默认分区

删掉默认分区后, 出现待配置的/dev/sda?分区, 选中它后再点击绿色箭头 , 进行新建分区区的操作, 如下图所示。如果配置有误, 点击  按钮即可重新配置。



图 1.5-24 新建分区

新建分区后会生成/dev/sda1分区, 选中此分区, 按照下图依次设置挂载点为根节点, 文件系统格式为 ext4 格式, 勾选传递用户配置文件及用户数据, 选择安装引导程序为/dev/sda, 全部配置完后点击图中的⑩号箭头 , 应用此配置, 再点击下一步。如果配置有误, 点击  按钮即可重新配置。



图 1.5-25 分区设置

配置完的分区如下图所示, 确认无误后点击下一步。



图 1.5-26 配置完的分区

接下来会提示是否有还原点来安装系统, 我们点击开始即可安装系统。

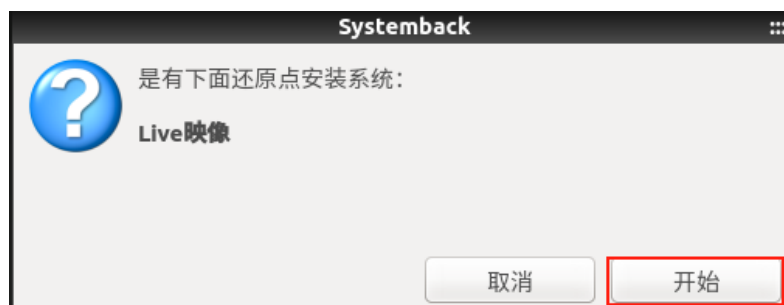


图 1.5-27 点击开始

接下来就是等待系统安装完成就可以了。



图 1.5-28 正在安装系统

最后提示安装系统已完成, 点击 OK 后会提示是否重启系统, 我们点击重启系统即可。



图 1.5-29 系统安装完成

如果重启过程中出现下图提示, 点击回车键即可。可以多按几次回车, 等待系统启动。

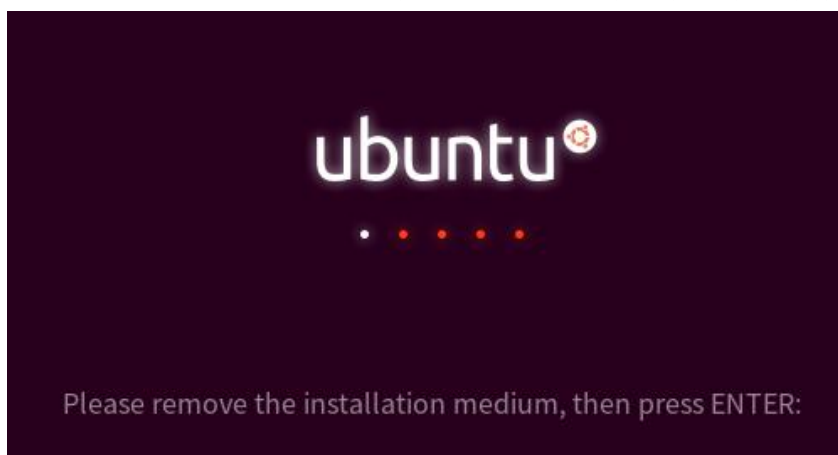


图 1.5-30 重启提示按下回车键

在登陆界面使用 Alientek 用户登陆即可, 密码为之前设置的, 笔者这里为 0。

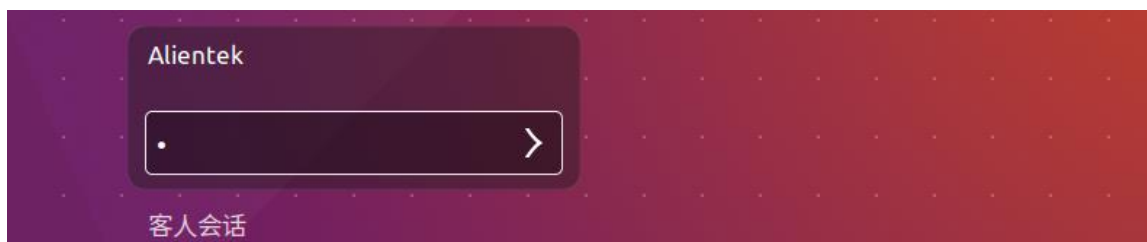


图 1.5-31 登陆系统

系统界面如下图, 在终端执行 ls 指令可以查看家目录下的文件。



图 1.5-32 系统界面

1.6 配置网络环境

每个人的电脑网络拓扑不一定一样, 搭建网络环境的方法也不一样。为此, 正点原子团队总结了一份《**【正点原子】I.MX6U 网络环境 TFTP&NFS 搭建手册**》。如果是直接使用正点原子虚拟机的用户, 请根据此文档**第一章**的内容, 根据自己的上网环境来选择适合的小节来搭建环境, 这里就不再赘述。TFTP 和 NFS 工具在镜像里已经配置好了。

第二章 虚拟机常用目录汇总

2.1 概念

2.1.1 根目录 (/)

执行 `cd /` 即可进入根目录, `/` 就是根目录名。

```
alientek@ubuntu16:~$ cd /
alientek@ubuntu16:/$ ls
bin      dev      initrd.img  lib32      media     proc      sbin      sys      var
boot     etc      initrd.img.old  lib64      mnt       root      snap      tmp      vmlinuz
cdrom    home     lib         lost+found  opt       run       srv       usr      vmlinuz.old
```

图 2.1-1 根目录

2.1.2 用户主目录 (~)

执行 `cd ~` 即可进入根目录, `~` 就是根目录名。

```
alientek@ubuntu16:/$ cd ~
alientek@ubuntu16:~$ ls
alpha      linux      openssh    qt5.12.9    tslib-1.21  视频  下载
examples.desktop  opencv     openssl    qt_demo     公共的      图片  音乐
ftp_alpha   opencv_IMX6U  other      rootfs_all  模板        文档  桌面
```

图 2.1-2 用户主目录

2.1.3 当前路径

执行 `pwd` 指令即可看到当前所在的路径信息, 例如此时笔者所在路径为 `/home/alientek`

```
alientek@ubuntu16:~$ pwd
/home/alientek
```

图 2.1-3 当前路径

2.1.4 绝对路径

写法: 只能由根目录写起, 例如到 `alpha` 目录下。

`cd /home/alientek/alpha`

```
alientek@ubuntu16:~$ cd /home/alientek/alpha/
alientek@ubuntu16:~/alpha$
```

图 2.1-4 绝对路径

2.1.5 相对路径

写法: 不是由根目录写起, 例如到 `alpha` 目录下。

`cd ~/alpha`

```
alientek@ubuntu16:~/alpha$ cd ~/alpha/
alientek@ubuntu16:~/alpha$ cd ../alpha/
alientek@ubuntu16:~/alpha$ cd ../../alientek/alpha/
```

图 2.1-5 相对路径

更多指令可以网上找下学习资源: <https://www.runoob.com/linux/linux-file-content-manage.html>

2.2 用户主目录文件

2.2.1 ~/alpha

```
cd ~/alpha/
tree -L 2
```

```

alientek@ubuntu16:~/alpha$ cd ~/alpha/
alientek@ubuntu16:~/alpha$ tree -L 2
.
├── alientek-alpha      出厂系统源码
│   ├── kernel-alientek  出厂系统内核源码
│   ├── qt-alientek     出厂系统QT源码
│   ├── rootfs-alientek  出厂系统文件系统
│   └── uboot-alientek   出厂系统uboot源码
├── alientek-zzk        教程源码
│   ├── demo            教程例程源码
│   ├── kernel-zzk      移植好的教程源码
│   ├── rootfs-zzk      教程busybox文件系统
│   └── uboot-zzk        移植好的教程uboot源码
└── nxp                 NXP源码
    ├── kernel-nxp      NXP官方内核源码
    └── uboot-nxp        NXP官方uboot源码

13 directories, 0 files

```

图 2.2-1 alpha 目录

这里要注意下，因为出厂系统会不定时维护更新，因此用户当前开发板上的源码版本可能和此虚拟机里的源码版本不一致，用户需要自己在正点原子网盘或者 gitee 上下载最新源码/适配开发板当前版本的源码，拷贝到虚拟机中编译。

这里出厂内核版本为 linux-imx-4.1.15-2.1.0-gad512fa-v1.7.tar.bz2。

```
alientek@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$ ls
```

arch	Documentation	ipc	MAINTAINERS	net	sound	vmlinux
block	drivers	Kbuild	Makefile	README	System.map	vmlinux.o
build.sh	firmware	Kconfig	mm	REPORTING-BUGS	tmp	
COPYING	fs	kernel	modules.builtin	samples	tools	
CREDITS	include	lib	modules.order	scripts	usr	
crypto	init	linux-imx-4.1.15-2.1.0-gad512fa-v1.7.tar.bz2	Module.symvers	security	virt	

图 2.2-2 虚拟机中出厂系统内核版本为 v1.7

这里出厂 uboot 版本为 uboot-imx-2016.03-2.1.0-ge468cdc-v1.5.tar.bz2。

```
alientek@ubuntu16:~/alpha/alientek-alpha/uboot-alientek$ ls
```

api	config.mk	examples	Licenses	README	tools	u-boot-nodtb.bin
arch	configs	fs	MAINTAINERS	scripts	u-boot	u-boot.srec
board	disk	include	MAKEALL	snapshot.commit	u-boot.cfg	u-boot.syn
build.sh	doc	Kbuild	Makefile	System.map	uboot-imx-2016.03-2.1.0-ge468cdc-v1.5.tar.bz2	
cmd	drivers	Kconfig	net	test	u-boot.lds	
common	dtb	lib	post	tmp	u-boot.map	

图 2.2-3 虚拟机中出厂系统 uboot 版本为 v1.5

进入到虚拟机里的出厂系统文件系统目录下，解压压缩包后即可执行指令查看文件系统版本。

```
cd ~/alpha/alientek-alpha/rootfs-alientek
tar vxjf rootfs.tar.bz2           //解压出厂系统压缩包
cat etc/version
```

```

alientek@ubuntu16:~/alpha/alientek-alpha/rootfs-alientek$ ls
bin  dev  home  media  opt  rootfs.tar.bz2  sbin  tmp  var
boot  etc  lib  mnt  proc  run  sys  usr  文件系统版本说明.txt
alientek@ubuntu16:~/alpha/alientek-alpha/rootfs-alientek$ cat /etc/version
cat: /etc/version: 没有那个文件或目录
alientek@ubuntu16:~/alpha/alientek-alpha/rootfs-alientek$ cat etc/version
Version: v1.6
Author: 正点原子
Date: 2020.11.18

```

图 2.2-4 虚拟机中出厂文件系统版本为 V1.6

在~/alpha/alientek-zzk 目录下是教程源码。

```

cd ~/alpha/alientek-zzk
tree -L 2

```

```

alientek@ubuntu16:~/alpha/alientek-alpha/rootfs-alientek$ cd ~/alpha/alientek-zzk
alientek@ubuntu16:~/alpha/alientek-zzk$ tree -L 2
.
├── demo                    例程源码
│   ├── A7-demo            裸机例程
│   └── linux-demo         驱动例程
├── kernel-zzk
│   ├── linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek 教程移植好的内核源码
│   └── linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek.tar.xz
├── rootfs-zzk
│   └── 3、文件系统
├── uboot-zzk
│   ├── uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek 教程移植好的uboot源码
│   └── uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek.tar.xz
└── 9 directories, 2 files

```

图 2.2-5 alientek-zzk 目录

注意这里的源码压缩包格式为 tar.xz，解压指令为 `tar -vxf 文件名.tar.xz`

```

tar -vxf uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek.tar.xz
tar vxf linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek.tar.xz

```

2.2.2 ~/linux

```

cd ~/linux
tree -L 2

```

```

alientek@ubuntu16:~/linux$ tree -L 2
.
├── IMX6ULL
│   ├── linux
│   └── tool
├── nfs
│   ├── buildrootfs
│   ├── mytest.c
│   ├── rootfs
│   └── ubuntu_rootfs
├── tftp
│   ├── imx6ull-alientek-emmc.dtb 教程设备树文件
│   ├── test.c                    tftp测试文件
│   └── zImage                    教程内核镜像文件
├── tool
│   ├── arm-linux-gnueabi-hf-gcc 教程交叉编译器
│   ├── deb                      教程软件包目录
│   ├── imxdownload              教程裸机烧录工具
│   ├── poky-arm-gcc             出厂系统交叉编译工具
│   └── vmware-tool              VMware-tool工具
└── 14 directories, 4 files

```

图 2.2-6 ~/linux 目录

这里要注意下, 虚拟机里提供了两种交叉编译工具链。

arm-linux-gnueabi-hf-gcc 是教程系统的交叉编译工具链, 用于编译教程内核、uboot、例程等源码。

poky-arm-gcc 里面的 fsl-imx-x11-glibc-x86_64-meta-toolchain-qt5-cortexa7hf-neon-toolchain-4.1.15-2.1.0.sh 是出厂系统交叉编译工具链, 用于编译出厂内核、uboot、出厂 QT 系统、自己 QT 应用等等。

查询编译器版本号:

```
arm-linux-gnueabi-hf-gcc -v
allientek@ubuntu16:~/linux/tool/arm-linux-gnueabi-hf-gcc$ arm-linux-gnueabi-hf-gcc -v
Using built-in specs.
COLLECT_GCC=arm-linux-gnueabi-hf-gcc
COLLECT_LTO_WRAPPER=/usr/local/arm/gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi-hf/bin/./libexec/gcc/arm-
linux-gnueabi-hf/4.9.4/lto-wrapper
Target: arm-linux-gnueabi-hf
Configured with: /home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg-build/target/arm-
linux-gnueabi-hf/snapshots/gcc-linaro-4.9-2017.01/configure SHELL=/bin/bash --with-mpc=/home/tcwg-builds
lave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg-build/target/arm-linux-gnueabi-hf/_build/build
s/destdir/x86_64-unknown-linux-gnu --with-mpfr=/home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-
trusty-amd64-tcwg-build/target/arm-linux-gnueabi-hf/_build/builds/destdir/x86_64-unknown-linux-gnu --with-gmp
=/home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg-build/target/arm-linux-gnue
abi-hf/_build/builds/destdir/x86_64-unknown-linux-gnu --with-gnu-as --with-gnu-ld --disable-libmudflap --enabl
e-lto --enable-objc-gc --enable-shared --without-included-gettext --enable-nls --disable-sjlj-exceptions --en
able-gnu-unique-object --enable-linker-build-id --disable-libstdc++-pch --enable-c99 --enable-clocale=gnu --e
nable-libstdc++-debug --enable-long-long --with-cloog=no --with-ppl=no --with-isl=no --disable-multilib --wit
h-float=hard --with-mode=thumb --with-tune=cortex-a9 --with-arch=armv7-a --with-fpu=vfpv3-d16 --enable-thread
s-posix --enable-multiarch --enable-libstdc++-time=yes --with-build-sysroot=/home/tcwg-buildslave/workspace/t
cwg-make-release/label/docker-trusty-amd64-tcwg-build/target/arm-linux-gnueabi-hf/_build/sysroots/arm-linux-gn
ueabi-hf --with-sysroot=/home/tcwg-buildslave/workspace/tcwg-make-release/label/docker-trusty-amd64-tcwg-build
/target/arm-linux-gnueabi-hf/_build/builds/destdir/x86_64-unknown-linux-gnu/arm-linux-gnueabi-hf/libc --enable-
checking=release --disable-bootstrap --enable-languages=c,c++,fortran,lto --build=x86_64-unknown-linux-gnu --
host=x86_64-unknown-linux-gnu --target=arm-linux-gnueabi-hf --prefix=/home/tcwg-buildslave/workspace/tcwg-make-
release/label/docker-trusty-amd64-tcwg-build/target/arm-linux-gnueabi-hf/_build/builds/destdir/x86_64-unknown-
linux-gnu
Thread model: posix
gcc version 4.9.4 (Linaro GCC 4.9-2017.01)
```

