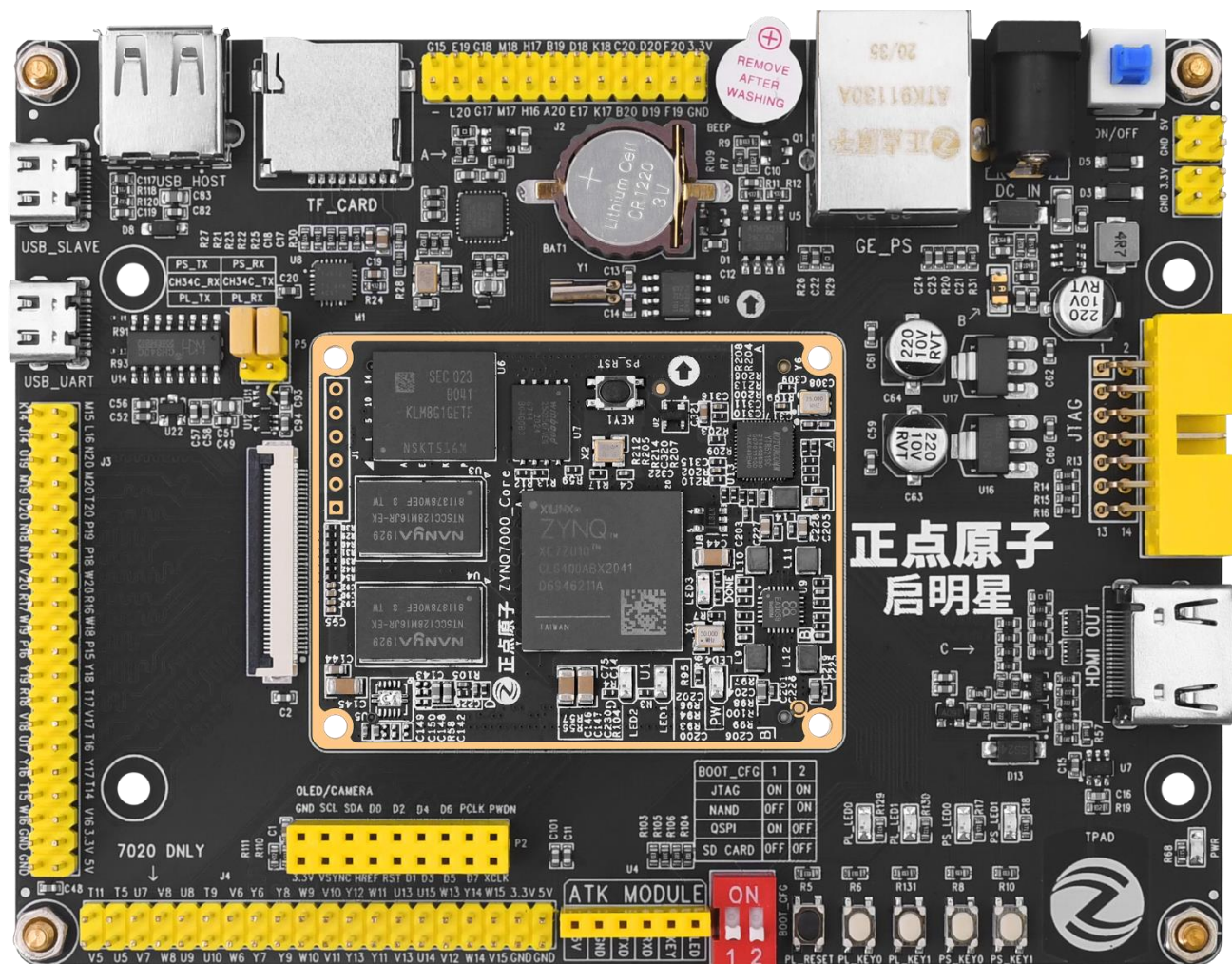


# 启明星 ZYNQ

## 之编译出厂镜像 V1.0





正点原子公司名称: 广州市星翼电子科技有限公司

公司电话: 020 - 38271790

公司网址: [www.alientek.com](http://www.alientek.com)

在线教学平台: [www.yuanzige.com](http://www.yuanzige.com)

开源电子网: [www.openedv.com/forum.php](http://www.openedv.com/forum.php)

天猫旗舰店: <https://zhengdianyuanzi.tmall.com>

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

扫码关注正点原子公众号, 获取更多嵌入式学习资料

扫码下载原子哥 App, 提供数千讲免费开源视频学习

扫码关注 B 站正点原子官方, 所有视频均可免费在线观看

扫码关注抖音正点原子, 技术与娱乐结合体验双倍快乐



扫码关注正点原子公众号



扫码下载“原子哥”APP



扫码关注正点原子B站



扫码关注正点原子抖音账号

## 目录

前言.....	4
第一章 编译出厂镜像.....	5
1.1 安装ZYNQ-7000交叉编译工具链.....	6
1.2 拷贝源码到Ubuntu系统.....	6
1.2.1 拷贝u-boot源码.....	6
1.2.2 拷贝内核源码.....	8
1.2.3 拷贝xsa文件.....	9
1.3 编译.....	10
1.4 设置交叉编译工具链的工作环境.....	10
1.4.1 创建Petalinux工程.....	12
1.4.2 编译出厂源码u-boot、制作BOOT.BIN.....	12
1.4.3 编译内核、设备树.....	14
1.5 启动.....	16
1.6 扩展.....	17
1.6.1 如何使出厂镜像从QSPI启动? .....	17

## 前言

鉴于很多用户需要基于出厂镜像进行开发,其中难免遇到需要修改的地方,比如添加新的驱动和添加新的应用,然后重新生成启动文件。本文讲解如何从源码编译出厂镜像启动文件,如果用户修改了 uboot 和内核源码,可以按照文档重新编译 uboot 和内核,根文件系统的修改就直接将其解压到 sd 卡上,然后添加或删除相应文件即可。

## 第一章 编译出厂镜像

本章介绍如何编译和生成开发板出厂时所烧录的镜像文件，包括 BOOT.BIN 镜像文件、u-boot 镜像文件以及内核镜像文件和设备树。

## 1.1 安装 ZYNQ-7000 交叉编译工具链

编译 uboot 和内核源码需要在 linux 系统（虚拟机）中安装相应的交叉编译工具链。安装 ZYNQ-7000 系列芯片的交叉编译工具链需要 sdk.sh 文件。

把**开发板资料盘 B 盘:\sdk\202002\sdk.sh** 拷贝到 Ubuntu 虚拟机。拷贝完成后，在虚拟机中切换到 sdk.sh 文件所在目录，为 sdk.sh 文件赋予可执行权限并安装，命令如下：

