小眼睛科技

提供一站式 FPGA&嵌入式解决方案

Kosmo 之 Linux 使用指导手册

Kosmo 系列（PG2K100-6IMBG400）

开发板实验教程

紫光同创 Kosmo 系列开发平台

文档版本修订记录

版本号	发布日期	修订记录
V1.0	2025/10/20	初始版本

公司名称：深圳市小眼睛科技有限公司

地址：深圳市宝安区西乡街道 F518 时尚创意园

官方网址：www.meyesemi.com

官方淘宝店铺：小眼睛半导体

B 站：小眼睛半导体（视频教程免费学）

* 加入 FPGA 开发者技术交流与 5000+FPGA 开发者实时沟通

QQ2 群： 442106123QQ3 群：882634519)

*配套资料下载、技术答疑请登录逻辑矩阵技术论坛

LOGIC MATRIX 逻辑矩阵

逻辑矩阵技术论坛欢迎各位发烧友加入
让我们共建开源生态，持续赋能行业发展
https://www.szlogicmatrix.com/
技术论坛

*扫码注册开源技术论

*扫一扫关注官微

* 官方旗舰

目录

1.Linux 环境搭建

1.1.实验简介

实验目的：

搭建 Linux 系统环境。

实验环境：

Window11

VMware

硬件环境：

PG2K100-6IMBG400

1.2.Linux 前置环境安装

1.2.1.VMware 简介

VMware 是一款功能强大的虚拟化软件，简称为称为虚拟机。虚拟机顾名思义就是虚拟出一个机器，然后你就可以在这个机器上安装任何你想要的系统，相当于在克隆出一个你的电脑，这样在主机上运行 Windows 系统，当我们需要用到 Ubuntu 的话就打开安装有 Ubuntu 系统的虚拟机。而 VMware 是一款功能强大的虚拟化软件，它的核心作用是在你现有的物理电脑上，利用软件技术模拟出一套完整的硬件环境，从而让你能够像运行普通应用程序一样，在窗口中运行另一个甚至多个独立的操作系统。

对于开发者而言，VMware 就像是一个安全且高效的“系统实验室”。它最显著的优势在于环境隔离，你可以在虚拟机里进行各种高风险的实验，比如修改系统内核、测试不稳定的驱动程序或运行可疑软件，由于虚拟机与物理主机之间存在天然的隔离屏障，即便虚拟机内的系统彻底崩溃，也不会对你原本的文档和系统造成任何损伤。这种特性为 Linux 驱动开发或复杂的 C++ 编译环境搭建提供了极高的容错率。

此外，VMware 提供的“快照”功能极大地提升了工作效率。在进行大规模的环境配置（如安装复杂的 EDA 工具或配置 Buildroot 交叉编译链）之前，你可以随时保存当前系统的状态。一旦后续操作失误导致环境混乱，只需一键即可让系统秒级回溯到正常状态，完全省去了重装系统的痛苦过程。

在硬件协作方面，VMware 具备优秀的外设透传能力。它可以接管主机的 USB 接口，将连接在电脑上的 JTAG 下载器、串口线或 FPGA 开发板直接分配给虚拟机使用。这意味着你可以在 Windows 系统下利用 Blender 进行 3D 建模或处理 CVD 实验数据的同时，在虚拟机窗口里无缝进行嵌入式系统的调试与烧录。这种“一机多用、互不干扰”的工作模式，使其成为了现代计算机开发与科研工作中不可或缺的辅助工具。

1.2.1.1.支持平台

软件工具	Windows 操作系统
VMware	Windows 7,10,11、

1.2.1.2.软件安装

VMware Workstation Pro 是行业标准桌面 Hypervisor，使用它可在 Windows 或 Linux 桌面上运行 Windows、Linux 和 BSD 虚拟机。

VMware Workstation Pro 之前曾为付费软件，自 2024 年 11 月，VMware by Broadcom 宣布 VMware Fusion 和 Workstation 现在对所有用户免费。VMware Fusion 和 VMware Workstation 都将从付费订阅模式过渡，现在可以免费使用这些工具。这些产品的付费版本（Workstation Pro 和 Fusion Pro）不再可供购买。而且当前免费版将包含付费版中依赖的所有功能。