图 2.2-7 教程交叉编译器版本号为 4.9.4

```
gcc -v
allientek@ubuntu16:~/linux/tool/poky-arm-gcc$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/5/lto-wrapper
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 5.4.0-6ubuntu1-16.04.12' --with-bugurl=file://
/usr/share/doc/gcc-5/README.Bugs --enable-languages=c,ada,c++,java,go,d,fortran,objc,obj-c++ --prefix=/usr --
program-suffix=-5 --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --
enable-threads=posix --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --enable-libstdc++
-debug --enable-libstdc++-time=yes --with-default-libstdc++-abi=new --enable-gnu-unique-object --disable-vtab
le-verify --enable-libmpx --enable-plugin --with-system-zlib --disable-browser-plugin --enable-java-awt=gtk --
enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-5-amd64/jre --enable-java-home --with-jvm-root
-dir=/usr/lib/jvm/java-1.5.0-gcj-5-amd64 --with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-5-amd64 --wit
h-arch-directory=amd64 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --enable-multiarch --d
isable-werror --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-t
une=generic --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-
gnu
Thread model: posix
gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1-16.04.12)
```

图 2.2-8 Ubuntu16 的 gcc 版本号为 5.4

第三章 文件互传

3.1 Ubuntu 和 Windows 主机互传

前提: **Ubuntu 和 Windows 能 ping 通**。如何测试网络互通和搭建方法请看网络搭建手册。

3.1.1 filezilla

关于 filezilla 软件的使用方法,在驱动开发指南里有详细说明,这里不再赘述。主要讲解 samba 的使用。

3.1.2 samba 服务器

正点原子提供的虚拟机镜像中已经搭建好 samba 服务器,接下来说明下如何使用。

首先关闭电脑和 Ubuntu 的防火墙,Ubuntu 执行 `sudo ufw disable` 即可关闭防火墙。执行指令 `ifconfig` 可以查看 Ubuntu 的 IP 信息,这里笔者的 IP 地址为 192.168.1.29。

```
alientek@ubuntu16:~$ ifconfig
ens33  Link encap:以太网 硬件地址 00:0c:29:1c:81:85
        inet 地址:192.168.1.29 广播:192.168.1.255 掩码:255.255.255.0
        inet6 地址: fe80::9061:580f:a724:f0bd/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 跃点数:1
        接收数据包:108176 错误:0 丢弃:1 过载:0 帧数:0
        发送数据包:10532 错误:0 丢弃:0 过载:0 载波:0
        碰撞:0 发送队列长度:1000
        接收字节:22698501 (22.6 MB) 发送字节:950527 (950.5 KB)
```

图 3.1-1 查看虚拟机 IP 地址

回到 Windows 端,按下 Win 键和 R 键,弹出运行框,输入 `\\IP 地址`,点击确定。

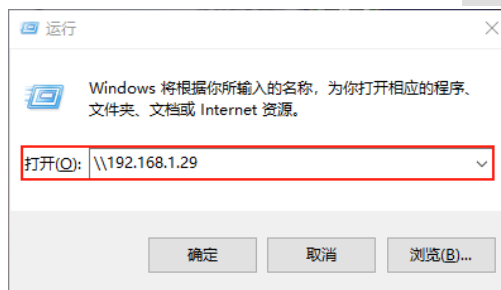


图 3.1-2 运行框输入 Ubuntu 的 IP 地址

然后输入连接凭据,即 Ubuntu 的用户名和密码即可连接到 Ubuntu。



图 3.1-3 输入 Ubuntu 用户名和密码

如果连接成功会出现一个虚拟机的 IP 地址的网络。没有的话请检查 Ubuntu 和 Windows 网络能否互通、防火墙是否关闭、用户名及密码是否正确。右键虚拟机网络，设置映射网络驱动器，如下图所示。（映射不是必须做的，只要能进入文件夹就行，这里笔者电脑为 windows10 系统，仅供参考）

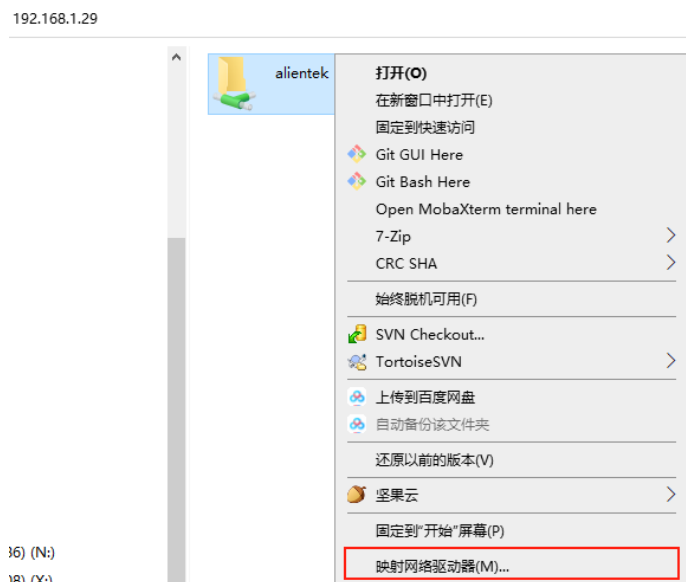


图 3.1-4 映射网络驱动器

选择一个驱动器，使用默认的即可，这里笔者默认为 W，点击完成。

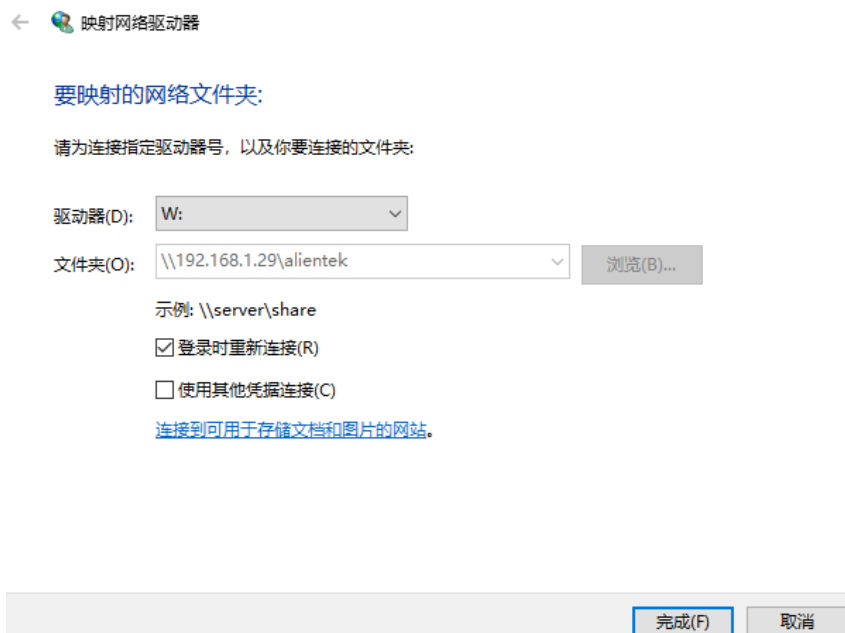


图 3.1-5 选择驱动器，点击完成

此时会弹出 Ubuntu 的目录，在“此电脑”里也可以看到这网络，如下图所示。如果虚拟机是关机状态的，则会显示下图中红叉的样式。即我们在使用 samba 的时候，要保证虚拟机是开启的状态。



图 3.1-6 samba

进入 alientek 盘, 屏蔽隐藏的项目, 即可看到和虚拟机用户主目录一样的目录。注意是用户主目录, 即~目录, 不是根目录。

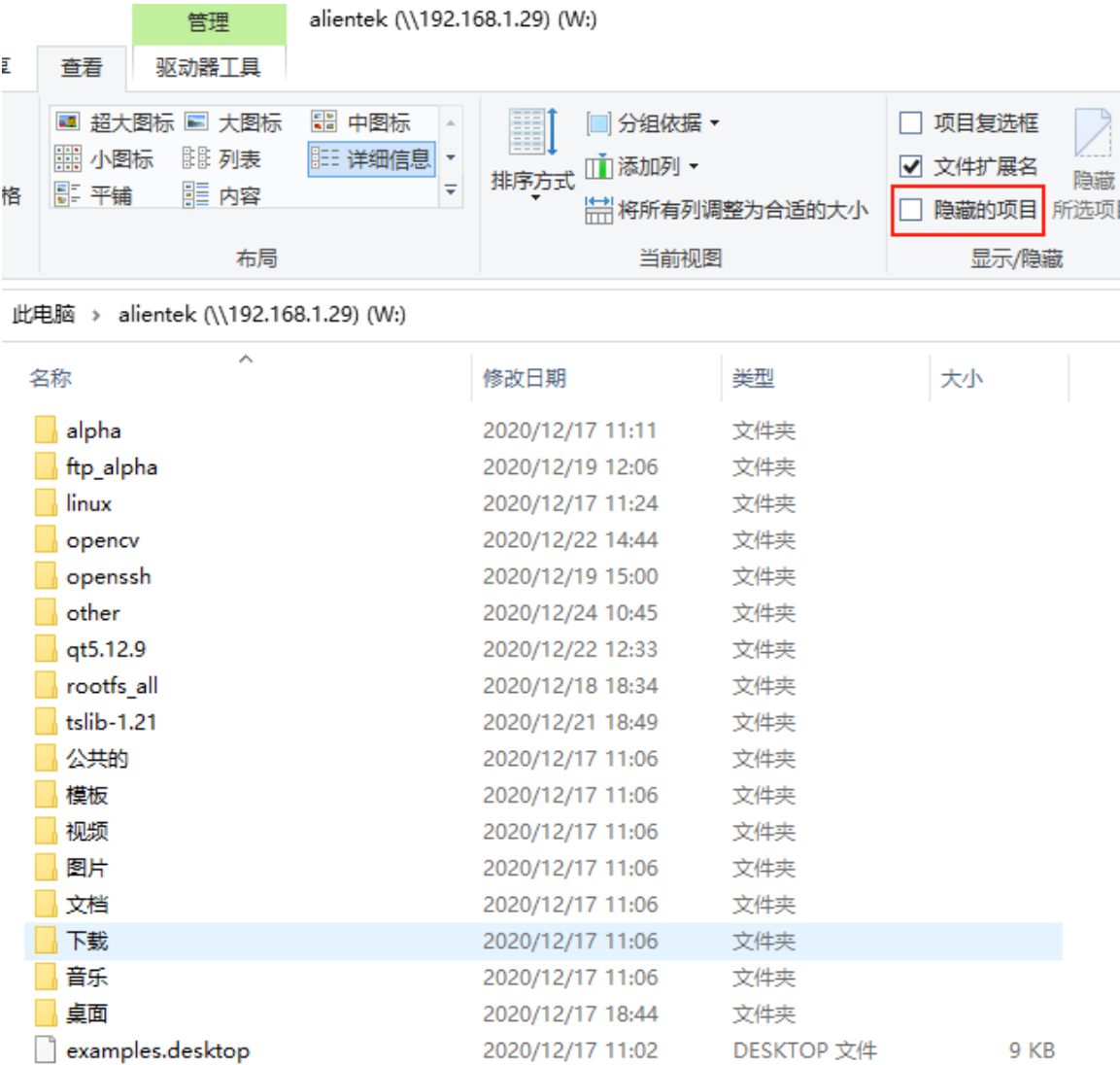


图 3.1-7 Windows 端下的虚拟机用户主目录

这样我们就可以直接拷贝 Windows 和 Ubuntu 端的文件进行互传了。

第四章 编译开发板出厂镜像

这里示范下编译出厂镜像, 因为出厂系统是持续维护的, 此虚拟机里的出厂镜像是 V1.4 版本的, 请根据开发板的实际情况, 下载最新的出厂镜像拷贝到虚拟机里编译, 编译方法如下。

4.1 编译出厂 uboot

首先去到出厂 uboot 源码路径下, 这里以虚拟机自带的为例子。

```
cd ~/alpha/alientek-alpha/uboot-alientek
```

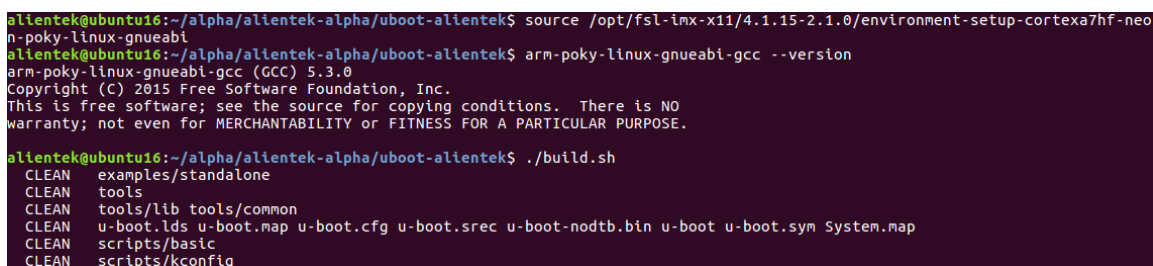
使能出厂交叉编译工具, 并查看版本号。

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

如果出现下图的 arm-poky-linux-gnueabi-gcc (GCC) 5.3.0, 即表示**当前终端**已经使能出厂交叉编译工具, 可以进行编译操作 (**注意**: 如果打开了新终端进行编译, 也需要先使能。即使能后只有此终端下才生效, 新终端无效, 需要重新使能)。

此时执行 build.sh 脚本即可开始编译出厂 uboot 源码。

```
./build.sh
```



```
alientek@ubuntu16:~/alpha/alientek-alpha/uboot-alientek$ source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neo
n-poky-linux-gnueabi
alientek@ubuntu16:~/alpha/alientek-alpha/uboot-alientek$ arm-poky-linux-gnueabi-gcc --version
arm-poky-linux-gnueabi-gcc (GCC) 5.3.0
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
alientek@ubuntu16:~/alpha/alientek-alpha/uboot-alientek$ ./build.sh
CLEAN examples/standalone
CLEAN tools
CLEAN tools/lib tools/common
CLEAN u-boot.lds u-boot.map u-boot.cfg u-boot.srec u-boot-nodtb.bin u-boot u-boot.syn System.map
CLEAN scripts/basic
CLEAN scripts/kconfig
```

图 4.1-1 编译出厂 uboot(旧图, v1.4 版本)

最终编译完成。



```
CFGFS board/freescale/mx6ullevk/imximage-ddr512.cfg.cfgtmp
MKIMAGE u-boot.imx
编译完成, 请查看当前目录下的tmp文件夹查看编译好的目标文件
alientek@ubuntu16:~/alpha/alientek-alpha/uboot-alientek$ cd tmp
alientek@ubuntu16:~/alpha/alientek-alpha/uboot-alientek/tmp$ ls
u-boot-imx6ull-14x14-ddr256-emmc.bin u-boot-imx6ull-14x14-ddr512-emmc.bin
u-boot-imx6ull-14x14-ddr256-emmc.imx u-boot-imx6ull-14x14-ddr512-emmc.imx
u-boot-imx6ull-14x14-ddr256-nand.bin u-boot-imx6ull-14x14-ddr512-nand.bin
u-boot-imx6ull-14x14-ddr256-nand.imx u-boot-imx6ull-14x14-ddr512-nand.imx
u-boot-imx6ull-14x14-ddr256-nand-sd.bin u-boot-imx6ull-14x14-ddr512-nand-sd.bin
u-boot-imx6ull-14x14-ddr256-nand-sd.imx u-boot-imx6ull-14x14-ddr512-nand-sd.imx
```

图 4.1-2 编译出厂 uboot 完成

4.2 编译出厂内核源码及设备树

首先去到出厂内核源码路径下, 这里以虚拟机自带的为例子。

```
cd ~/alpha/alientek-alpha/kernel-alientek
```

使能出厂交叉编译工具, 并查看版本号。

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

如果出现下图的 arm-poky-linux-gnueabi-gcc (GCC) 5.3.0, 即表示**当前终端**已经使能出厂交叉编译工具, 可以进行编译操作 (**注意**: 如果打开了新终端进行编译, 也需要先使能。即使能后只有此终端下才生效, 新终端无效, 需要重新使能)。