#如果之前没有安装过 build-essential、git、u-boot-tools，请先执行如下命令进行安装

```
sudo apt update
```

```
sudo apt -y install build-essential git u-boot-tools
```

#安装 SDK

```
chmod +x sdk.sh
```

```
./sdk.sh
```

默认是安装在 /opt/petalinux/2020.2 目录下，如果想安装在其他目录下，可以输入相应的路径，此处笔者保持默认，按回车键继续，结果如下图所示：

```
cx@cx-ubtu1:/mnt/hgfs/share$
cx@cx-ubtu1:/mnt/hgfs/share$ l sdk.sh
sdk.sh*
cx@cx-ubtu1:/mnt/hgfs/share$ chmod +x sdk.sh
cx@cx-ubtu1:/mnt/hgfs/share$ ./sdk.sh
Petalinux SDK installer version 2020.2
=====
Enter target directory for SDK (default: /opt/petalinux/2020.2):
You are about to install the SDK to "/opt/petalinux/2020.2". Proceed [Y/n]?
[sudo] password for cx:
Extracting SDK.....
.....done
Setting it up...done
SDK has been successfully set up and is ready to be used.
Each time you wish to use the SDK in a new shell session, you need to source the
environment setup script e.g.
$ . /opt/petalinux/2020.2/environment-setup-aarch64-xilinx-linux
cx@cx-ubtu1:/mnt/hgfs/share$
```

按回车键继续

图 1.1.1 安装 SDK

再次确认是否将 SDK 安装在 /opt/petalinux/2020.2 目录下，默认为“Y”，也就是“是”，按回车键继续往下执行，显示要输入用户密码，输入完成后回车进行安装，等待安装完成。安装完成后提示每次当你在一个新终端中使用 SDK，需要先执行“./opt/petalinux/2020.2/environment-setup-cortexa9t2hf-neon-xilinx-linux-gnueabi”以设置相应的环境变量，其中的“.”和 source 具有相同含义。

## 1.2 拷贝源码到 Ubuntu 系统

### 1.2.1 拷贝 u-boot 源码



开发板出厂镜像使用的 u-boot 源码路径为: 开发板资料盘 (A 盘)\4\_SourceCode\3\_Embedded\_Linux\资源文件\出厂镜像相关, 在该目录下有一个名为 atk-zup-uboot-xlnx.tar.gz 的压缩包文件, 如下所示:

名称	修改日期	类型	大小
 atk-zynq-uboot-xlnx.tar.gz	2023/4/17 14:29	GZ 压缩文件	18,744 KB

图 1.2.1 u-boot 源码

atk-zynq-uboot-xlnx.tar.gz 是专门用于开发板出厂测试的 u-boot 源码压缩包文件。我们将 atk-zynq-uboot-xlnx.tar.gz 压缩包文件拷贝到 Ubuntu 系统中, 如下所示:

```
2023-04-17 14:58:00 py-2.7.17 cx-ubtu in ~
o → 1 /mnt/hgfs/share/source_code/
总用量 19M
-rwxrwxrwx 1 root root 19M 4月 17 14:29 atk-zynq-uboot-xlnx.tar.gz*

2023-04-17 14:58:14 py-2.7.17 cx-ubtu in ~
```

图 1.2.2 将 u-boot 压缩包文件拷贝到 Ubuntu

接下来将其解压, 对应的解压目录就是 U-Boot 源码目录, 这个解压目录大家可以自己设置。因为解压后会自动在解压目录中创建一个名为 atk-zynq-uboot-xlnx 的文件夹, 该文件夹下即是 uboot 源码。为了避免和教程中使用的 uboot 相混淆, 笔者这里选择将其解压到用户家目录下的~/workspace/src 目录中。

执行如下命令将其解压到~/workspace/src/目录中:

```
mkdir -p ~/workspace/src/           #创建~/workspace/src/文件夹
cd /mnt/hgfs/share/source_code/      #切换到 uboot 压缩包文件所在目录
tar -xzf atk-zynq-uboot-xlnx.tar.gz -C ~/workspace/src/  #解压
ls ~/workspace/src/
ls ~/workspace/src/atk-zynq-uboot-xlnx/
```

```

2023-04-17 14:59:28 py-2.7.17 cx-ubuntu in ~
o → mkdir -p ~/workspace/src/

2023-04-17 15:00:01 py-2.7.17 cx-ubuntu in ~
o → cd /mnt/hgfs/share/source_code/

2023-04-17 15:00:13 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → mkdir -p ~/workspace/src/

2023-04-17 15:00:17 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → tar -xzf atk-zynq-uboot-xlnx.tar.gz -C ~/workspace/src/

2023-04-17 15:00:27 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → ls ~/workspace/src/
atk-zynq-uboot-xlnx

```

**1 解压后得到的uboot源码根目录**

```

2023-04-17 15:00:35 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → ls ~/workspace/src/atk-zynq-uboot-xlnx/
api                cmd                doc                fs                Licenses          README
arch               common            drivers            include           MAINTAINERS       scripts
atk-zynq-boot.cmd.default config.mk          dts                Kbuild            Makefile           test
atk-zynq-boot.cmd.default configs            env                Kconfig           net                tools
board              disk              examples            lib                post

```

```

2023-04-17 15:01:09 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code

```

图 1.2.3 解压 U-Boot 源码包

解压完成后, atk-zynq-uboot-xlnx 文件夹存放的即是出厂镜像使用的 uboot 源码。

### 1.2.2 拷贝内核源码

开发板出厂镜像使用的 Linux 内核源码路径为: 开发板资料盘(A盘)\4\_SourceCode\3\_Embedded\_Linux\资源文件\出厂镜像相关, 在该目录下有一个名为 atk-zynq-linux-xlnx.tar.gz 的压缩包文件, 如下所示:

名称	修改日期	类型	大小
 atk-zynq-linux-xlnx.tar.gz	2023/4/17 14:30	GZ 压缩文件	169,579 KB

图 1.2.4 内核源码压缩包文件

atk-zynq-linux-xlnx.tar.gz 是专门用于开发板出厂镜像的 Linux 内核源码压缩包文件。我们将 atk-zynq-linux-xlnx.tar.gz 压缩包文件拷贝到 Ubuntu 系统中, 如下所示:

```

2023-04-17 15:01:09 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → ls /mnt/hgfs/share/source_code
总用量 184M
-rwxrwxrwx 1 root root 166M 4月 17 14:30 atk-zynq-linux-xlnx.tar.gz*
-rwxrwxrwx 1 root root 19M 4月 17 14:29 atk-zynq-uboot-xlnx.tar.gz*