资料包中有提供 VMware-workstation-17.0.2 版本安装包，若用户需要安装其他版本可自行安装。

1.2.1.3.注意事项

软件安装过程中有如下注意事项：

1）软件建议安装我们推荐的版本，否则版本差异会导致一些奇怪的问题；

2）开始安装软件前，关闭电脑的杀毒软件以及防火墙；

3）开始安装软件前，检查个人电脑账户名称是否是中文或特殊字符，如果含有这两个其中任意的一个，可能会导致软件安装失败或者安装完成也不能正常工作；

4）安装软件的路径可以自定义，但是自定义的路径不能出现中文还有特殊字符，否则会导致软件安装失败（建议使用默认安装否则会有部分步骤操作与文档不同）；

5）开发工具安装时比较耗时，需要耐心等待，中间不能出现强行终止或者意外中断，否则只能重新进行安装。

1.2.1.4.安装程序

首先在资料包中找到 VMware 安装包，双击安装包中的安装程序 VMware.exe，启动安装程序

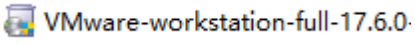


图 1.2-1

启动安装程序后的界面如下：

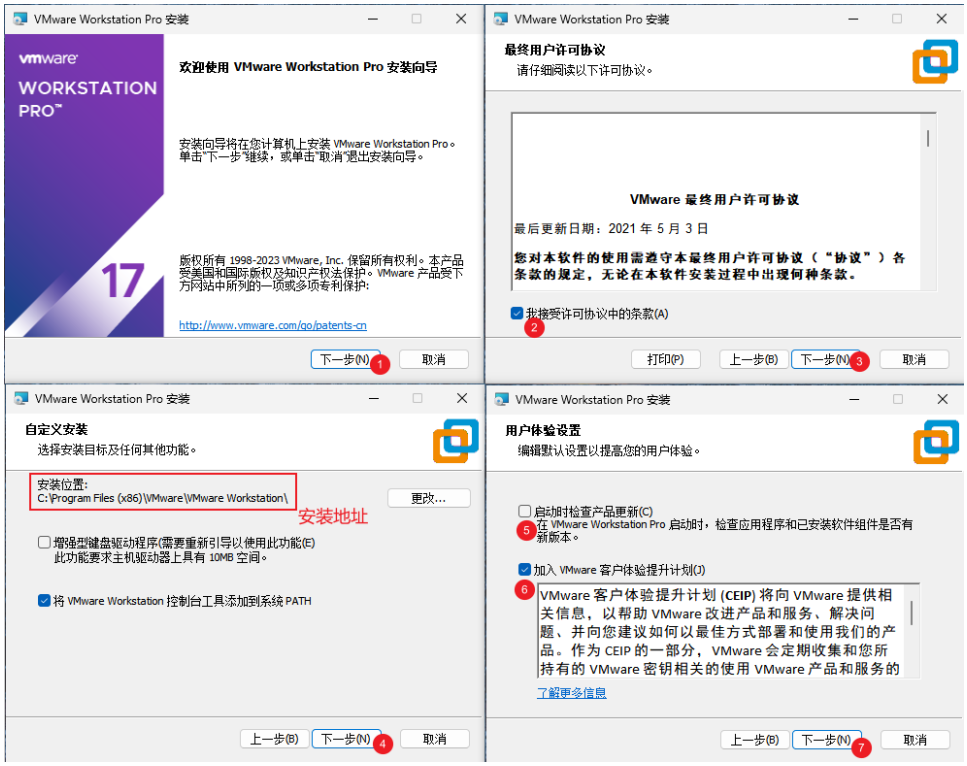