此时执行 build.sh 脚本即可开始编译出厂内核源码。

```
./build.sh

alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$ ls
arch      Documentation  ipc              MAINTAINERS     net             sound           vmlinux
block     drivers        Kbuild          Makefile         README          System.map     vmlinux.o
build.sh  firmware      Kconfig         modules.builtin  REPORTING-BUGS  tmp            tools
COPYING   fs             kernel          modules.order    samples         scripts        usr
CREDITS   include       lib             Module.symvers   security        security       virt
crypto    init          linux-imx-4.1.15-2.1.0-gad512fa-v1.7.tar.bz2
alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$ source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$ arm-poky-linux-gnueabi-gcc --version
arm-poky-linux-gnueabi-gcc (GCC) 5.3.0
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$ ./build.sh
CLEAN      .
CLEAN      arch/arm/kernel
CLEAN      arch/arm/vdso
CLEAN      drivers/tty/vt
CLEAN      drivers/video/logo
CLEAN      firmware
CLEAN      kernel/time
```

图 4.2-1 编译出厂内核

最终编译完成。

```
./4.1.15/modules.builtin.bin
./4.1.15/modules.softdep
/home/alientek/alpha/alientek-alpha/kernel-alientek
编译完成, 请查看当前目录下的tmp文件夹查看编译好的目标文件
alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$
alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$
alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$ cd tmp/
alien@ubuntu16:~/alpha/alientek-alpha/kernel-alientek/tmp$ ls
imx6ull-14x14-emmc-10.1-1280x800-c.dtb  imx6ull-14x14-nand-4.3-480x272-c.dtb
imx6ull-14x14-emmc-4.3-480x272-c.dtb    imx6ull-14x14-nand-4.3-800x480-c.dtb
imx6ull-14x14-emmc-4.3-800x480-c.dtb    imx6ull-14x14-nand-7-1024x600-c.dtb
imx6ull-14x14-emmc-7-1024x600-c.dtb     imx6ull-14x14-nand-7-800x480-c.dtb
imx6ull-14x14-emmc-7-800x480-c.dtb      imx6ull-14x14-nand-hdmi.dtb
imx6ull-14x14-emmc-hdmi.dtb             imx6ull-14x14-nand-vga.dtb
imx6ull-14x14-emmc-vga.dtb              modules.tar.bz2
imx6ull-14x14-nand-10.1-1280x800-c.dtb  zImage
```

图 4.2-2 编译出厂内核源码完成

4.3 编译出厂 Qt GUI 综合 Demo

首先去到出厂 Qt GUI 综合 Demo 源码路径下, 这里以虚拟机自带的为例子。

```
cd ~/alpha/alientek-alpha/qt-alientek/QDesktop/
```

使能出厂交叉编译工具, 并查看版本号。

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

如果出现下图的 arm-poky-linux-gnueabi-gcc (GCC) 5.3.0, 即表示**当前终端**已经使能出厂交叉编译工具, 可以进行编译操作 (**注意**: 如果打开了新终端进行编译, 也需要先使能。即使能后只有此终端下才生效, 新终端无效, 需要重新使能)。

执行 qmake, 生成 Makefile 用于编译。

```
qmake
ls Makefile -l
```

此时执行 make 即可编译出厂 Qt GUI 综合 Demo, 如果之前有编译过的, 可以先 clean 下。

```

make clean
make -j16
alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop/
alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-
-cortexa7hf-neon-poky-linux-gnueabi
alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ arm-poky-linux-gnueabi-gcc --version
arm-poky-linux-gnueabi-gcc (GCC) 5.3.0
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ qmake
alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ ls Makefile -l
-rw-rw-r-- 1 alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ make -j16
make: Nothing to be done for 'first'.
alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ make clean
rm -f qrc_qml.cpp
rm -f moc_predefs.h
rm -f moc_cameramedia.cpp moc_fileio.cpp moc_mymodel.cpp moc_weatherdata.cpp moc_lyricmodel.cpp moc_playlistmodel.cpp
moc_medialistmodel.cpp moc_wirelessmodel.cpp moc_mytcpclient.cpp moc_mytcpserver.cpp moc_desktop.cpp moc_udpreciver.
cpp moc_udpserver.cpp moc_photolistmodel.cpp moc_beep.cpp moc_ap3216c.cpp moc_icm20608.cpp moc_keythread.cpp moc_rad
io.cpp
rm -f cameramedia.o fileio.o main.o myjsonparse.o mymodel.o weatherdata.o lyricmodel.o playlistmodel.o medialistModel
.o wirelessmodel.o mytcpclient.o mytcpserver.o desktop.o udpreciver.o udpserver.o photolistmodel.o beep.o ap3216c.o icm
20608.o keythread.o radio.o qrc_qml.o moc_cameramedia.o moc_fileio.o moc_mymodel.o moc_weatherdata.o moc_lyricmodel.
o moc_playlistmodel.o moc_medialistModel.o moc_wirelessmodel.o moc_mytcpclient.o moc_mytcpserver.o moc_desktop.o moc
udpserver.o moc_udpserver.o moc_photolistmodel.o moc_beep.o moc_ap3216c.o moc_icm20608.o moc_keythread.o moc_radio.
o
rm -f *~ core *.core
alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ make -j16
arm-poky-linux-gnueabi-g++ -march=armv7ve -mfpu=neon -mfloat-abi=hard -mcpu=cortex-a7 --sysroot=/opt/fsl-imx-x11/4.
1.15-2.1.0/sysroots/cortexa7hf-neon-poky-linux-gnueabi -c -pipe --sysroot=/opt/fsl-imx-x11/4.1.15-2.1.0/sysroots/cort

```

图 4.3-1 编译出厂 QT

编译过和中有报警告，可忽略。编译完成如下图，编译得到的 QDesktop 文件，可以直接拷贝到出厂系统下直接执行 ./QDesktop 即可。

```

/opt/fsl-imx-x11/4.1.15-2.1.0/sysroots/cortexa7hf-neon-poky-linux-gnueabi/usr/lib/libQt5Network.so /opt/fsl-imx-x11/4.1.15-2.1.0/sysroots/cortexa7hf-neon-poky-linux-gnueabi/usr/lib/libQt5Positioning.so /opt/fsl-imx-x11/4.1.15-2.1.0/sysroots/cortexa7hf-neon-poky-linux-gnueabi/usr/lib/libQt5Core.so /opt/fsl-imx-x11/4.1.15-2.1.0/sysroots/cortexa7hf-neon-poky-linux-gnueabi/usr/lib/libGLv2.so -lpthread
alien@ubuntu16:~/alpha/alientek-alpha/qt-alientek/QDesktop$ ls
aflex      main.qml      moc_lyricmodel.o      moc_udpserver.o      radio
aircondition  Makefile      moc_medialistModel.cpp moc_weatherdata.cpp  radio.o
alarms     media         moc_medialistModel.o  moc_weatherdata.o    sensor
ap3216c.o  mediaListModel.o moc_mymodel.cpp      moc_wirelessmodel.cpp settings
beep.o     moc_ap3216c.cpp moc_mymodel.o        moc_wirelessmodel.o  src
calculator moc_ap3216c.o  moc_mytcpclient.cpp  music                 system
cameramedia moc_beep.cpp   moc_mytcpclient.o    myjsonparse.o        tcpclient
cameramedia.o moc_beep.o     moc_mytcpserver.cpp  mymodel.o             tcpserver
desktop    moc_cameramedia.cpp moc_mytcpserver.o    mytcpclient.o         udpchat
desktop.o  moc_cameramedia.o moc_photolistmodel.cpp mytcpserver.o         udpreciver.o
fileio.o   moc_desktop.cpp moc_photolistmodel.o photolistmodel.o      udpserver.o
fileview   moc_desktop.o  moc_playlistmodel.cpp photoview              weather
helpbutton moc_fileio.cpp moc_playlistmodel.o  playlistmodel.o       weatherdata.o
icm20608.o moc_fileio.o   moc_predefs.h        QDesktop               wireless
iotest     moc_icm20608.cpp moc_radio.o          QDesktop.pro           wirelessmodel.o
keythread.o moc_keythread.cpp moc_radio.o          qml.qrc
lyricmodel.o moc_keythread.o moc_udpserver.o      qrc_qml.o
main.cpp   moc_keythread.o moc_udpserver.o      qrc_qml.o
main.o     moc_lyricmodel.cpp moc_udpserver.o      qtquickcontrols2.conf

```

图 4.3-2 编译出厂 QT 完成

第五章 编译教程镜像

5.1 编译教程移植好的 uboot 源码

首先去到教程 uboot 源码路径下，这里以虚拟机自带的为例子。

```
cd ~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek/
```

查看教程交叉编译器版本号。

```
arm-linux-gnueabi-hf-gcc --version
```

执行编译脚本，emmc 版本的执行 `mx6ull_alientek_emmc.sh`；nand 版本的执行 `mx6ull_alientek_nand.sh`。这里以 emmc 版本为例。

```
./mx6ull_alientek_emmc.sh
```

```
alientek@ubuntu16:~$ cd ~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek/
alientek@ubuntu16:~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ ls
api      fs      Makefile      u-boot
arch     imxdownload  mx6ull_14x14_evk_emmc.sh  u-boot.bin
board    imxdownload.c  mx6ull_alientek_emmc.sh  u-boot.cfg
cmd      imxdownload.h  mx6ull_alientek_nand.sh  u-boot.code-workspace
common   include      net      u-boot.inx
config.mk  Kbuild      post     u-boot.lds
configs   Kconfig     README   u-boot.map
disk      lib         scripts  u-boot.nodtb.bin
doc       Licenses    snapshot.commit  u-boot.srec
drivers   load.imx    System.map  u-boot.sym
dts       MAINTAINERS test

alientek@ubuntu16:~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ arm-linux-gnueabi-hf-gcc --version
arm-linux-gnueabi-hf-gcc (Linaro GCC 4.9-2017.01) 4.9.4
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
alientek@ubuntu16:~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ ./mx6ull_alientek_emmc.sh
CLEAN examples/standalone
CLEAN tools
CLEAN tools/lib tools/common
```

图 5.1-1 编译教程 uboot

编译完成，会生成 u-boot.imx、u-boot.bin 等文件。

```
COPY u-boot.bin
CFGs board/freescale/mx6ull_alientek_emmc/imximage.cfg.cftgmp
MKIMAGE u-boot.imx

alientek@ubuntu16:~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$
alientek@ubuntu16:~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ ls
api      disk      imxdownload.c  load.imx      net      tools      u-boot.map
arch     doc       imxdownload.h  MAINTAINERS   post     u-boot     u-boot.nodtb.bin
board    drivers   include      MAKEALL       README   u-boot.bin  u-boot.srec
cmd      dts       Kbuild      Makefile      scripts  u-boot.cfg  u-boot.sym
common   examples  Kconfig     mx6ull_14x14_evk_emmc.sh  snapshot.commit  u-boot.code-workspace
config.mk fs        lib         mx6ull_alientek_emmc.sh  System.map      u-boot.imx
configs  imxdownload  Licenses    mx6ull_alientek_nand.sh  test            u-boot.lds

alientek@ubuntu16:~/alpha/alientek-zzk/uboot-zzk/uboot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ ls u-boot.imx -l
-rw-rw-r-- 1 alientek alientek 429056 3月  6 17:28 u-boot.imx
```

图 5.1-2 编译教程 uboot 成功

将 TF 卡接入到虚拟机中，使用 `imxdownload` 工具烧写 u-boot.bin 文件到 TF 卡，使用 TF 卡来启动开发板测试。将 TF 卡接读卡器，接到电脑的 USB 接口，在 VMware 菜单栏的“虚拟机”->“可移动设备”->你的 USB 设备名（这里笔者为 Card Reader），连接设备到虚拟机。

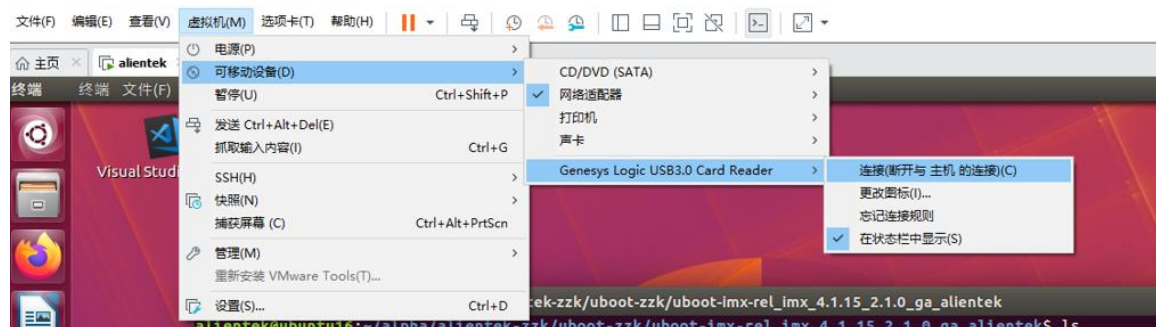


图 5.1-3 TF 卡接入虚拟机

在终端查看下盘符。

```
ls /dev/sd*
```

```
allientek@ubuntu16:~/alpha/alientek-zzk/u-boot-zzk/u-boot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ ls /dev/sd*
/dev/sda /dev/sda1 /dev/sdb /dev/sdb1
```

图 5.1-4 查看 TF 卡盘符

这里笔者 TF 卡盘符为/dev/sdb 和/dev/sdb1。其中, /dev/sdb 为 TF 卡的设备名, /dev/sdb1 为 TF 卡的分区。我们烧写裸机程序和 u-boot 都是烧写到卡设备中, 因此使用 imxdownload 工具都是烧写到/dev/sdb, 而不是/dev/sdb1。

执行以下指令烧录 u-boot.bin 到 TF 卡设备中。

```
./imxdownload u-boot.bin /dev/sdb
```

```
allientek@ubuntu16:~/alpha/alientek-zzk/u-boot-zzk/u-boot-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ ./imxdownload u-boot.bin /dev/sdb
I.MX6ULL bin download software
Edit by:zuozechongkat
Date:2019/6/10
Version:V1.1
log:V1.0 initial version,just support 512MB DDR3
V1.1 and support 256MB DDR3
file u-boot.bin size = 424596Bytes
Board DDR SIZE: 512MB
Delete Old load.imx
Create New load.imx
Download load.imx to /dev/sdb .....
[sudo] alientek 的密码:
记录了835+1 的读入
记录了835+1 的写出
427668 bytes (428 kB, 418 KiB) copied, 1.06997 s, 400 kB/s
```

图 5.1-5 烧写 u-boot 到 TF 卡

烧写完成, 将可移动设备断开连接, 把 TF 卡接到开发板上, 开发板拨码开关选择 SD 卡启动, 在串口终端可以看到 u-boot 启动信息, 可以看到编译 u-boot 的时间。

```
U-Boot 2016.03 (Mar 06 2021 - 17:28:11 +0800)

CPU: Freescale i.MX6ULL rev1.1 69 MHz (running at 396 MHz)
CPU: Industrial temperature grade (-40C to 105C) at 35C
Reset cause: POR
Board: MX6ULL ALIENTEK EMMC
I2C: ready
DRAM: 512 MiB
MMC: FSL_SDHC: 0, FSL_SDHC: 1
*** Warning - bad CRC, using default environment

Display: TFT7016 (1024x600)
Video: 1024x600x24
In: serial
Out: serial
Err: serial
switch to partitions #0, OK
mmc0 is current device
Net: FEC1
Error: FEC1 address not set.

Normal Boot
Hit any key to stop autoboot: 0
=>
```

图 5.1-6 u-boot 启动信息

5.2 编译教程内核源码

首先去到教程内核源码路径下, 这里以虚拟机自带的为例子。

```
cd ~/alpha/alientek-zzk/kernel-zzk/linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek
```