```

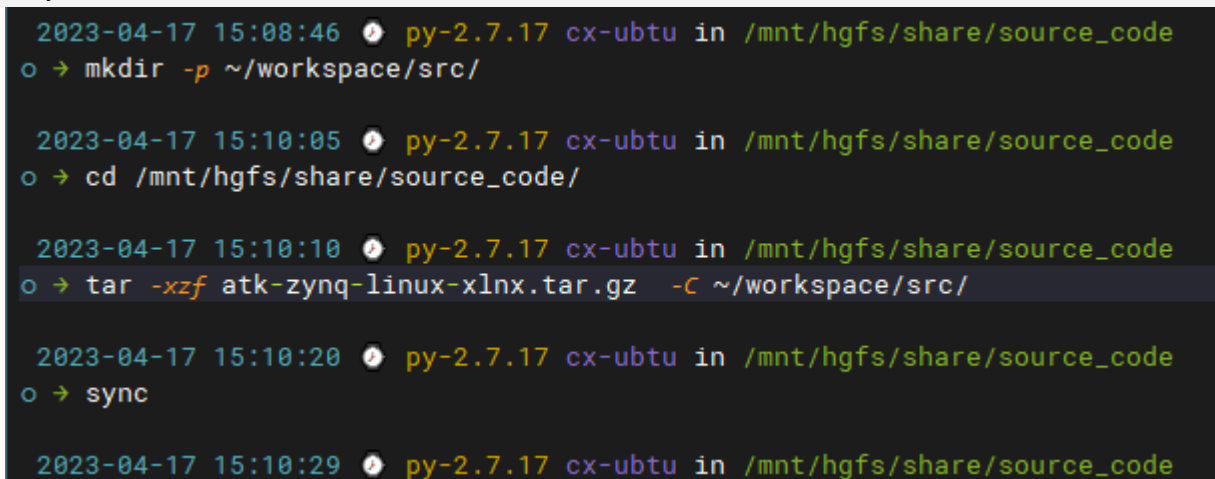
图 1.2.5 将内核源码压缩包文件拷贝到 Ubuntu



接下来将其解压, 对应的解压目录就是 Linux 内核源码目录, 这个解压目录大家可以自己设置。因为解压后会自动在解压目录中创建一个名为 linux-xlnx 的文件夹, 该文件夹下即是 Linux 内核源码。为了避免和教程中使用的 Linux 内核源码相混淆, 笔者这里选择将其解压到用户家目录下的 `~/workspace/src` 目录中:

执行如下命令将其解压到 `~/workspace/src` 目录中:

```
mkdir -p ~/workspace/src/
cd /mnt/hgfs/share/source_code/          #切换到 uboot 压缩包文件所在目录
tar -xzf atk-zynq-linux-xlnx.tar.gz -C ~/workspace/src/  #解压
sync
```



```
2023-04-17 15:08:46 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → mkdir -p ~/workspace/src/

2023-04-17 15:10:05 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → cd /mnt/hgfs/share/source_code/

2023-04-17 15:10:10 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → tar -xzf atk-zynq-linux-xlnx.tar.gz -C ~/workspace/src/

2023-04-17 15:10:20 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → sync

2023-04-17 15:10:29 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
```

图 1.2.6 解压内核源码

解压完成后, 得到的 `atk-zynq-linux-xlnx` 文件夹即是出厂镜像使用的 Linux 内核源码根目录。使用 `ls` 命令 (`ls ~/workspace/src/atk-zynq-linux-xlnx`), 可看到 Linux 内核源码目录结构, 如下图所示:



```
2023-04-17 15:10:29 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → ls ~/workspace/src/
atk-zynq-linux-xlnx  atk-zynq-uboot-xlnx

2023-04-17 15:11:00 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
o → ls ~/workspace/src/atk-zynq-linux-xlnx
arch      Documentation  Kbuild      Makefile     scripts      zynq-fit-image.its
block     drivers        Kconfig     mm            security
certs     fs             kernel      mpsoc-fit-image.its  sound
COPYING   include        lib         net           tools
CREDITS   init           LICENSES    README        usr
crypto    ipc           MAINTAINERS samples       virt

2023-04-17 15:11:36 py-2.7.17 cx-ubuntu in /mnt/hgfs/share/source_code
```

图 1.2.7 Linux 内核源码目录结构

### 1.2.3 拷贝 xsa 文件

使用 xsa 文件以创建相应的 Petalinux 工程,

在开发板资料包中, 已经给大家提供了开发板出厂时所对应的 vivado 工程。

对于启明星 7020 开发板, 使用的是 Phosphor\_7020 工程,

对于启明星 7010 开发板, 使用的是 Phosphor\_7010 工程。

笔者以启明星 7020 开发板为例，路径为：开发板资料盘(A 盘)\4\_SourceCode\3\_Embedded\_Linux\vivado\_prj，在这个目录下有一个压缩包文件 Phosphor\_7020.zip，将其在 Windows 系统下解压，解压后如下图所示：

名称	修改日期	类型	大小
ip_repo	2023/5/17 10:17	文件夹	
Phosphor_7020.cache	2023/5/15 9:25	文件夹	
Phosphor_7020.gen	2023/5/15 9:08	文件夹	
Phosphor_7020.hw	2023/5/15 9:08	文件夹	
Phosphor_7020.ip_user_files	2023/5/15 9:34	文件夹	
Phosphor_7020.runs	2023/5/15 9:35	文件夹	
Phosphor_7020.sim	2023/5/15 9:09	文件夹	
Phosphor_7020.srcs	2023/5/15 9:08	文件夹	
Phosphor_7020.xpr	2023/5/15 9:45	XPR 文件	80 KB
system_wrapper.xsa	2023/5/15 9:48	XSA 文件	942 KB

图 1.2.8 vivado 工程目录

system\_wrapper.xsa 为 Vivado 导出的 xsa 文件，这里直接将 system\_wrapper.xsa 文件夹拷贝到 Ubuntu 系统某个目录下，例如/mnt/hgfs/share/xsa/7020/，如下图所示：

```

2023-04-17 15:25:04 py-2.7.17 cx-ubuntu in ~
o → 1 /mnt/hgfs/share/xsa/7020/
总用量 1.1M
-rwxrwxrwx 1 root root 1.1M 4月 14 14:22 system_wrapper.xsa*
2023-04-17 15:25:13 py-2.7.17 cx-ubuntu in ~

```

图 1.2.9 将 xsa 文件拷贝到 Ubuntu

## 1.3 编译

在上一节中我们已经将编译所需要的所有“原材料”拷贝到 Ubuntu 系统了，接下来可以进行编译了，需要注意的是，在编译之前，需要安装 Xilinx 的 petalinux 工具，以及设置交叉编译工具链的工作环境。如果还没安装 petalinux 的，可以参考【正点原子】启明星 ZYNQ 之嵌入式 Linux 驱动开发指南 V1.x.pdf 的第 5 章内容 Petalinux 的安装进行安装；设置交叉编译工具链的工作环境请参考下一小节。