图 1.2-2

如图所示，安装路径建议采用默认路径，若使用自定义安装路径需注意路径不要出现中文和特殊字符。可以选择勾选掉步骤 5、6，点击“下一步”后则跳转到安装界面。

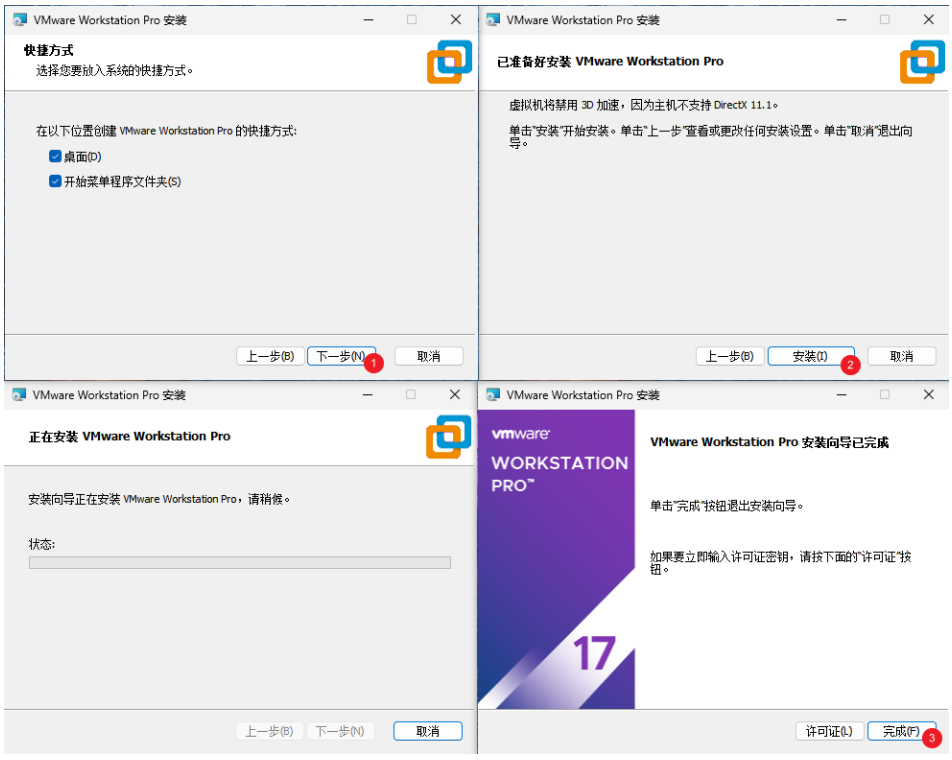


图 1.2-6

继续点击“下一步”直到完成全部安装过程，点击“完成”，安装完毕。

结束安装。在桌面上看到如下图标，双击后选个人使用即可：

图 1.2-3

1.3.Linux 开发环境搭建

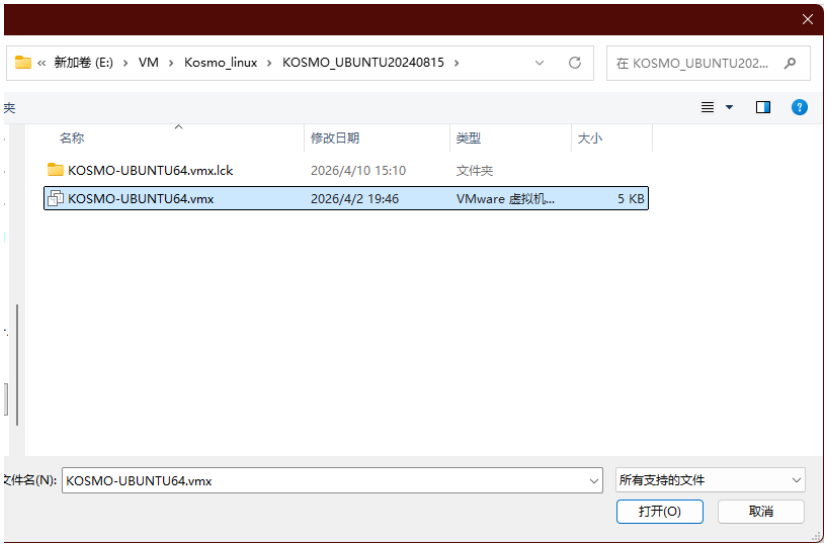
1.3.1.Linux 镜像搭建

资料包里面提供了 ubuntu 压缩包，默认配置 8G 内存和 2 个处理器，需要电脑主机配置至少 16G 内存和 4 核处理器。启动 ubuntu 的硬盘剩余空间不低于 60G。

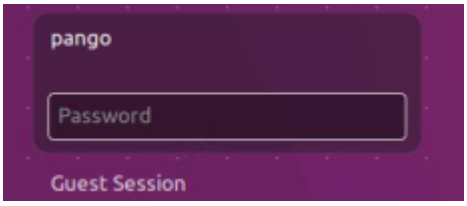
虚拟机压缩包在资料包\ubuntu 目录下，ubuntu 版本为 16.04，如果想使用自己的 ubuntu 开发环境，ubuntu 版本也需是 16.04。将压缩包拷贝到启动目录，解压。