查看教程交叉编译器版本号。

```
arm-linux-gnueabi-hf-gcc --version
```

执行编译脚本, emmc 版本的执行 `mx6ull_alientek_emmc.sh`; nand 版本的执行 `mx6ull_alientek_nand.sh`。这里以 emmc 版本为例。

```
./mx6ull_alientek_emmc.sh
```

```
alientek@ubuntu16:~$ cd ~/alpha/alientek-zzk/kernel-zzk/linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek
alientek@ubuntu16:~/alpha/alientek-zzk/kernel-zzk/linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ arm-linux-gnueabi-gcc --version
arm-linux-gnueabi-gcc (Linaro GCC 4.9-2017.01) 4.9.4
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
alientek@ubuntu16:~/alpha/alientek-zzk/kernel-zzk/linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ ./mx6ull_alientek_emmc.sh
CLEAN .
CLEAN arch/arm/kernel
CLEAN arch/arm/vdso
CLEAN drivers/tty/vt
```

图 5.2-1 编译教程内核源码

编译过程中会弹出图形配置选项, 按两下 Esc 键即可关闭。

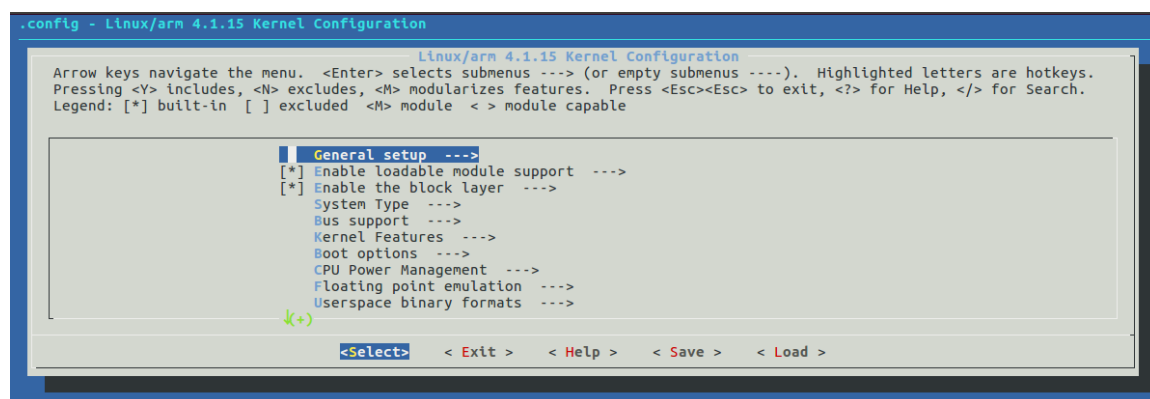


图 5.2-2 图形配置选项

最终编译成功会生成 zImage 镜像和 imx6ull-alientek-emmc.dtb 设备树文件 (如果是 nand 的脚本则会生成 imx6ull-alientek-nand.dtb 设备树文件)

```
LD [M] lib/crc7.ko
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
alientek@ubuntu16:~/alpha/alientek-zzk/kernel-zzk/linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek$ cd arch/arm/boot/
alientek@ubuntu16:~/alpha/alientek-zzk/kernel-zzk/linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek/arch/arm/boot$ ls
bootp compressed dts Image install.sh Makefile zImage
alientek@ubuntu16:~/alpha/alientek-zzk/kernel-zzk/linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek/arch/arm/boot$ ls dts/imx6ull-alientek-emmc.dtb -l
-rw-rw-r-- 1 alientek alientek 39280 3月 6 18:12 dts/imx6ull-alientek-emmc.dtb
```

图 5.2-3 编译内核成功

生成的 zImage 文件在 arch/arm/boot 路径下, 生成的设备树文件在 arch/arm/boot/dts 路径下。

第六章 网络挂载教程镜像

6.1 配置开发板 uboot 的网络信息

前提: 开发板 ENET2 网线直连电脑。请先按照《【正点原子】I.MX6U 网络环境 TFTP&NFS 搭建手册》先搭建好网络环境。

开机进入 uboot 命令行, 设置开发板的网络信息, 这里笔者设置信息如下, 用户请根据自己实际情况来设置。

```
setenv ipaddr 192.168.1.120
setenv ethaddr 00:04:9f:04:d2:35
setenv gatewayip 192.168.1.1
setenv netmask 255.255.255.0
setenv serverip 192.168.1.29
saveenv
```

ipaddr: 开发板 IP 地址, 根据实际情况填写, 要和虚拟机在同一网段, 且不发生 IP 冲突。

ethaddr: 开发板网络设备地址, 填写 00:04:9f:04:d2:35 即可。

gatewayip: 开发板和 Ubuntu 局域网的网关, 根据实际情况填写。

netmask: 子网掩码。一般为 255.255.255.0。

serverip: 虚拟机 IP 地址, 根据实际情况填写。

最后开发板能 ping 通虚拟机 IP 地址即可。

```
ping 192.168.1.29
```

```
Net: FEC1
Error: FEC1 address not set.

Normal Boot
Hit any key to stop autoboot: 0
=> setenv ipaddr 192.168.1.120
=> setenv ethaddr 00:04:9f:04:d2:35
=> setenv gatewayip 192.168.1.1
=> setenv netmask 255.255.255.0
=> setenv serverip 192.168.1.165
=> saveenv
Saving Environment to MMC...
Writing to MMC(0)... done
=> ping 192.168.1.29
FEC1 Waiting for PHY auto negotiation to complete.... done
Using FEC1 device
host 192.168.1.29 is alive
```

图 6.1-1 设置开发板 uboot

6.2 TFTP 加载 zImage 和设备树

正点原子虚拟机镜像中已经搭建好 TFTP 环境, TFTP 工作目录为 `~/linux/tftp/`。如果想加载自己的镜像文件, 只要将镜像文件和设备树拷贝到此目录下, 赋予文件可执行权限即可, `bootcmd` 中设置对应的文件名。**注意 bootz 中的 - 符号是英文的, 两边各有一个空格, 示例中用绿色标注。**

示例: 在开发板 uboot 命令行中, 设置 `bootcmd` 信息, 用于引导网络挂载虚拟机里的文件。

```
setenv bootcmd 'tftp 80800000 zImage;tftp 83000000 imx6ull-alientek-emmc.dtb;bootz 80800000 - 83000000'
saveenv
```

```
=> setenv bootcmd 'tftp 80800000 zImage;tftp 83000000 imx6ull-alientek-emmc.dtb;bootz 80800000 - 83000000'
=> saveenv
Saving Environment to MMC...
Writing to MMC(0)... done
=>
```

```
fec 20b4000.ethernet eth0: Freescale FEC PHY driver [SMSC LAN8710/LAN8720] (miibus:phy_addr=20b4000.ethernet:01, irq=-1)
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
fec 20b4000.ethernet eth0: Link is Up - 100Mbps/Full - flow control rx/tx
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
IP-Config: Complete:
    device=eth0, hwaddr=00:04:9f:04:d2:35, ipaddr=192.168.1.120, mask=255.255.255.0, gw=192.168.1.1
    host=192.168.1.120, domain=, nis-domain=(none)
    bootserver=192.168.1.29, rootserver=192.168.1.29, rootpath=
gpio dvfs: disabling
can-3v3: disabling
ALSA device list:
    No soundcards found.
VFS: Mounted root (nfs filesystem) on device 0:14.
devtmpfs: mounted
Freeing unused kernel memory: 404K (809a1000 - 80a06000)
ALSA: Restoring mixer setting.....

Please press Enter to activate this console. /sbin/alsactl: load_state:1735: No soundcards found...
random: nonblocking pool is initialized
/var/empty must be owned by root and not group or world-writable.
ls
/ #
/ # cd /
/ # ls
bin          include      proc         tmp
dev          lib          root         usr
drivers      linuxrc      rootfs.tar.bz2 var
etc          minicom.log sbin         video
gdbtest      mnt          share        中文测试
home         music        sys
```

图 6.3-2 挂载文件系统

第七章 网络挂载出厂镜像

7.1 TFTP 挂载出厂内核和设备树

拷贝出厂内核到 TFTP 目录。

```
cd ~/alpha/alientek-alpha/kernel-alientek/arch/arm/boot/
```

```
cp zImage ~/linux/tftp/
```

根据自己开发板和屏幕的型号, 拷贝对应出厂设备树到 TFTP 目录, 这里笔者开发板为 emmc 版本, 屏幕为 4.3 寸 800x480 分辨率的屏幕。如果没有屏幕, 也选择这个设备树来启动。

```
cp dts/imx6ull-14x14-emmc-4.3-800x480-c.dtb ~/linux/tftp/
```

```
alientek@ubuntu16:~/alpha/alientek-alpha/kernel-alientek$ cd ~/alpha/alientek-alpha/kernel-alientek/arch/arm/boot/
alientek@ubuntu16:~/alpha/alientek-alpha/kernel-alientek/arch/arm/boot$
alientek@ubuntu16:~/alpha/alientek-alpha/kernel-alientek/arch/arm/boot$ cp dts/imx6ull-14x14-emmc-4.3-800x480-c.dtb ~/linux/tftp/
alientek@ubuntu16:~/alpha/alientek-alpha/kernel-alientek/arch/arm/boot$ cp zImage ~/linux/tftp/
```

图 7.1-1 拷贝出厂内核和设备树到 TFTP 目录

启动开发板, 在 uboot 界面设置 bootcmd 来引导启动挂载的出厂系统镜像。注意 bootz 中的 - 符号是英文的, 两边各有一个空格, 示例中用绿色标注。

```
setenv bootcmd 'tftp 80800000 zImage;tftp 83000000 imx6ull-14x14-emmc-4.3-800x480-c.dtb;bootz 80800000 - 83000000'
```

```
saveenv
```

```
Hit any key to stop autoboot: 0
=> setenv bootcmd 'tftp 80800000 zImage;tftp 83000000 imx6ull-14x14-emmc-4.3-800x480-c.dtb;bootz 80800000 - 83000000'
=> saveenv
Saving Environment to MMC...
Writing to MMC(0)... done
```

图 7.1-2 开发板 uboot 设置 bootcmd

7.2 NFS 挂载出厂文件系统

拷贝出厂文件系统到 NFS 目录。

```
cd ~/alpha/alientek-alpha/
```

```
cp rootfs-alientek/ ~/linux/nfs/ -vr
```

拷贝完在 NFS 共享目录下有这个文件系统。

```
alientek@ubuntu16:~$ cd linux/nfs/
alientek@ubuntu16:~/linux/nfs$ ls
buildrootfs  mytest.c  rootfs  rootfs-alientek  ubuntu_rootfs
alientek@ubuntu16:~/linux/nfs$
```

图 7.2-1 NFS 下的出厂系统

在 uboot 界面设置 bootargs 来引导启动挂载的出厂系统。

```
setenv bootargs 'console=ttyMXC0,115200 root=/dev/nfs\
nfsroot=192.168.1.29:/home/alientek/linux/nfs/rootfs-alientek,proto=tcp rw\
ip=192.168.1.120:192.168.1.29:192.168.1.1:255.255.255.0::eth0:off'

saveenv
boot
```

在 LCD 屏幕上可以看到启动的 LOGO 和 QT 界面, 在串口终端可以看到文件系统启动。

```
Starting FTP Server: vsftpd... done.  
Starting crond: OK  
Running local boot scripts (/etc/rc.local).  
  
root@ATK-IMX6U:~# ls  
driver shell  
root@ATK-IMX6U:~# cd /  
root@ATK-IMX6U:~# ls  
bin boot dev etc home lib media mnt opt proc rootfs.tar.bz2 run sbin sys tmp usr var 文件系统版本说明.txt  
root@ATK-IMX6U:~#
```

图 7.2-2 串口打印信息

第八章 Qt 相关

8.1 Qt Creator

推荐使用脚本打开 Qt Creator。使用如下指令打开 Qt Creator。“&”的作用是后台运行。
(推荐使用这种方式打开 Qt Creator)

```
/opt/Qt5.12.9/Tools/QtCreator/bin/qtcreator.sh &
```

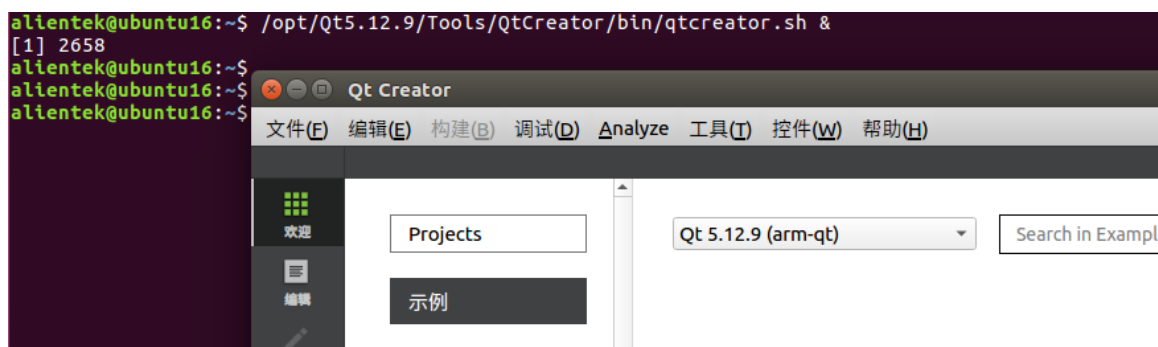


图 8.1-1 使用指令打开 Qt Creator

点击菜单栏的【工具】->【选项】->【Kits】，可以看见有以下 4 种编译套件。

Desktop Qt 5.12.9 GCC 64bit: 用于编译 Qt 程序在本 Ubuntu 上运行。

ATK-I.MX6U: 用于编译 Qt 程序在开发板出厂系统上运行（适配 Qt 教程）

Arm-Qt5.12.9: 用于编译 Qt 程序在移植好 Qt 环境的 busybox 文件系统中运行

IMX6U-rsync: 用于编译 Qt 程序在开发板出厂系统上进行远程调试（适配部分 Qt 教程）



图 8.1-2 四种套件

8.2 虚拟机上运行出厂系统 Qt GUI 界面

使用 Qt 5.12.9 GCC 64bit 套件可以编译出 Qt 程序运行在当前虚拟机上，以下用出厂系统 Qt 综合源码为示例。

8.2.1 导入出厂 Qt 工程

打开 Qt Creator，导入出厂系统源码。

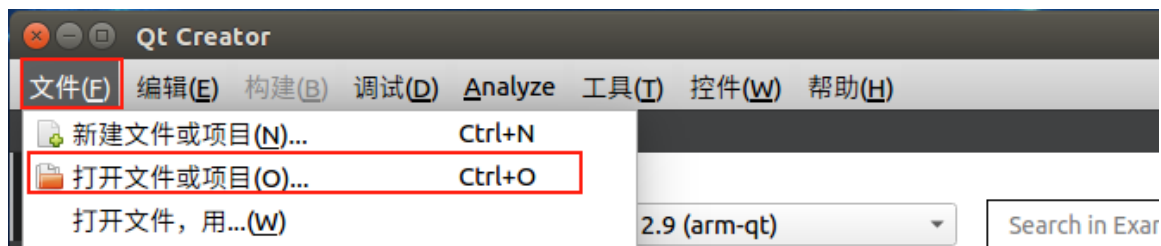


图 8.2-1 打开项目

找到虚拟机中存放出厂系统 Qt 综合源码的路径, 这里为 `~/alpha/alientek-alpha/qt-alientek/QDesktop`, 选择 `QDesktop.pro` 并打开。