## 1.4 设置交叉编译工具链的工作环境

每打开一个新终端，都需要在终端中执行如下命令设置 SDK 的环境变量以使用交叉编译器：

```

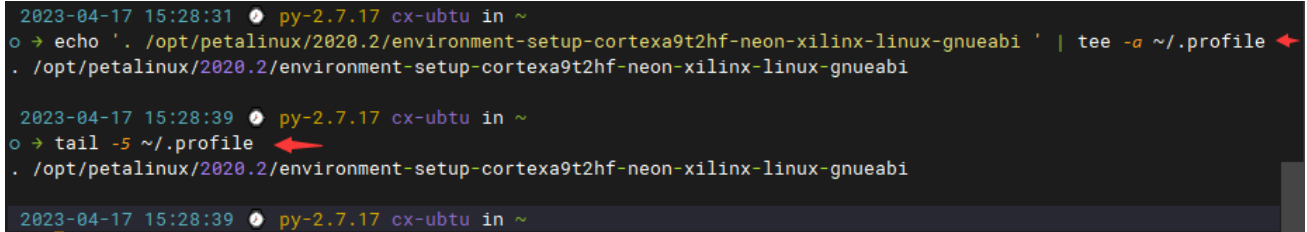
. /opt/petalinux/2020.2/environment-setup-cortexa9t2hf-neon-xilinx-linux-
gnueabi

```

如果不想每打开一个新终端就执行这个命令，可以将其放入 ~/.profile 或者 /etc/profile 文件中，命令如下：

```
echo '. /opt/petalinux/2020.2/environment-setup-cortexa9t2hf-neon-xilinx-linux-gnueabi' | tee -a ~/.profile #注意看下图, 这条命令是完整的一行
tail -5 ~/.profile
```

结果如下图所示:



```
2023-04-17 15:28:31 py-2.7.17 cx-ubtu in ~
o → echo '. /opt/petalinux/2020.2/environment-setup-cortexa9t2hf-neon-xilinx-linux-gnueabi' | tee -a ~/.profile
. /opt/petalinux/2020.2/environment-setup-cortexa9t2hf-neon-xilinx-linux-gnueabi

2023-04-17 15:28:39 py-2.7.17 cx-ubtu in ~
o → tail -5 ~/.profile
. /opt/petalinux/2020.2/environment-setup-cortexa9t2hf-neon-xilinx-linux-gnueabi

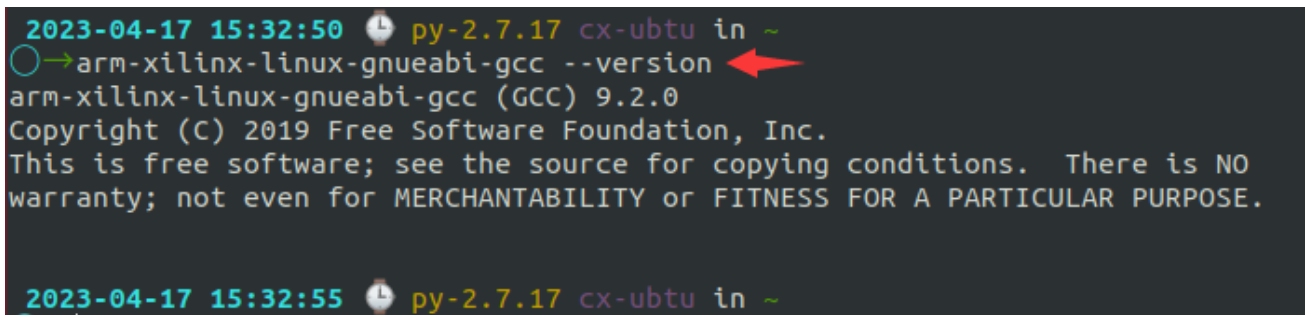
2023-04-17 15:28:39 py-2.7.17 cx-ubtu in ~
```

图 1.4.1 加载 ~/.profile 文件时设置 SDK 环境变量

这样, 将命令 “. /opt/petalinux/2020.2/environment-setup-cortexa9t2hf-neon-xilinx-linux-gnueabi” 写入到 ~/.profile 文件中, 启动虚拟机登录当前用户后会自动执行该命令。不过当前终端不可用, 需要重启才能生效。

注: 如果读者之前用相同的方式设置过 Petalinux 2020.2 版本的 SDK, 请先从 ~/.profile 等文件中删除相关的行, 或者直接使用之前的 SDK, 如果可用的话。

设置 SDK 的环境变量后, 在终端输入命令 `arm-xilinx-linux-gnueabi-gcc --version` 来查看当前使用的交叉编译器版本号。看到如下结果, 表明环境变量已经生效。



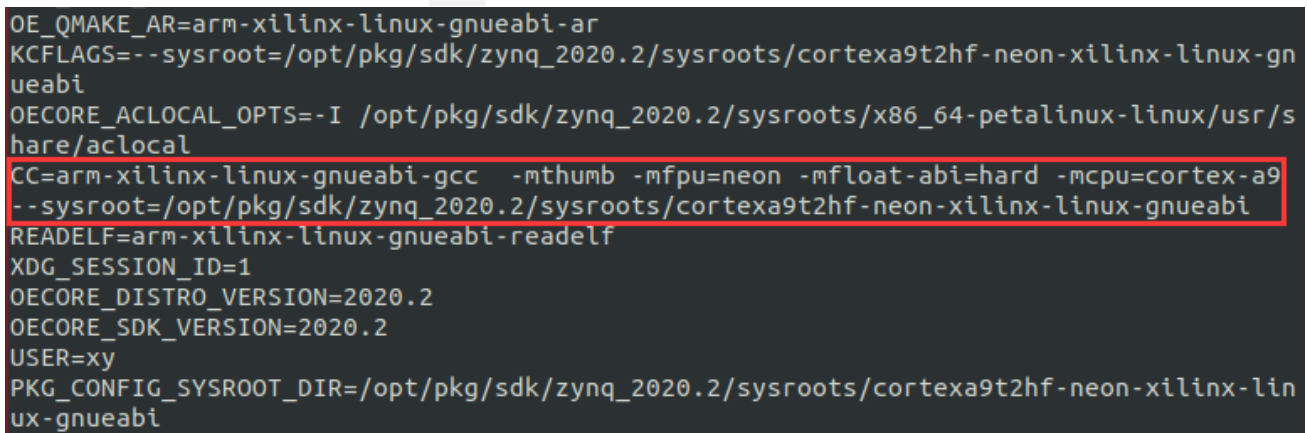
```
2023-04-17 15:32:50 py-2.7.17 cx-ubtu in ~
o → arm-xilinx-linux-gnueabi-gcc --version
arm-xilinx-linux-gnueabi-gcc (GCC) 9.2.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

2023-04-17 15:32:55 py-2.7.17 cx-ubtu in ~
```