解压完成后，打开 Vmware，选择打开虚拟机，如下图所示：

打开虚拟机后选择存放虚拟机文件夹，点击开启此虚拟机， 如下图所示：



第一次打开虚拟机时，系统提示“此虚拟机可能已被移动或复制”，选择“我已复制该虚拟机”。进入登录界面，如下图所示：



密码： root

此虚拟机安装了编译系统所用的交叉编译链、vscode 和 gedit 工具外，同时也已经安装了依赖包，包括：tofrodos、iproute、gawk、gcc、git-core、make、net-tools、ncurses-dev、libncurses5-dev、tftpd、zlib1g-dev、libssl-dev、flex、bison、lib32z1、lib32ncurses5、libbz2-1.0:i386、lib32stdc++6、libselinux1。

若想自行安装，按照如下命令安装即可：

```
sudo apt-get install tofrodos
sudo apt-get install iproute
sudo apt-get install gawk
sudo apt-get install gcc
sudo apt-get install git-core
sudo apt-get install make
sudo apt-get install net-tools
sudo apt-get install ncurses-dev
sudo apt-get install libncurses5-dev
sudo apt-get install tftpd
```

sudo apt-get install zlib1g-dev

sudo apt-get install libssl-dev

sudo apt-get install flex

sudo apt-get install bison

sudo apt-get install lib32z1

sudo apt-get install lib32ncurses5

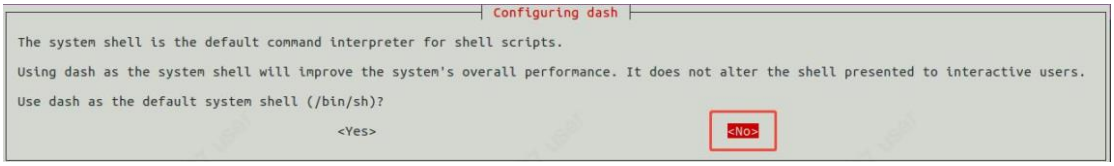
sudo apt-get install libbz2-1.0:i386

sudo apt-get install lib32stdc++6

sudo apt-get install libselinux1

sudo dpkg-reconfigure dash

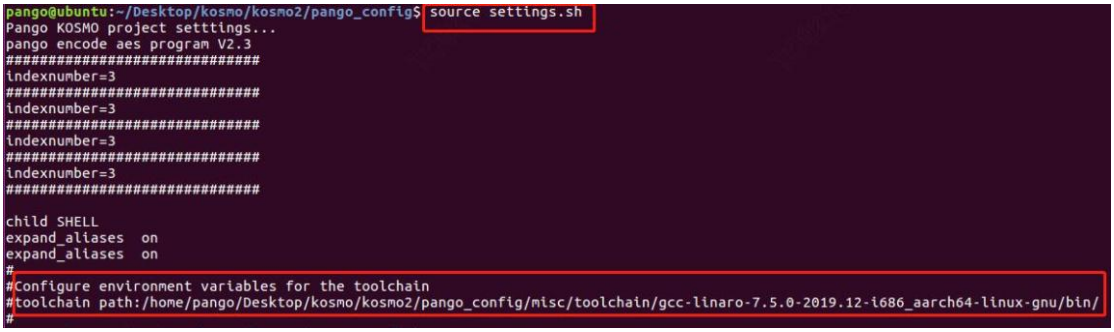
执行 sudo dpkg-reconfigure dash 命令的时候会弹出 Configuring dash 选择界面， 如下图所示：