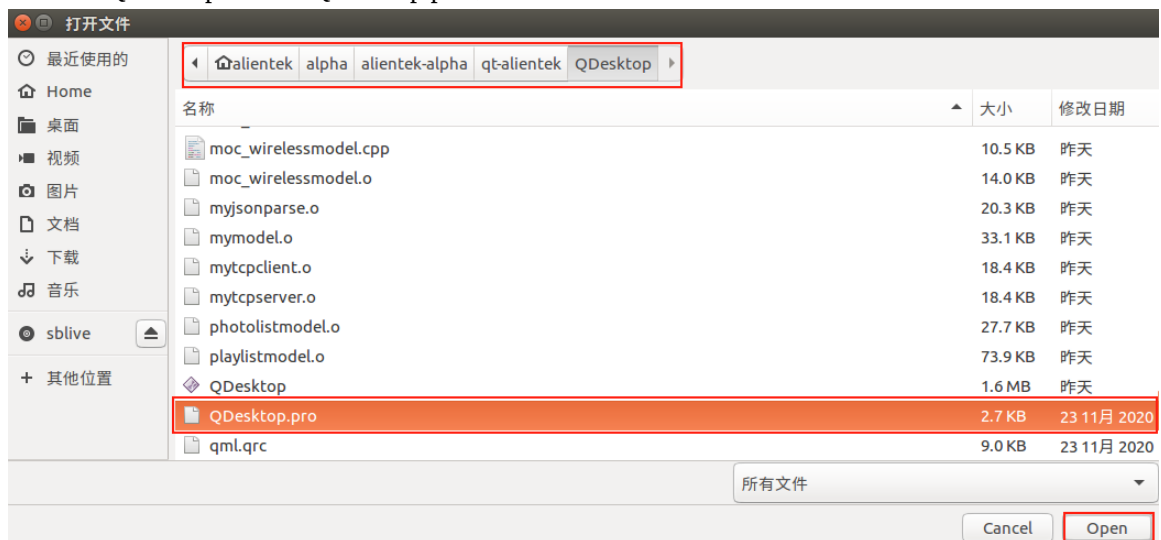


图 8.2-2 打开源码中的 QDesktop.pro

因为是要运行在虚拟机上看效果, 我们选择使用 Desktop Qt 5.12.9 GCC 64bit 套件, 并点击导入工程。

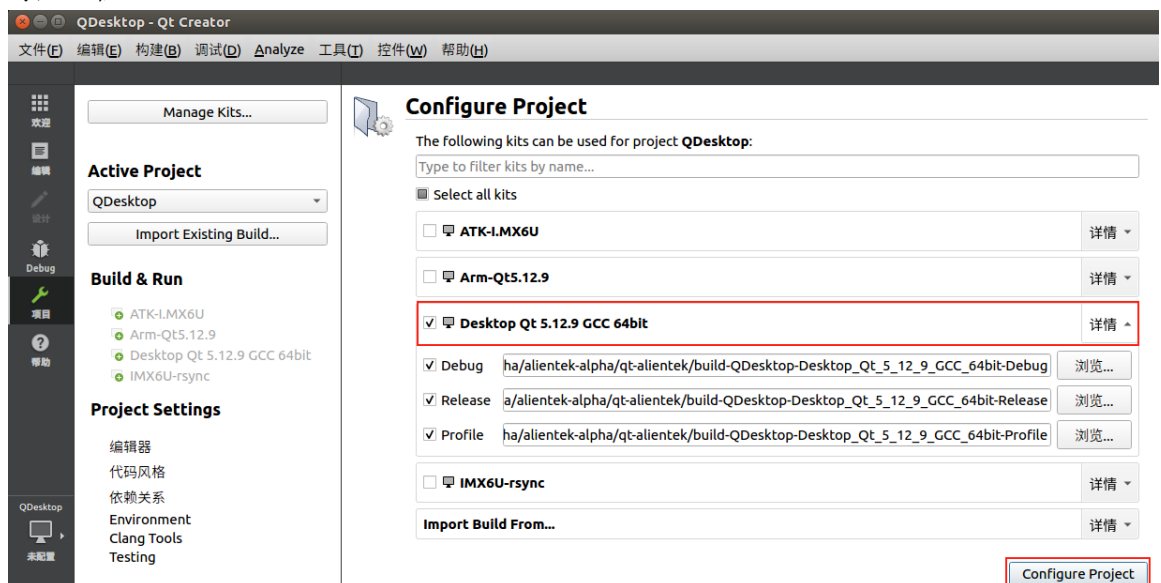


图 8.2-3 使用 Desktop Qt 5.12.9 GCC 64bit

等待工程导入完毕, 选择使用 Desktop Qt 5.12.9 GCC 64bit 套件来编译程序, 再点击构建编译。如下图所示。在编译之前, 请执行指令安装 libgl1-mesa-dev 库, 不然可能会报 -IGL 错误。

```
sudo apt-get install libgl1-mesa-dev
```

确认构建套件无误后, 点击  按钮就可以构建并运行程序。

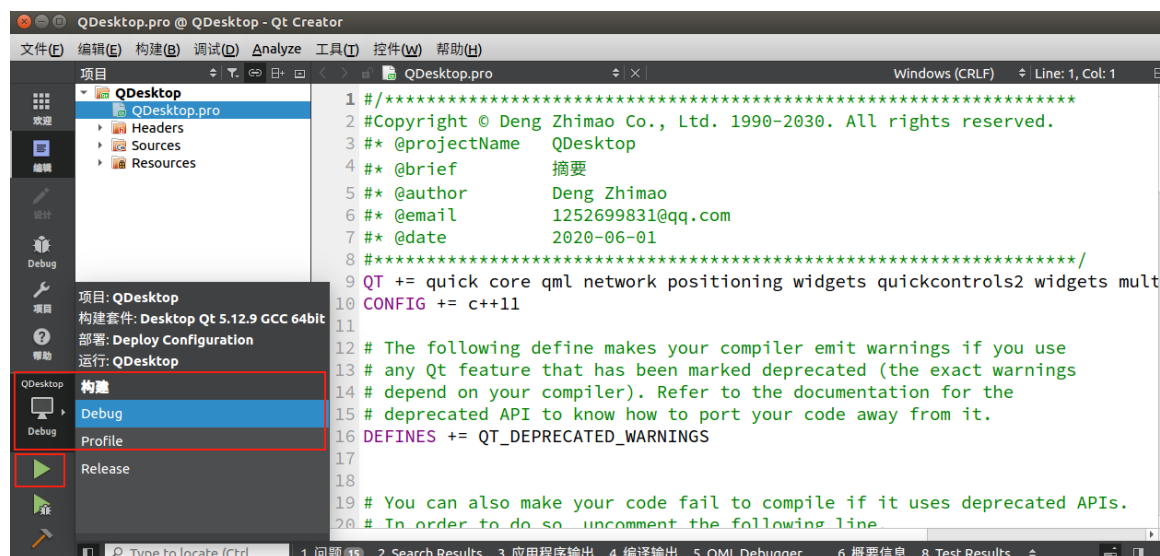


图 8.2-4 使用 Desktop Qt 5.12.9 GCC 来编译

编译过程中会有一些警告, 这些不用理会。如果编译失败, 有:-1: error: cannot find -IGL 错误, 请重新检查下是否安装了 libgl1-mesa-dev 库, 安装完后重新编译。

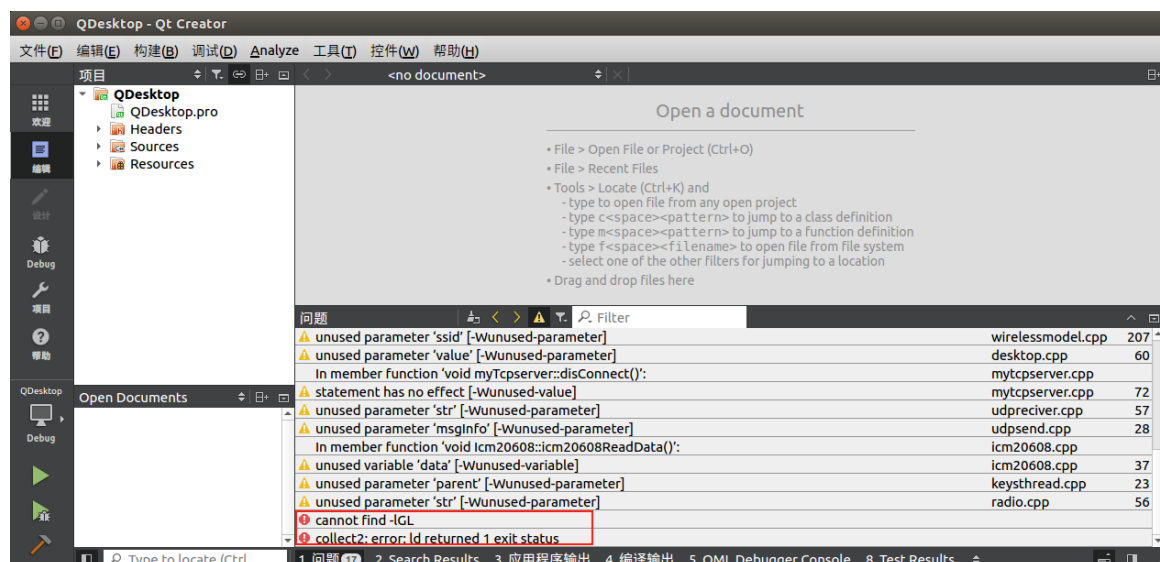


图 8.2-5 error: cannot find -IGL

编译成功后, 效果如下图所示。默认是全屏效果, 这里笔者缩放了窗口。



图 8.2-6 编译效果

至此，虚拟机上已经可以显示出系统 Qt 的 GUI 界面，构建完成会在 QDesktop 同级目录下生成一个 build-QDesktop-Desktop_Qt_5_12_9_GCC_64bit-Debug 目录。

```
alientek@ubuntu16:~/alpha/alientek-alpha/qt-alientek$ ls
build-QDesktop-Desktop_Qt_5_12_9_GCC_64bit-Debug  QDesktop  test
alientek@ubuntu16:~/alpha/alientek-alpha/qt-alientek$
```

图 8.2-7 build-QDesktop-Desktop_Qt_5_12_9_GCC_64bit-Debug

8.2.2 显示图片

单单显示 GUI 界面还不行，还要处理一些细节方面的事情。没有图片或者文件的话，需要把 QDesktop 源码下的 src 文件夹拷贝到 build-QDesktop-Desktop_Qt_5_12_9_GCC_64bit-Debug 文件夹下。

```
cd ~/alpha/alientek-alpha/qt-alientek/QDesktop/
cp -r src/ ../build-QDesktop-Desktop_Qt_5_12_9_GCC_64bit-Debug
```

然后重新构建项目即可，点击 GUI 里的相簿。

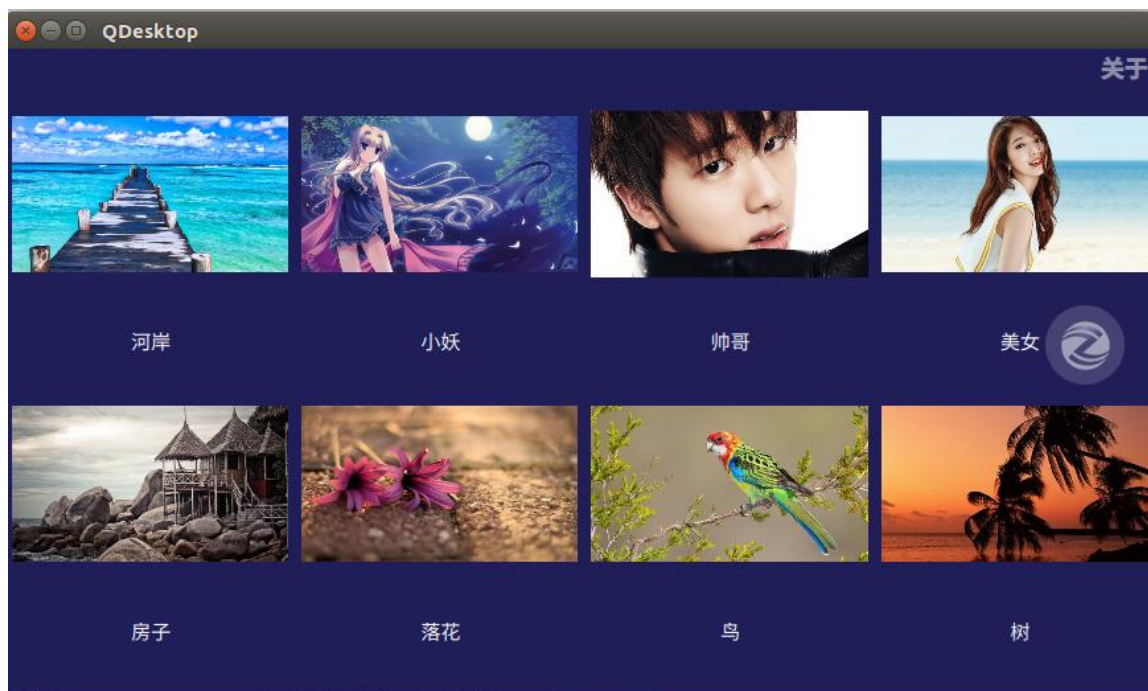


图 8.2-8 相簿

8.2.3 播放音频

Ubuntu 需要安装 Gstreamer 才能播放视频和音乐。执行以下指令。

```
sudo apt-get install gstreamer1.0-plugins-base gstreamer1.0-plugins-good  
sudo apt-get install gstreamer1.0-plugins-ugly gstreamer1.0-pulseaudio
```

安装完成后, 重新构建编译出厂 Qt 工程后就可以播放音频。

可以把耳机接入电脑, 如果虚拟机没有声音, 还需要检查以下配置。首先是设置虚拟机声卡连接, 即把耳机接入虚拟机。

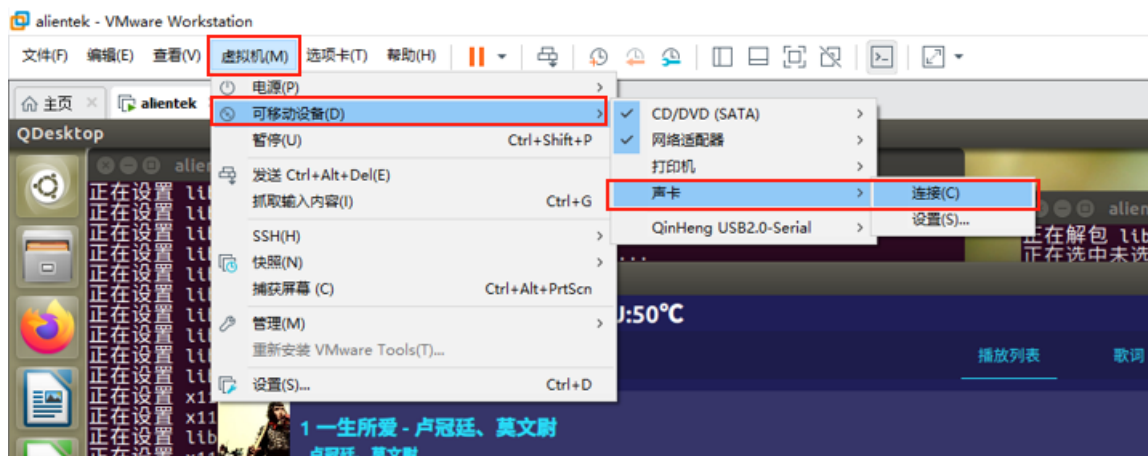


图 8.2-9 设置虚拟机声卡

设置虚拟机音量, 可能虚拟机当前音量过低或者没用音量。

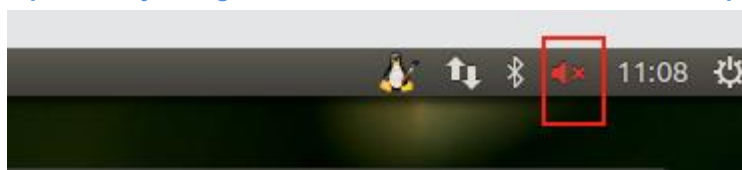


图 8.2-10 当前虚拟机为静音

设置成如下。



图 8.2-11 设置虚拟机音量

第九章 在出厂系统上运行 Qt 教程源码

使用 Qt creator 里的 ATK-I.MX6U 套件可以编译 Qt 教程的程序,生成可执行文件,运行在正点原子 I.MX6ULL 开发板的出厂系统环境上。出厂系统上集成了 Qt 环境,Qt creator 启动脚本中添加了出厂交叉编译器环境变量,所以二者环境以及搭建适配好,可以直接使用。

因为 Qt 在 Ubuntu 和开发板的存在样式差异,同一个 Qt 程序,在 Ubuntu 和嵌入式开发板屏幕上显示多少会有点区别,请以实际为准。这里以 05_opencv_camera 为例,其他的工程也是一样的方法。