图 1.4.2 查看 gcc 版本信息

可以看到, 我们使用的 zynq 交叉编译器版本为 9.2.0。

设置环境变量后可以使用 `env` 命令查看生效的环境变量, 下图为部分截图:



```
OE_QMAKE_AR=arm-xilinx-linux-gnueabi-ar
KCFLAGS=-sysroot=/opt/pkg/sdk/zynq_2020.2/sysroots/cortexa9t2hf-neon-xilinx-linux-gnueabi
OECORE_ACLOCAL_OPTS=-I /opt/pkg/sdk/zynq_2020.2/sysroots/x86_64-petalinux-linux/usr/share/aclocal
CC=arm-xilinx-linux-gnueabi-gcc -mthumb -mfpu=neon -mfloat-abi=hard -mcpu=cortex-a9
--sysroot=/opt/pkg/sdk/zynq_2020.2/sysroots/cortexa9t2hf-neon-xilinx-linux-gnueabi
READELF=arm-xilinx-linux-gnueabi-readelf
XDG_SESSION_ID=1
OECORE_DISTRO_VERSION=2020.2
OECORE_SDK_VERSION=2020.2
USER=xy
PKG_CONFIG_SYSROOT_DIR=/opt/pkg/sdk/zynq_2020.2/sysroots/cortexa9t2hf-neon-xilinx-linux-gnueabi
```

图 1.4.3 查看设置后的环境变量

不同的环境变量有不同的作用。比如环境变量 CC, 可以看出该环境变量已经配置好 gcc 交叉编译器编译时所用的参数, 比如使用 arm-xilinx-linux-gnueabi-gcc 编译器, 可直接使用环境变量 \$CC, 如交叉编译某个 .c 文件, 如 hello.c, 直接使用 \$CC hello.c 就可以了。

### 1.4.1 创建 Petalinux 工程

创建 Petalinux 工程的步骤在【正点原子】启明星 ZYNQ 之嵌入式 Linux 驱动开发指南 V1.x.pdf 第六章 Petalinux 设计流程实战中已讲解, 本章就不细述, 也可直接使用第六章 Petalinux 设计流程实战的 Petalinux 工程, 重新导入 xsa 文件即可。

先在 Ubuntu 主机终端中选一个合适的路径以创建出厂镜像使用的 Petalinux 工程, 然后在终端中输入如下命令:

```
cd <相应的目录> #切换到创建 Petalinux 工程的目录
sptl #设置 petalinux 工作环境
petalinux-create -t project --template zynq -n base #创建 Petalinux 工程
cd base #进入到 petalinux 工程目录下
petalinux-config --get-hw-description /mnt/hgfs/share/xsa/7020 #导入相应的 xsa 文件
```

以上命令执行完成之后会自动在当前目录下创建一个名为 base 的文件夹, 也就是我们出厂镜像的 petalinux 工程对应的工程目录。

xsa 文件导入成功之后会自动弹出 petalinux 工程配置窗口, 如下图所示:

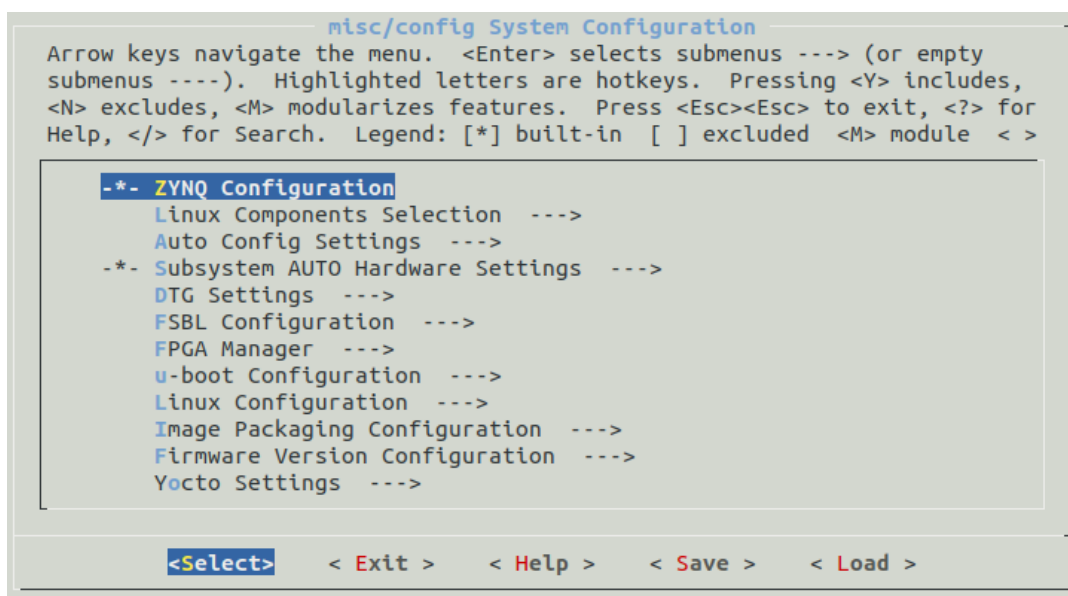


图 1.4.4 petalinux 工程配置窗口

使用默认配置即可, 保存并退出。

### 1.4.2 编译出厂源码 u-boot、制作 BOOT.BIN

进入到 u-boot 源码根目录下。

启明星开发板对应的配置文件为: configs/xilinx\_zynq\_virt\_defconfig

启明星开发板对应的设备树文件为: arch/arm/dts/zynq-atk.dts

执行如下命令编译 u-boot 源码:

```

make distclean           # 清理工程
make xilinx_zynq_virt_defconfig #配置 uboot
make -j                  # 编译

```

```

2023-04-17 15:44:04 py-2.7.17 cx-ubtu in ~
o → cd workspace/src/atk-zynq-uboot-xlnx/

2023-04-17 15:44:06 py-2.7.17 cx-ubtu in ~/workspace/src/atk-zynq-uboot-xlnx
o → make distclean

2023-04-17 15:44:09 py-2.7.17 cx-ubtu in ~/workspace/src/atk-zynq-uboot-xlnx
o → make xilinx_zynq_virt_defconfig
HOSTCC scripts/basic/fixdep
HOSTCC scripts/kconfig/conf.o
YACC scripts/kconfig/zconf.tab.c
LEX scripts/kconfig/zconf.lex.c
HOSTCC scripts/kconfig/zconf.tab.o
HOSTLD scripts/kconfig/conf
#
# configuration written to .config
#

2023-04-17 15:44:14 py-2.7.17 cx-ubtu in ~/workspace/src/atk-zynq-uboot-xlnx
o → make -j
scripts/kconfig/conf --syncconfig Kconfig
CHK include/config.h
UPD include/config.h
CFG u-boot.cfg
GEN include/autoconf.mk.dep
CFG spl/u-boot.cfg
GEN include/autoconf.mk
GEN spl/include/autoconf.mk
CHK include/config/uboot.release