选择<No>按钮。

1.3.2.交叉编译工具链配置

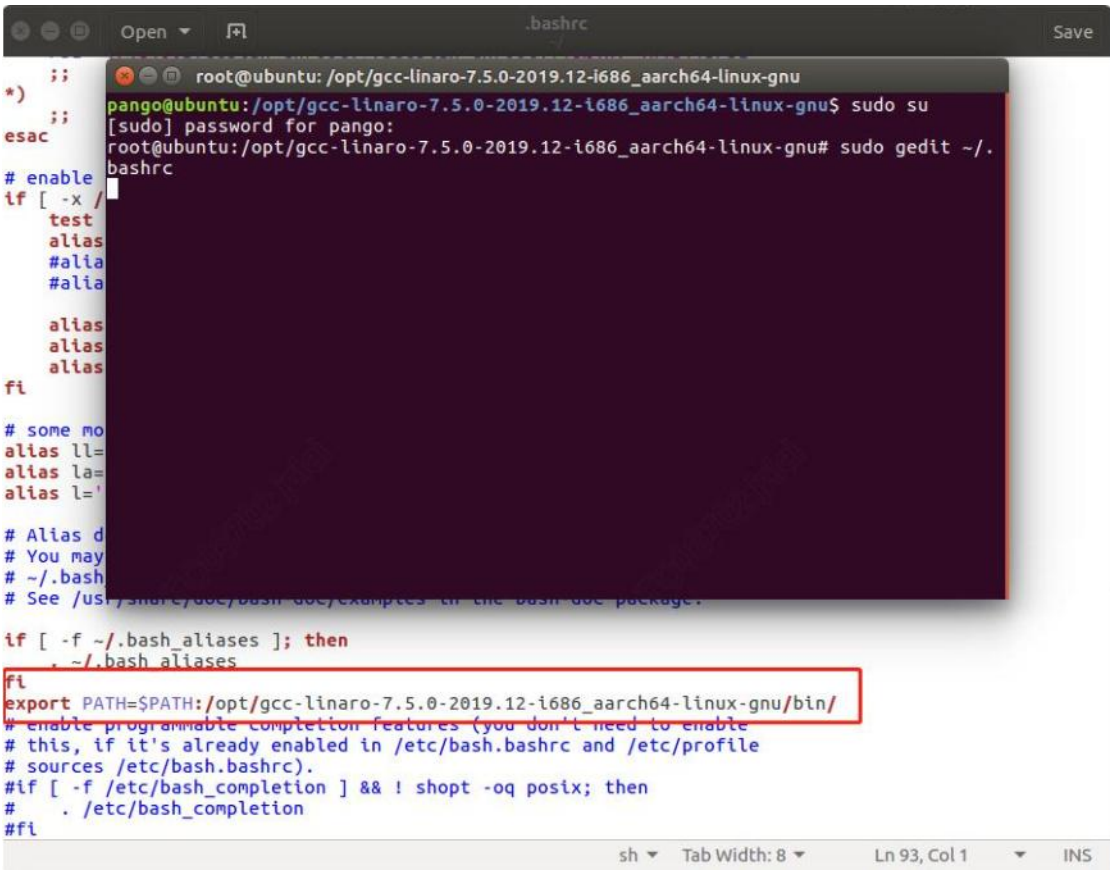
在当前控制台，进入 pango_config 目录，执行 source settings.sh，会在当前控制台部署好交叉编译工具链，如下图所示：



如果需要自己搭建交叉编译工具链，先把交叉编译工具链文件夹（交叉编译工具链目录：kosmo2/pango_config/toolchain/gcc-linaro-7.5.0-2019.12-i686_aarch64-linux-gnu）复制到/opt 目录，如下图所示：