9.1 关闭出厂系统桌面

首先我们要挂载出厂系统或者直接启动开发板上的出厂系统,设置退出桌面。因为我们如果要运行 Qt 程序的话,会和出厂系统原有的桌面冲突。

点击出厂 Qt GUI 界面的“设置”,点击“退出桌面”。这个只是暂时退出桌面,重启后还会有桌面的,如果想永久关闭出厂系统桌面,请参考《【正点原子】I.MX6U 用户快速体验》5.3 小节的方法。

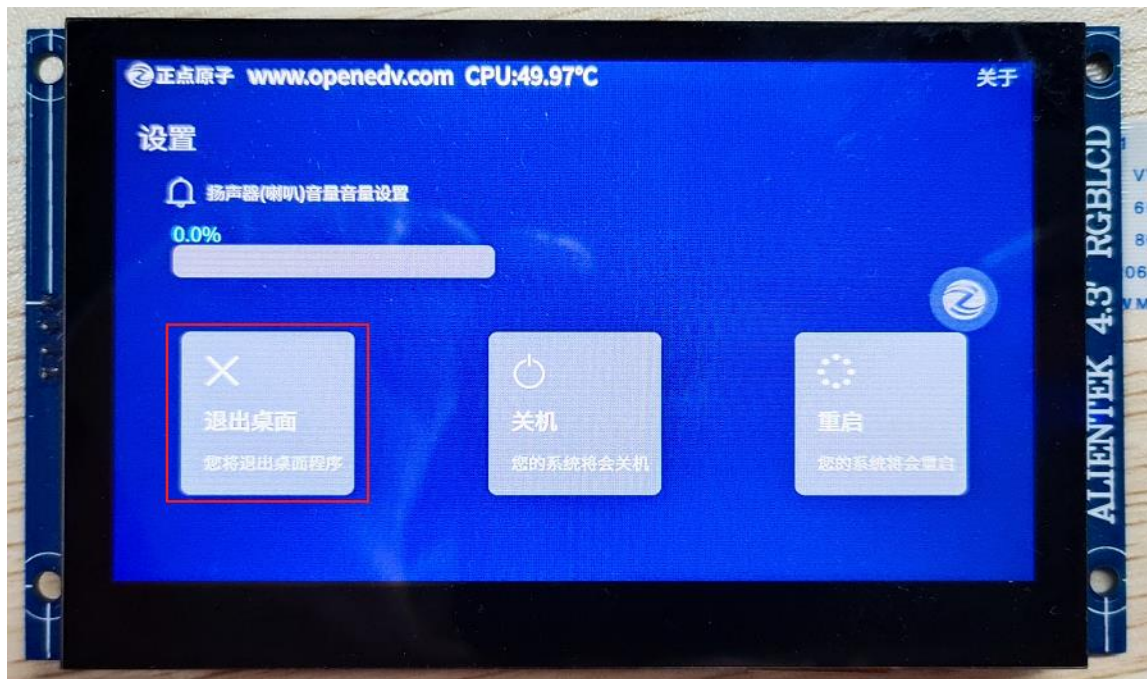


图 9.1-1 退出桌面

9.2 使能出厂交叉编译器环境变量

Qt 教程是基于出厂系统环境下去操作的,所以我们在编译程序之前要确保 ubuntu 中已经安装了 arm-poky-linux-gnueabi-gcc。使用以下指令使能环境变量后,可以查看是否安装。

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

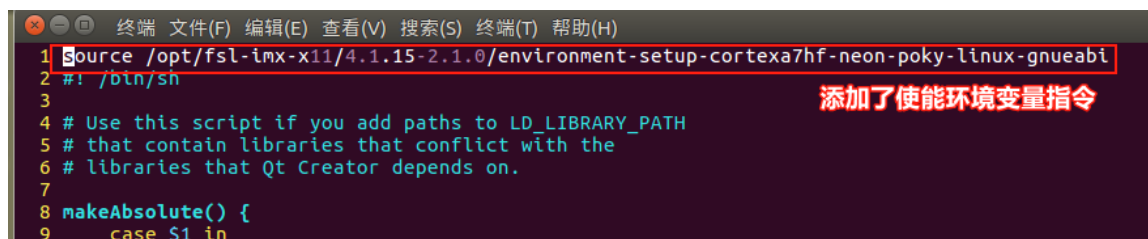
```
alientek@ubuntu16:~$ source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
alientek@ubuntu16:~$ arm-poky-linux-gnueabi-gcc --version
arm-poky-linux-gnueabi-gcc (GCC) 5.3.0
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

图 9.2-1 使能并查看出厂交叉编译器

使能环境后, 就可以执行指令打开 Qt creator。

`/opt/Qt5.12.9/Tools/QtCreator/bin/qtcreator.sh &`

在本文档之前的 V1.1 版本中, Qt creator 启动脚本是没有使能环境变量的指令的, 需要手动执行使能指令, 在 V1.2 版本的系统里添加了使能指令, 可以使用 vi 打开上面的 `qtcreator.sh` 查看。如果不是运行在出厂系统上的 Qt 程序, 可以将此指令屏蔽。