```

图 1.4.5 编译 u-boot 源码

编译成功之后, 会在 u-boot 目录下生成镜像文件, 如下所示:

```

2023-04-17 15:44:30 py-2.7.17 cx-ubtu in ~/workspace/src/atk-zynq-uboot-xlnx
o → ls
api                doc                Licenses          tools             u-boot-elf.0
arch               drivers            MAINTAINERS       u-boot            u-boot.img
atk-zup-boot.cmd.default dts              Makefile          u-boot.bin        u-boot.lds
atk-zynq-boot.cmd.default env               net               u-boot.cfg        u-boot.map
board              examples          post              u-boot.cfg.configs u-boot-nodtb.bin
cmd                fs                README            u-boot.dtb        u-boot.srec
common             include           scripts           u-boot-dtb.bin    u-boot.sym
config.mk          Kbuild           spl               u-boot-dtb.img
configs            Kconfig          System.map        u-boot.elf
disk               lib               test              u-boot-elf.lds

```

2023-04-17 15:45:22 py-2.7.17 cx-ubtu in ~/workspace/src/atk-zynq-uboot-xlnx

图 1.4.6 生成镜像文件

u-boot.elf 是一个 elf 格式的 uboot 可执行文件, u-boot.dtb 是 uboot 的设备树文件。



输入如下命令生成 boot.scr 启动脚本文件:

```
./tools/mkimage -c none -A arm -T script -d atk-zynq-boot.cmd.default boot.scr
```

```
2023-04-17 15:45:22 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-uboot-xlnx
o → ./tools/mkimage -c none -A arm -T script -d atk-zynq-boot.cmd.default boot.scr
Image Name:
Created:      Mon Apr 17 15:46:29 2023
Image Type:   ARM Linux Script (gzip compressed)
Data Size:    2542 Bytes = 2.48 KiB = 0.00 MiB
Load Address: 00000000
Entry Point:  00000000
Contents:
Image 0: 2534 Bytes = 2.47 KiB = 0.00 MiB

2023-04-17 15:46:29 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-uboot-xlnx
```

图 1.4.7 生成 boot.scr

boot.scr 是出厂镜像中 uboot 用于从相应的存储设备加载内核镜像和设备树的脚本文件。

切换到创建的 Petalinux 工程目录, 执行下面的命令制作启明星开发板所需的启动文件 BOOT.BIN:

```
petalinux-build -c bootloader #生成 fsbl.elf 文件
petalinux-package --boot --fsbl --fpga \
--u-boot ~/workspace/src/atk-zynq-uboot-xlnx/u-boot.elf \
--dtb ~/workspace/src/atk-zynq-uboot-xlnx/u-boot.dtb --force
```

```
2023-04-17 17:57:14 py-2.7.17 cx-ubuntu in ~/workspace/petalinux/base
o → petalinux-package --boot --fsbl --fpga --u-boot ~/workspace/src/atk-zynq-uboot-xlnx/u-boot.elf --dtb ~/workspace/src/atk-zynq-uboot-xlnx/u-boot.dtb --force
INFO: Sourcing build tools
INFO: File in BOOT BIN: "/home/xy/workspace/petalinux/base/images/linux/zynq_fsbl.elf"
INFO: File in BOOT BIN: "/home/xy/workspace/petalinux/base/project-spec/hw-description/system_wrapper.bit"
INFO: File in BOOT BIN: "/home/xy/workspace/src/atk-zynq-uboot-xlnx/u-boot.elf"
INFO: File in BOOT BIN: "/home/xy/workspace/src/atk-zynq-uboot-xlnx/u-boot.dtb"
INFO: Generating Zynq binary package BOOT.BIN...

***** Xilinx Bootgen v2020.2
***** Build date : Nov 15 2020-06:11:24
***** Copyright 1986-2020 Xilinx, Inc. All Rights Reserved.

[INFO] : Bootimage generated successfully
INFO: Binary is ready.

2023-04-17 17:59:17 py-2.7.17 cx-ubuntu in ~/workspace/petalinux/base
```

图 1.4.8 制作 BOOT.BIN 文件

注: 如果编译报设备树相关的错误, 请按照第六章 Petalinux 设计流程实战那样配置设备树, 我们不用 Petalinux 编译的设备树, 只是解决错误。

命令执行成功之后, 会在当前 Petalinux 工程的 images/linux 目录下生成 BOOT.BIN 启动文件, 如下图所示:

```
2023-04-17 18:02:46 py-2.7.17 cx-ubuntu in ~/workspace/petalinux/base
o → 1 images/linux/
总用量 5.2M
-rw-rw-r-- 1 xy xy 4.7M 4月 17 17:59 BOOT.BIN
-rw-r--r-- 1 xy xy 539K 4月 17 17:52 zynq_fsbl.elf
```

图 1.4.9 生成 BOOT.BIN 文件

### 1.4.3 编译内核、设备树



进入到 Linux 内核源码根目录下。

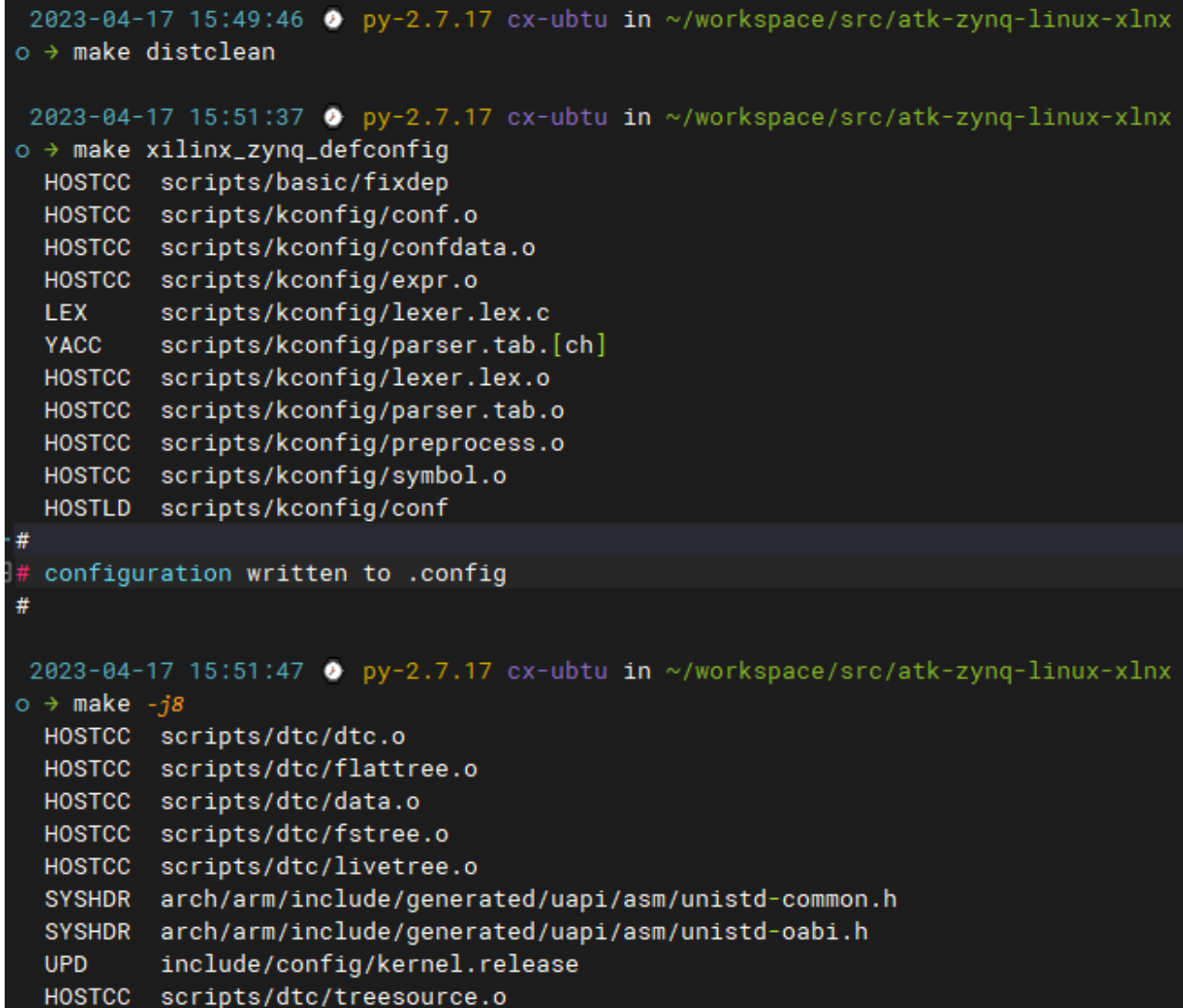
内核 defconfig 配置文件为: arch/arm/configs/xilinx\_zynq\_defconfig

启明星 7020 开发板对应的设备树文件为: arch/arm/boot/dts/atk-phosphor-7020.dts

启明星 7010 开发板对应的设备树文件为: arch/arm/boot/dts/atk-phosphor-7010.dts

执行如下命令编译内核源码:

```
make distclean          # 清理工程
make xilinx_zynq_defconfig # 配置
make -j8                # 编译 zImage 和设备树
```



```
2023-04-17 15:49:46 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-linux-xlnx
o → make distclean

2023-04-17 15:51:37 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-linux-xlnx
o → make xilinx_zynq_defconfig
HOSTCC scripts/basic/fixdep
HOSTCC scripts/kconfig/conf.o
HOSTCC scripts/kconfig/confdata.o
HOSTCC scripts/kconfig/expr.o
LEX scripts/kconfig/lexer.lex.c
YACC scripts/kconfig/parser.tab.[ch]
HOSTCC scripts/kconfig/lexer.lex.o
HOSTCC scripts/kconfig/parser.tab.o
HOSTCC scripts/kconfig/preprocess.o
HOSTCC scripts/kconfig/symbol.o
HOSTLD scripts/kconfig/conf
#
# configuration written to .config
#

2023-04-17 15:51:47 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-linux-xlnx
o → make -j8
HOSTCC scripts/dtc/dtc.o
HOSTCC scripts/dtc/flattree.o
HOSTCC scripts/dtc/data.o
HOSTCC scripts/dtc/fstree.o
HOSTCC scripts/dtc/livetree.o
SYSHDR arch/arm/include/generated/uapi/asm/unistd-common.h
SYSHDR arch/arm/include/generated/uapi/asm/unistd-oabi.h
UPD include/config/kernel.release
HOSTCC scripts/dtc/treesource.o
```

图 1.4.10 编译内核源码

编译得到的内核镜像文件 zImage 在 arch/arm/boot 目录下, 如下图所示:

```

2023-04-17 15:53:32 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-linux-xlnx
o → l arch/arm/boot/
总用量 14M
drwxrwxr-x 2 xy xy 4.0K 4月 14 17:56 bootp/
drwxrwxr-x 2 xy xy 4.0K 4月 17 15:53 compressed/
-rwxrwxr-x 1 xy xy 1.7K 4月 14 17:56 deflate_xip_data.sh*
drwxrwxr-x 2 xy xy 96K 4月 17 15:51 dts/
-rwxrwxr-x 1 xy xy 13M 4月 17 15:53 Image*
-rw-rw-r-- 1 xy xy 1.7K 4月 14 17:56 install.sh
-rw-rw-r-- 1 xy xy 3.1K 4月 14 17:56 Makefile
-rwxrwxr-x 1 xy xy 4.7M 4月 17 15:53 zImage*

```

图 1.4.11 生成的内核镜像文件

编译得到的设备树文件在 arch/arm/boot/dts/ 目录下, 如下图所示:

```

2023-05-17 10:29:14 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-linux-xlnx
o → l arch/arm/boot/dts/atk* ←
-rw-rw-r-- 1 xy xy 22K 5月 17 10:29 arch/arm/boot/dts/atk-navigator-7010.dtb
-rw-rw-r-- 1 xy xy 653 5月 17 10:29 arch/arm/boot/dts/atk-navigator-7010.dts
-rw-rw-r-- 1 xy xy 24K 5月 17 10:29 arch/arm/boot/dts/atk-navigator-7020.dtb
-rw-rw-r-- 1 xy xy 639 5月 17 10:29 arch/arm/boot/dts/atk-navigator-7020.dts
-rw-rw-r-- 1 xy xy 22K 5月 17 10:29 arch/arm/boot/dts/atk-phosphor-7010.dtb
-rw-rw-r-- 1 xy xy 595 5月 17 10:29 arch/arm/boot/dts/atk-phosphor-7010.dts
-rw-rw-r-- 1 xy xy 22K 5月 17 10:29 arch/arm/boot/dts/atk-phosphor-7020.dtb
-rw-rw-r-- 1 xy xy 595 5月 17 10:29 arch/arm/boot/dts/atk-phosphor-7020.dts

2023-05-17 10:29:17 py-2.7.17 cx-ubuntu in ~/workspace/src/atk-zynq-linux-xlnx

```

图 1.4.12 设成的设备树文件

atk-phosphor-7020.dtb 对应启明星 7020 开发板的设备树文件;

atk-phosphor-7010.dtb 对应启明星 7010 开发板的设备树文件。

至此, 启动开发板所需所有镜像文件都已经编译完成了, 包括 BOOT.BIN 文件、boot.scr、zImage、设备树 dtb 四个文件。

## 1.5 启动

制作一张 SD 启动卡, SD 启动卡的制作方法, 可以参考【正点原子】启明星 ZYNQ 之嵌入式 Linux 驱动开发指南 V1.x.pdf 第六章 Petalinux 设计流程实战中的制作 SD 启动卡小节。这里不再赘述!