```
1 source /opt/fsl-imx-x11/4.1.15-2.1.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
2 #! /bin/sh
3
4 # Use this script if you add paths to LD_LIBRARY_PATH
5 # that contain libraries that conflict with the
6 # libraries that Qt Creator depends on.
7
8 makeAbsolute() {
9     case $1 in
```

图 9.2-2 Qt 启动脚本中添加了使能环境变量指令

9.3 编译运行 Qt 摄像头例程

打开文件或项目。

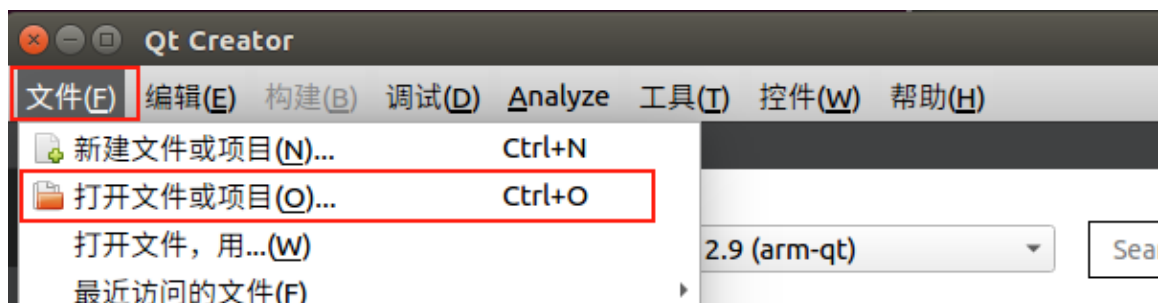


图 9.3-1 打开文件或项目

如图所示, 找到 05_opencv_camrea.pro 文件打开。

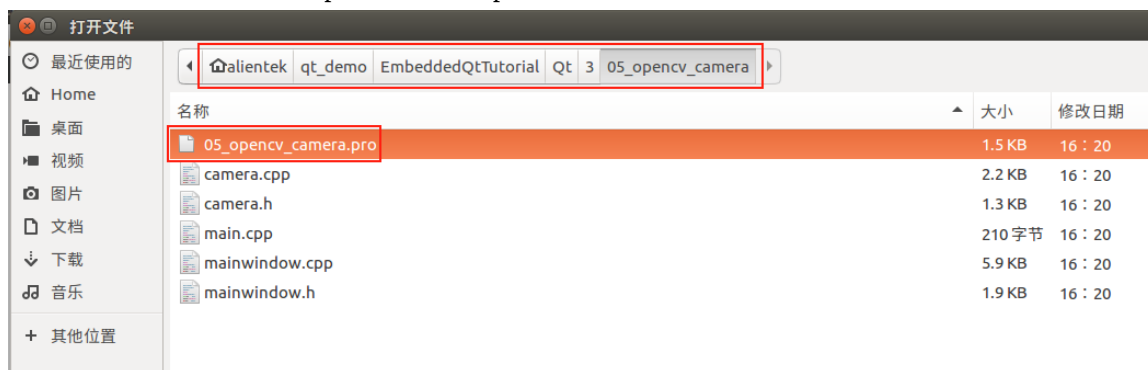


图 9.3-2 打开对应的工程文件

选择对应的构建套件, 这里演示 Desktop Qt 5.12.9、ATK-I.MX6U 和 IMX6U-rsync 三种。开发板使用的是出厂环境。

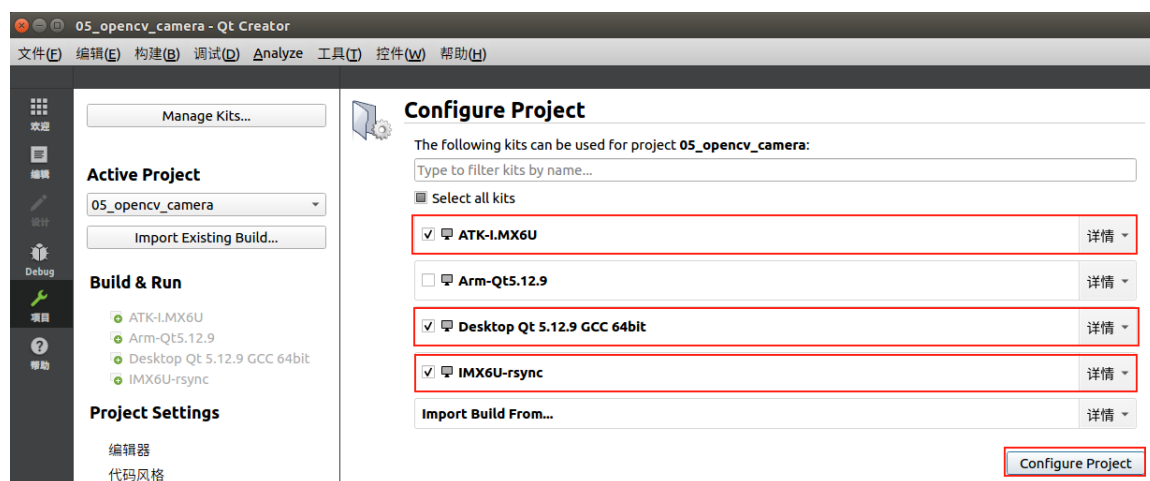


图 9.3-3 选择构建套件

9.3.1 Qt 程序运行在 Ubuntu 上

选择 Desktop Qt 5.12.9 构建套件。

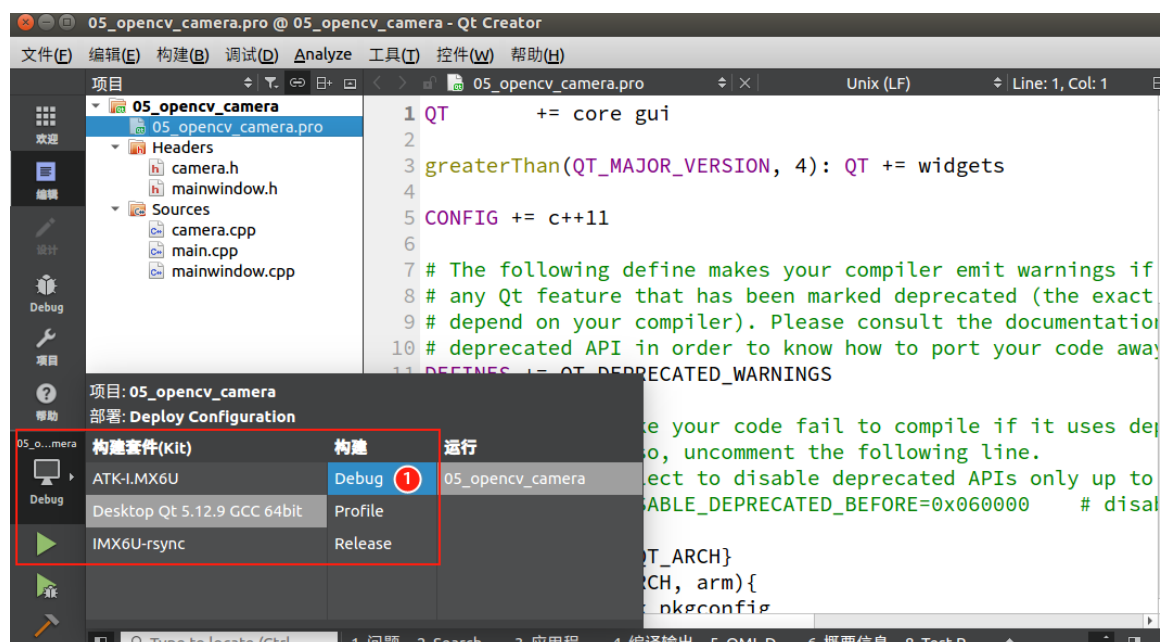


图 9.3-4 选择构建套件

选中项目工程文件夹, 右键点击构建, 构建没有报错后, 点击运行即可在 Ubuntu 上运行。

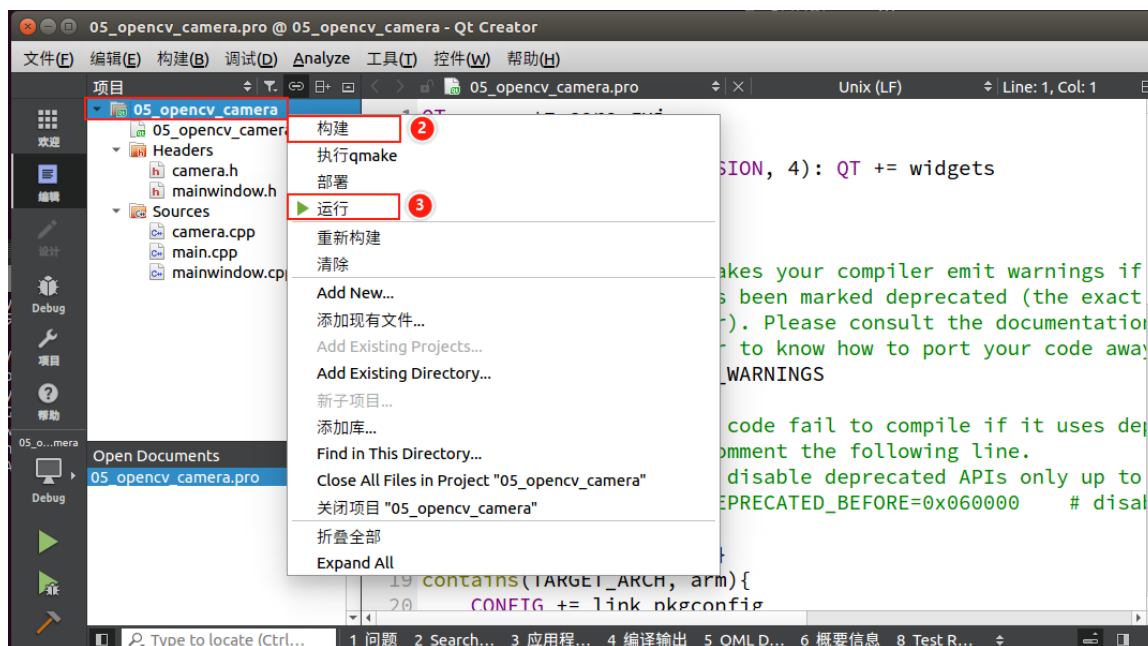


图 9.3-5 构建和运行

在 Ubuntu 界面上会出现对应的窗口, 如果提示无摄像头设备的话, 需要自行检查下虚拟机上是否接入摄像头设备。



图 9.3-6 无摄像头设备

这里笔者使用 USB 摄像头接到电脑上。首先设置 VMware 菜单栏【虚拟机】->【设置】->【USB 控制器】->【USB 兼容性】，设置 USB 兼容性为 USB3.0。

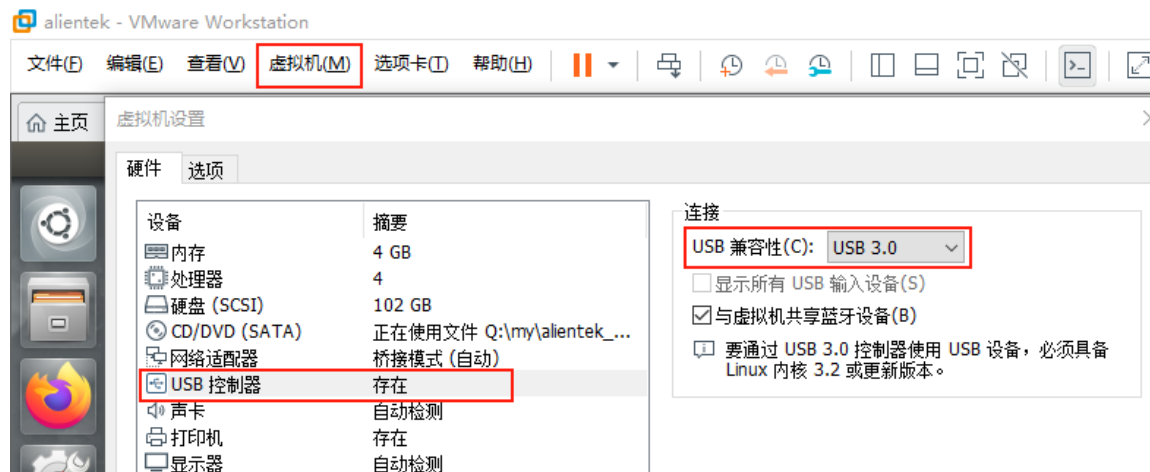


图 9.3-7 设置 USB 兼容性

然后将 USB 摄像头接到电脑 USB 接口, 连接到 Ubuntu 上。这里笔者已经接入 Ubuntu 了, 如果没有接入会显示【连接 (断开与主机的连接)】, 点击连接即可。

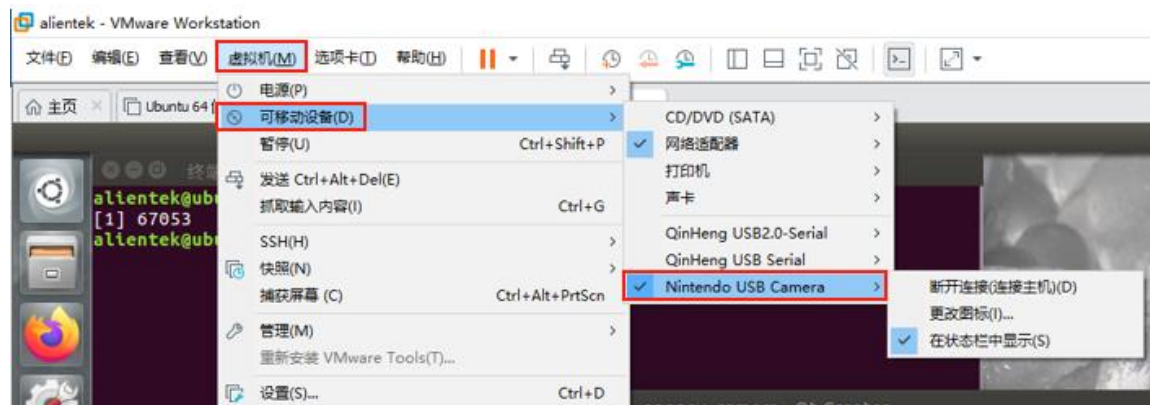


图 9.3-8 摄像头接入 Ubuntu

可以打开 Ubuntu 中的茄子拍照软件, 测试自己的摄像头在 Ubuntu 上能否使用。

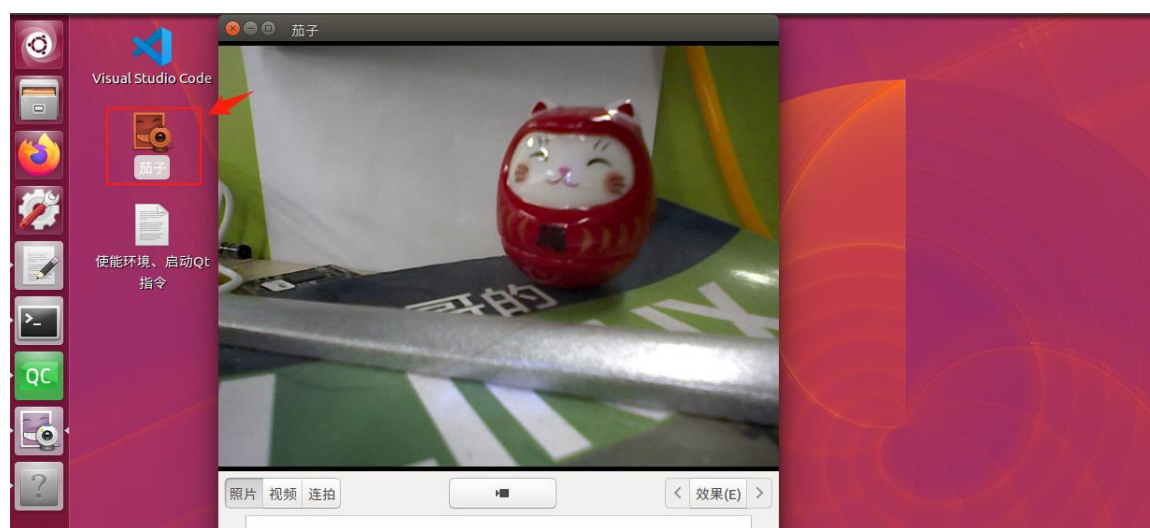


图 9.3-9 测试摄像头

测试完成后, 关闭茄子软件, 重新构建运行 Qt 摄像头例程。此时程序会识别到摄像头设备 video0, 点击开始即可打开摄像头。点击拍照, 在应用程序输出信息中可以看到照片保存位置。

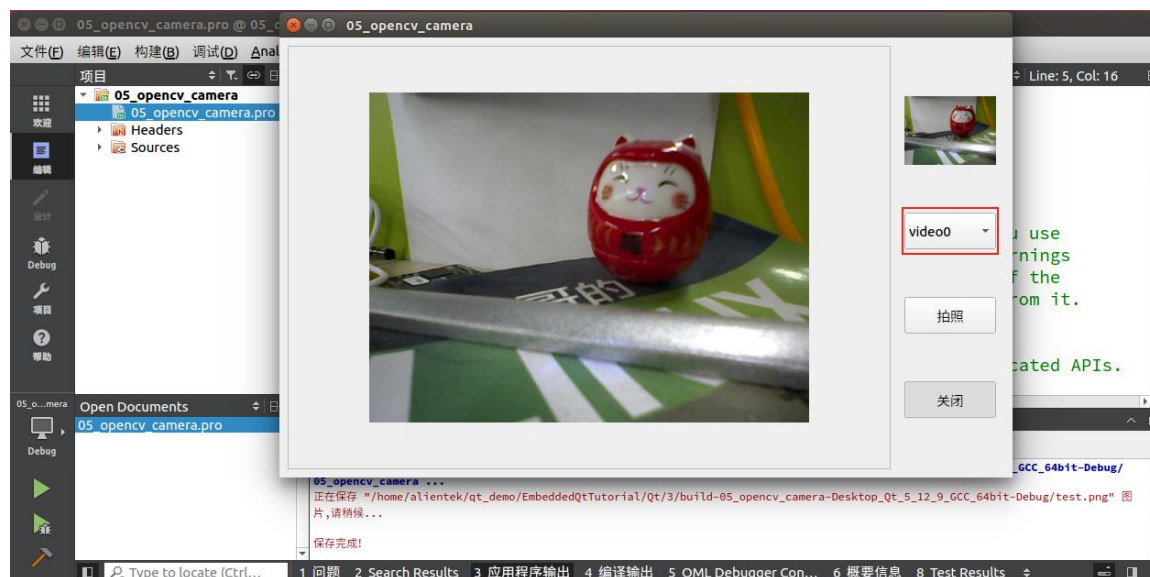


图 9.3-10 测试 Qt 摄像头例程

9.3.2 Qt 程序运行在出厂系统上

首先确保 I.MX6U 开发板上运行的是 Qt5.12.9 的出厂系统, 如果不是的话, 需要下载网盘上的最新的烧录工具, 烧录最新的出厂系统到板子上。

前面我们已经导入好程序和构建套件了, 这里选择 ATK-I.MX6U 构建套件。

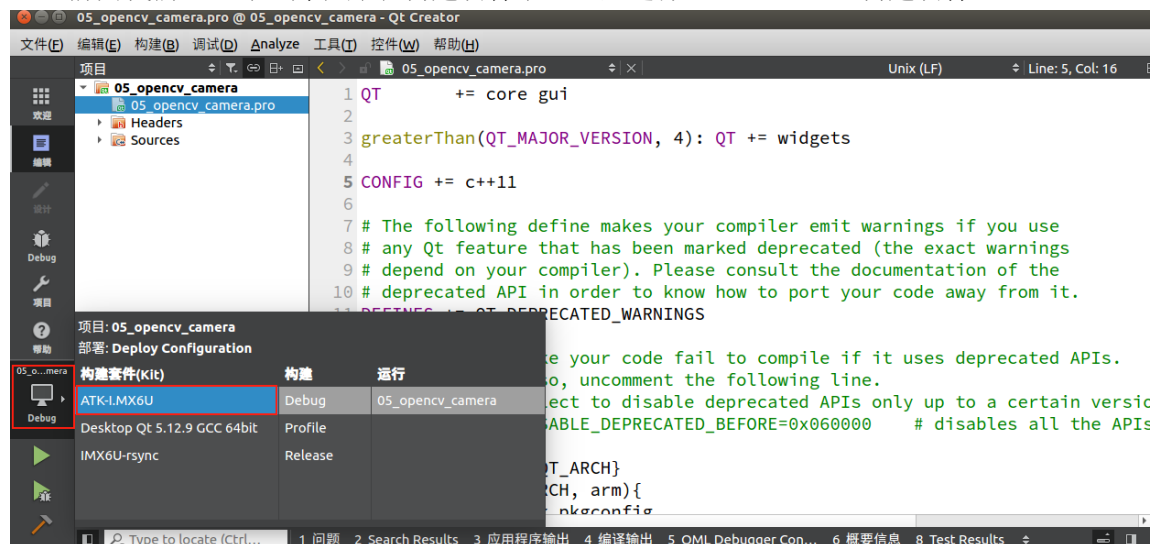


图 9.3-11 选择构建套件

注意这里不能点击运行, 因为 ATK-I.MX6U 构建套件编译出来的是 arm 架构的可执行文件, 调用的也是 arm 的库, 如果在 Ubuntu 上运行的话, 会报错/lib/ld-linux-armhf.so.3: No such file or directory。

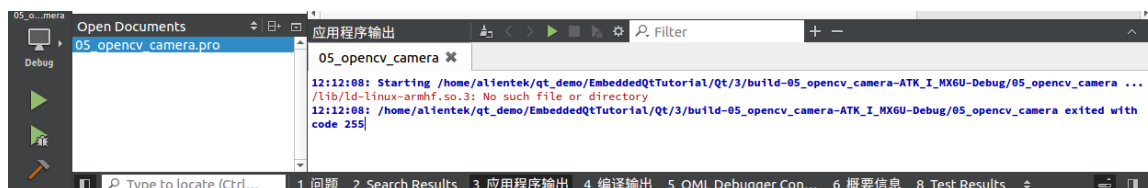


图 9.3-12 报错信息

我们只需要选中工程文件夹，右键点击构建即可。

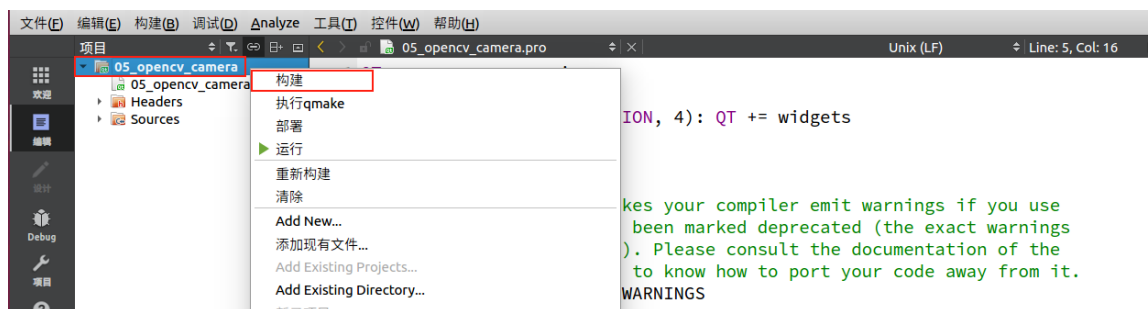


图 9.3-13 构建项目

然后在终端进入到构建生成的 build 工程文件目录下，找到 arm 架构的可执行文件。这里笔者 build 的目录为 qt_demo/EmbeddedQtTutorial/Qt/3/build-05_opencv_camera-ATK_I_MX6U-Debug。

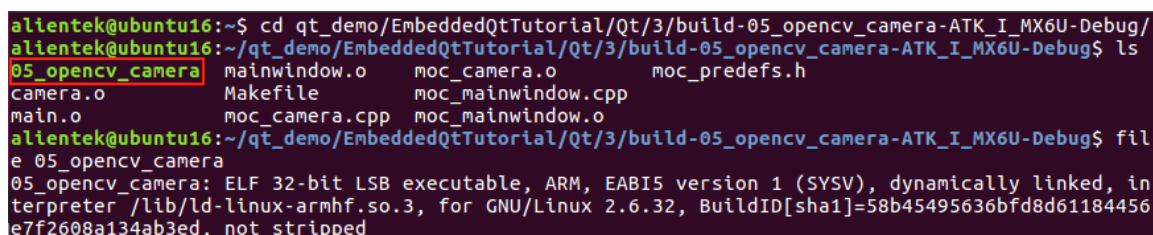


图 9.3-14 arm 架构可执行文件

在没有网络的情况下，我们可以通过 U 盘或者 TF 卡将此文件拷贝到开发板下去运行（注意为 FAT32 格式的设备）。设备挂载到 Ubuntu 后，执行 df 指令可以查看挂载点，将文件拷贝到挂载点即可。

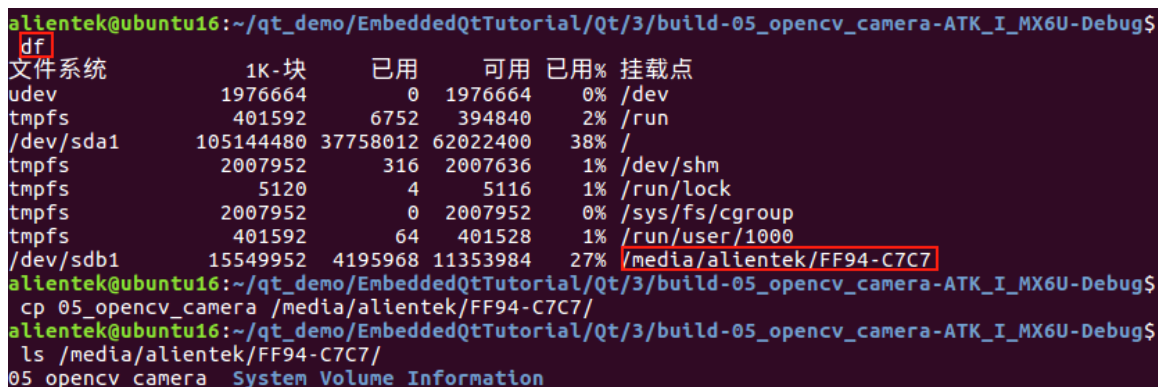


图 9.3-15 拷贝文件到存储设备

USB 摄像头接到开发板上, 启动开发板, 关闭出厂 Qt 界面, 将存储设备接入到开发板上, 出厂系统会默认挂载设备, 我们只需要进入到挂载目录下, 将可执行文件拷贝到开发板 /home/root 目录下去运行即可。