将上一节编译得到的四个镜像文件拷贝到 SD 启动卡的第一个分区, 也就是 Fat32 分区, 其中

BOOT.BIN 文件位于 Petalinux 工程的 images/linux 目录下;

boot.scr 位于 uboot 源码根目录下;

zImage 文件位于 linux 内核源码根目录下的 arch/arm/boot/ 目录下;

dtb 文件位于 linux 内核源码根目录下的 arch/arm/boot/dts/ 目录下。

拷贝完成后如下所示:

```

2023-04-17 16:07:53 py-2.7.17 cx-ubuntu in ~
o → cd /media/xy/BOOT/

2023-04-17 16:07:56 py-2.7.17 cx-ubuntu in /media/xy/BOOT
o → ls
atk-navigator-7010.dtb atk-navigator-7020.dtb BOOT.BIN boot.scr zImage

2023-04-17 16:07:57 py-2.7.17 cx-ubuntu in /media/xy/BOOT

```

图 1.5.1 SD 启动卡 Fat 分区文件

注: 一般只需要拷贝开发板对应的设备树 dtb 文件就可以了, 拷贝多个设备树文件也是可以的, 启动过程中只用到开发板对应的设备树 dtb 文件。

接下来我们要拷贝根文件系统到 SD 启动卡第二个分区, 在开发板资料包中已经给大家提供了出厂时使用的根文件系统, 在**开发板资料盘(A 盘)\4\_SourceCode\3\_Embedded\_Linux\资源文件\出厂镜像相关**目录下有一个名为 rootfs.tar.gz 的压缩包文件, 如下所示:




名称	修改日期	类型	大小
 rootfs.tar.gz	2022/6/27 17:24	GZ 压缩文件	232,352 KB
 atk-zup-uboot-xlnx.tar.gz	2022/6/22 15:07	GZ 压缩文件	193,959 KB
 atk-zup-linux-xlnx.tar.gz	2022/6/22 16:33	GZ 压缩文件	2,345,508...

图 1.5.2 根文件系统

rootfs.tar.gz 中包含的根文件系统是使用 Petalinux 编译出来的, 也是出厂镜像使用的根文件系统, 由于测试需要, 笔者也移植了一些软件到其中。该根文件系统比较大、内容比较多, 包括 Qt5、Python、opencv 等常用库文件。

将 rootfs.tar.gz 拷贝到 Ubuntu 系统中, 然后使用 tar 命令将 rootfs.tar.gz 压缩包文件解压到 SD 启动卡的第二个分区, 如下所示:

```

sudo tar -xzf rootfs.tar.gz -C /media/$USER/rootfs
ls /media/$USER/rootfs
sync

```

```

○ cx-ubuntu in /mnt/hgfs/share/source_code
→ sudo tar -xzf rootfs.tar.gz -C /media/$USER/rootfs
○ cx-ubuntu in /mnt/hgfs/share/source_code
→ ls /media/$USER/rootfs
bin dev home media modules proc sbin tmp var
boot etc lib mnt opt run sys usr
○ cx-ubuntu in /mnt/hgfs/share/source_code
→ sync
○ cx-ubuntu in /mnt/hgfs/share/source_code

```

图 1.5.3 解压根文件系统到 SD 卡第二分区

sync 同步完成之后, 卸载 SD 卡 (umount 命令), 拔掉 SD 卡将其插入开发板 SD 卡卡槽, 将开发板启动方式设置为 SD 卡启动, 连接串口、LCD 屏和电源启动开发板。

## 1.6 扩展

### 1.6.1 如何使出厂镜像从 QSPI 启动?

出厂镜像将启动文件放在 QSPI 中, 根文件系统存到 eMMC 中。从 QSPI 启动, 首先需要像出厂镜像那样将 zImage 文件和 dtb 文件打包成 image.ub 文件, 这也是 Petalinux 默认的文件形式, 打包方法如下:

首先进入到 Linux 内核源码根目录下。

对于 7020 核心板, 输入如下命令即可:

```
sed -i 's/navigator/phosphor/g' zynq-fit-image.its
mkimage -f zynq-fit-image.its image.ub
```

对于 7010 核心板, 输入如下命令即可:

```
sed -i 's/navigator/phosphor/g' zynq-fit-image.its
sed -i 's/7020/7010/g' zynq-fit-image.its
mkimage -f zynq-fit-image.its image.ub
```

生成的 image.ub 文件在内核源码根目录下, 输入命令 `ls -l image.ub` 即可看到。

打包完成后, 将 image.ub 文件和之前编译得到的 BOOT.BIN、boot.scr 文件拷贝到 SD 启动卡的第一个分区, 也就是 Fat32 分区, 其中

BOOT.BIN 文件位于 Petalinux 工程的 images/linux 目录下;

boot.scr 位于 uboot 源码根目录下;

image.ub 文件位于 Linux 源码根目录下。

根文件依旧如前解压到 SD 启动卡第二个分区。

从 SD 卡启动后, 在串口终端中输入如下命令将启动镜像固化到 QSPI 和拷贝根文件系统到 eMMC:

```
/opt/image/burn_qspi.sh
```

```
root@ALIENTEK-ZYNQ:~#
root@ALIENTEK-ZYNQ:~# /opt/image/burn_qspi.sh
#####
##### ALIENTEK ZYNQ Flash QSPI #####
##### Version: 1.0 #####
##### www.openedv.com #####
##### Author: DengTao #####
#####
Erase partition /dev/mtd0
█
```

图 1.6.1 执行固化脚本

系统固化完成之后, 蜂鸣器会响, 表明已经将系统镜像和根文件系统分别烧写到了 QSPI 和 eMMC 存储介质中了, 也意味着开发板可以从 QSPI 方式启动系统了。