```
root@ATK-IMX6U:~# [ 429.718580] usb 2-1.1: USB disconnect, device number 8
[ 538.003077] usb 2-1.1: new high-speed USB device number 9 using ci_hsrc
[ 538.119517] usb-storage 2-1.1:1.0: USB Mass Storage device detected
[ 538.140606] scsi host4: usb-storage 2-1.1:1.0
[ 539.144656] scsi 4:0:0:0: Direct-Access Generic STORAGE DEVICE 1532 PQ: 0 ANSI: 6
[ 539.463623] sd 4:0:0:0: [sda] 3116288 512-byte logical blocks: (15.9 GB/14.8 GiB)
[ 539.473928] sd 4:0:0:0: [sda] Write Protect is off
[ 539.480231] sd 4:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[ 539.504030] _sda: sda1
[ 539.518016] sd 4:0:0:0: [sda] Attached SCSI removable disk
[ 540.014464] FAT-fs (sda1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.

root@ATK-IMX6U:~# df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/root        7244864 665016   6205168  10% /
devtmpfs         187632    124    187508    1% /dev
tmpfs            40         0        40      0% /mnt/.psplash
tmpfs            253432    204    253228    1% /run
tmpfs            253432    164    253268    1% /var/volatile
/dev/mmcblkp1    129039    6896    122144    6% /run/media/mmcblkp1
/dev/sda1        15549952 1536    15548416  1% /run/media/sda1

root@ATK-IMX6U:~# cd /run/media/sda1
root@ATK-IMX6U:/run/media/sda1# ls
05_opencv_camera 'System Volume Information'
root@ATK-IMX6U:/run/media/sda1# cp 05_opencv_camera /home/root
root@ATK-IMX6U:/run/media/sda1#
root@ATK-IMX6U:/run/media/sda1#
root@ATK-IMX6U:/run/media/sda1# cd /home/root/
root@ATK-IMX6U:~# ls
05_opencv_camera driver shell
root@ATK-IMX6U:~# ./05_opencv_camera
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/var/volatile/tmp/runtime-root'
```

图 9.3-16 拷贝文件到开发板上运行

LCD 屏幕效果参考如下, USB 摄像头我们选择 video2, 如果是 ov5640 摄像头则选择 video1。

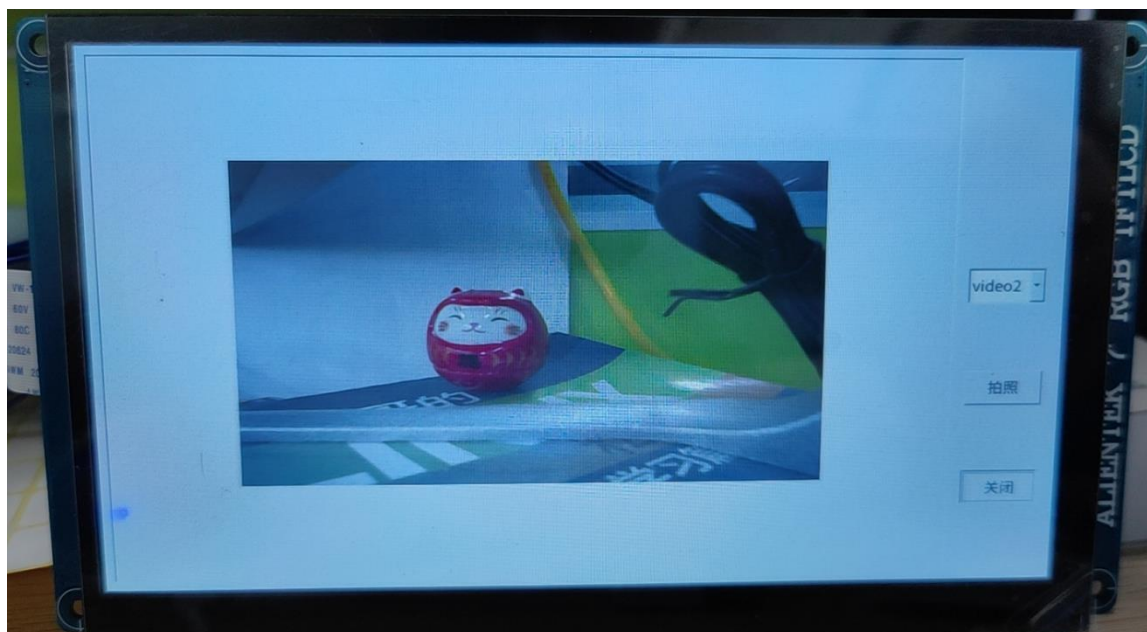


图 9.3-17 效果

9.3.3 使用网络进行 Qt 远程调试

如果开发板出厂系统和 Ubuntu 能够 ping 通, 表示网络通畅, 可以使用 IMX6U-rsync 构建套件来进行 Qt 远程调试。有些程序需要用到开发板本地文件, 就不建议使用远程调试方法。

前提: 关闭电脑和虚拟机防火墙, 开发板和 Ubuntu 在同一网段且能 ping 通。如果不行的话, 可以尝试参考《【正点原子】I.MX6U 网络环境 TFTP&NFS 搭建手册》的方法来搭建、检查下网络设备环境。关闭出厂 Qt 桌面。

这里笔者网络环境已经调试完成, 可以直接进行 Qt 调试。

获取开发板 IP: 进入开发板文件系统后, 执行 ifconfig 可以获取 IP。

```
root@ATK-IMX6U:~# ifconfig
eth0      Link encap:Ethernet  HWaddr b2:16:21:f3:3e:86
          inet addr:192.168.1.115  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::b016:21ff:fef3:3e86/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:15508  errors:0  dropped:156  overruns:0  frame:0
          TX packets:125  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1531645 (1.4 MiB)  TX bytes:19515 (19.0 KiB)
```

图 9.3-18 获取 IP

如果想设置固定的静态 IP, 可以参考《【正点原子】I.MX6U 常见问题汇总 V1.2》9.2 小节的方法。

Qt creator 中点击【工具】->【选项】->【设备】, 选择 IMX6U-rsync (通用 Linux 类型的默认设备), 主机名称中填入开发板的 IP。设置好后点击【Apply】和【OK】就可以应用了。

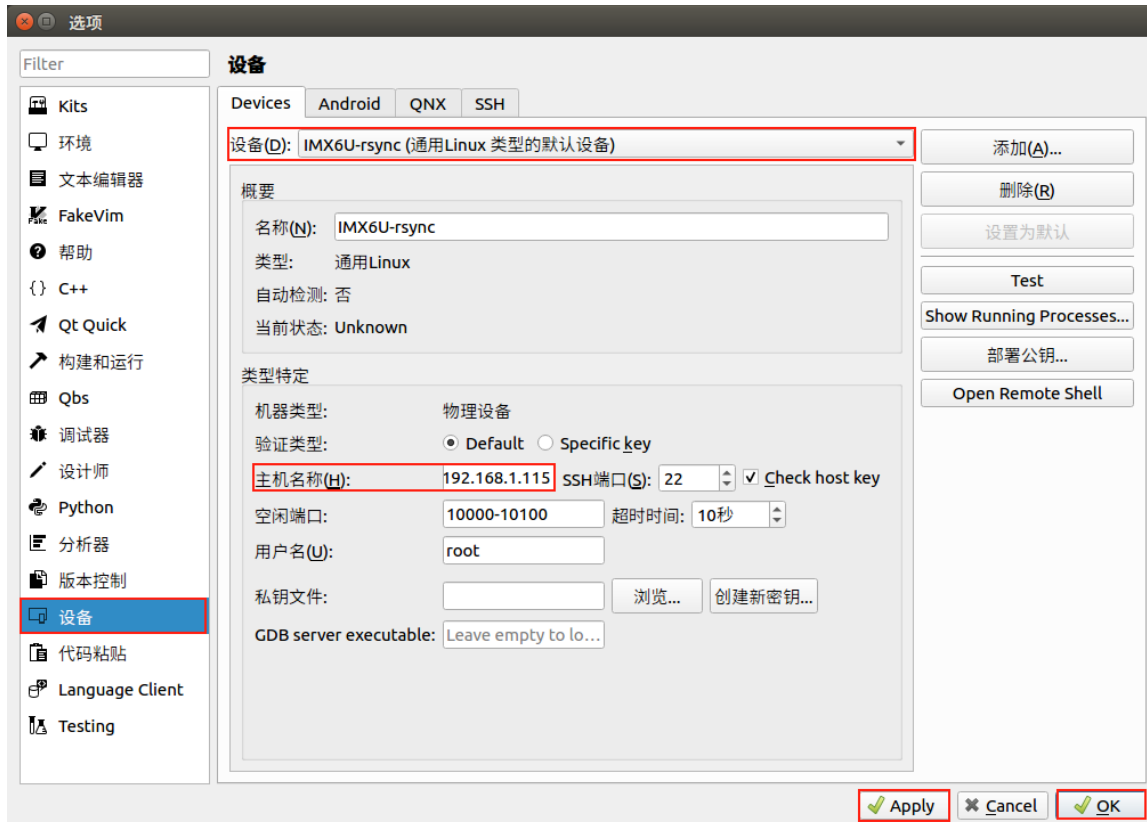


图 9.3-19 填写设备 IP

然后选择 IMX6U-rsync 构建套件, 点击  【运行】即可进行构建和运行在开发板上。

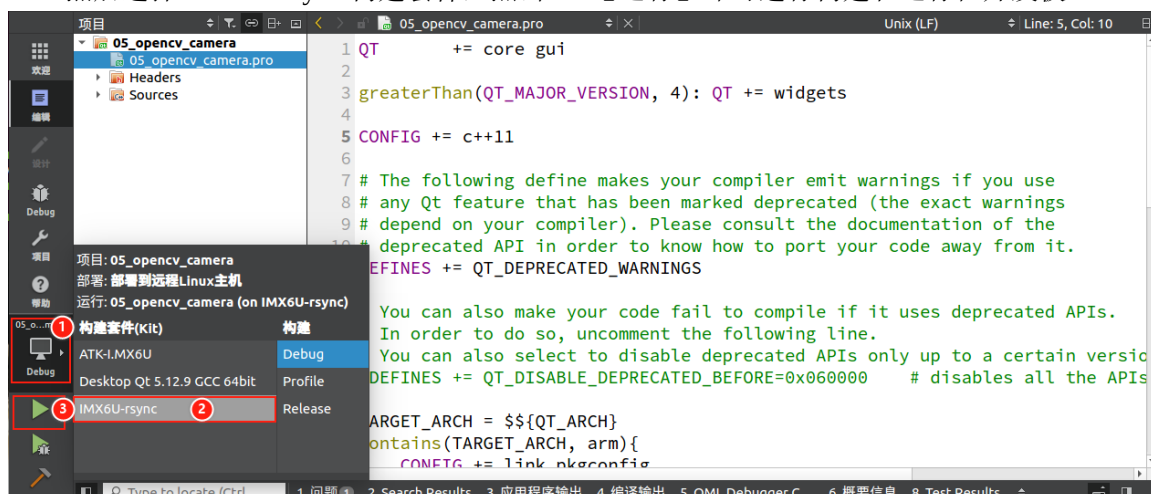


图 9.3-20 选择构建套件并运行

LCD 屏幕效果参考如下, USB 摄像头我们选择 video2, 如果是 ov5640 摄像头则选择 video1。

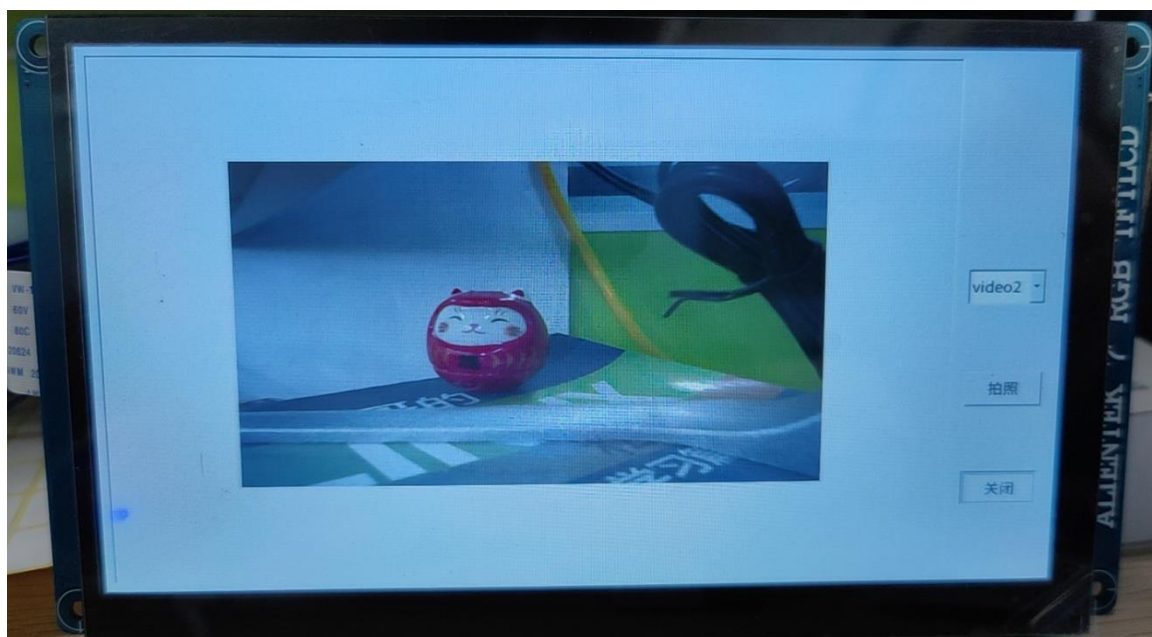


图 9.3-21 效果

点击拍照, 在【应用程序输出】有显示图片在开发板上保存的位置信息。点击这里的 stop 可以停止运行程序, 点击 Re-run 可以重新运行程序。

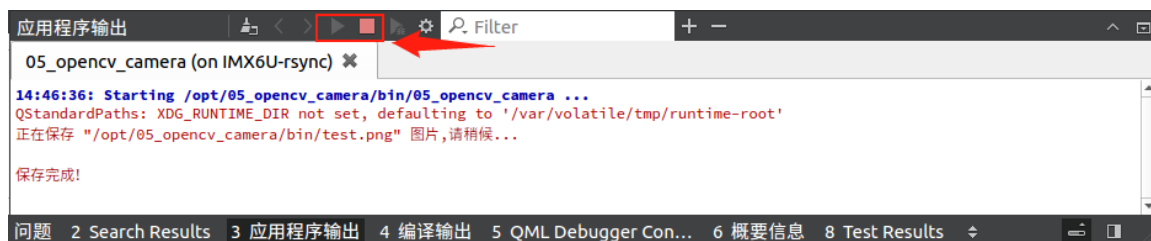


图 9.3-22 保存信息

9.4 重新选择构建套件

如果我们之前选错了构建套件, 怎么才能重新选择构建套件呢? 首先关闭 Qt creator, 在 Qt 程序对应的工程文件夹下找到*.user 文件并删除它, 重新打开 Qt creator 导入工程即可。这里以刚刚的 Qt 摄像头程序为例。

```
allientek@ubuntu16:~/qt_demo/EmbeddedQtTutorial/Qt/3$ cd 05_opencv_camera/  
allientek@ubuntu16:~/qt_demo/EmbeddedQtTutorial/Qt/3/05_opencv_camera$ ls  
05_opencv_camera.pro    camera.cpp  main.cpp    mainwindow.h  
05_opencv_camera.pro.user  camera.h   mainwindow.cpp  
allientek@ubuntu16:~/qt_demo/EmbeddedQtTutorial/Qt/3/05_opencv_camera$ rm 05_opencv_camera.pro.user  
allientek@ubuntu16:~/qt_demo/EmbeddedQtTutorial/Qt/3/05_opencv_camera$ ls  
05_opencv_camera.pro  camera.cpp  camera.h   main.cpp  mainwindow.cpp  mainwindow.h  
allientek@ubuntu16:~/qt_demo/EmbeddedQtTutorial/Qt/3/05_opencv_camera$
```

图 9.4-1 删除.user 文件

删除后, 重新打开 Qt creator 导入工程就可以重新选择构建套件了。