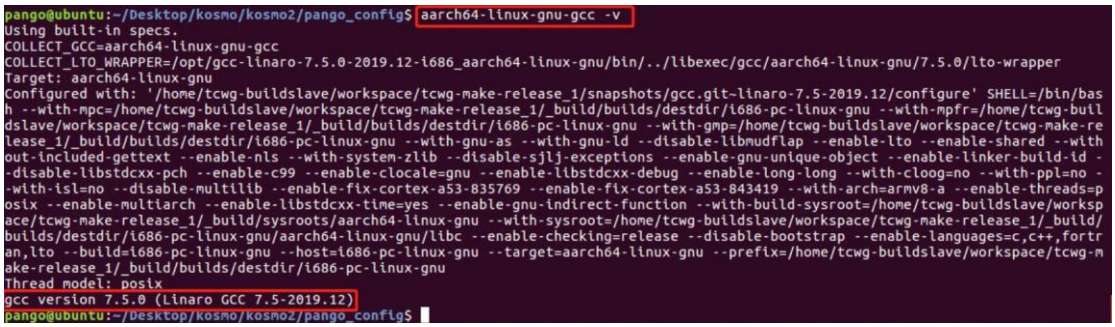
设置环境变量，在~/.bashrc 文件结尾添加如下信息：

export PATH=$PATH:/opt/gcc-linaro-7.5.0-2019.12-i686_aarch64-linux-gnu/bin/



配置生效命令： source ~/.bashrc

查看 gcc 版本，控制台输入命令：aarch64-linux-gnu-gcc -v



2.开发包目录说明

2.1.1.Kosmo 目录

1： atf 源码目录

2： pango_config 相关目录（包含命令、 工具链等）

3： kernel 源码目录

4： linux 驱动实例代码以及讲解

5： 文件系统源码目录

6： uboot 源码目录

7： readme

8： 发布说明

kosmo2.tar.gz： kosmo2 文件夹的压缩包， 作为备份用。

注意：

存放开发包的目录层级不能包含数字

2.1.2.Kosmo 目录

pango_config 目录说明， 如下图所示：

1： linux 打包工具， 生成 BOOT_PG.bin

2： 用户设备树自定义目录

3： hpc 文件存放目录（hpc 文件统一存放在这个目录）

4： fpga 执行文件（*.sbit） 存放目录 （PA 位流文件统一存放在这个目录）

5： icf 文件存放目录（icf 文件统一存放在这个目录）

6： images 目录， 存储编译源码（fsbl、 atf、 uboot、 kernel） 生成的目标文件

7： misc 杂项文件夹

8： rootfs 目录， 文件系统编译生成的目标文件

9： seetings.sh 脚本

2.1.3.Kosmo 目录

misc 目录说明， 如下图所示：

1： 存放自动设备树生成相关脚本

2： 存放 fsbl 编译相关脚本

3： 存放 cfg 配置相关源码

4： 存放 cfg 操作命令相关

5： 存放 edk 编译工具相关

6： 存放获取硬件平台信息相关脚本

7： 存放 atf、 u-boot、 kernel、 rootfs 交叉编译工具链

注意：

以上文件夹名称均不能修改。 若修改， 执行 pango 命令可能会导致未知问题。

3.Pango 命令说明

3.1.实验简介

实验目的：

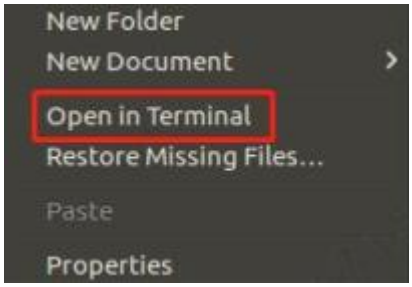
pango 操作命令共五种，分别是：pango-config、pango-build、pango-clear、pango-package、pango-info。下面对每个命令单独进行说明。

实验环境：

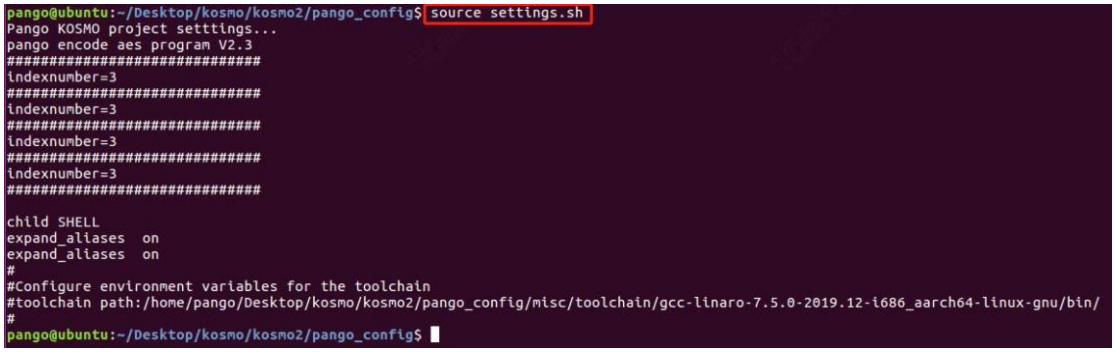
Linux

3.2.source settings.sh

在 pango_config 目录下点击右键，然后选择 Open in Terminal，打开控制台，如下图所示：



普通用户权限和 root 权限均支持编译，这里以普通用户权限为例，然后再执行 settings.sh（source settings.sh 命令）脚本，如下图所示：



settings.sh 脚本的作用：

实现 pango-config、pango-build、pango-clear、pango-package、pango-info 命令以及输入命令时可以 tab 键自动补全命令

配置当前控制台下交叉编译工具链可用

注意：

1.source settings.sh 是所有命令操作前的第一步，执行完后，才能执行接下来的所有命令操作。

2.配置脚本执行后，pango 命令只能在当前控制台生效，不能切换文件夹目录层级（即：只能在 pango_config 目录下操作），若重新打开一个控制台需要重新执行 source settings.sh。

3.3.pango-config

pango-config 包含 pango-config-hw、pango-config-c、pango-config-d、pango-config-s 和 pango-config -h 命令。

执行 pango-config -hw 命令，导入 hpc 文件，获取硬件平台信息

执行 pango-config -c 命令，打开配置界面，当于 make menuconfig。

执行 pango-config -d 命令，使能默认配置，当于 make xxx_deconfig。

执行 pango-config -s 命令，将当前的配置保存到 xxx_deconfig 文件。

执行 pango-config -h 命令，查看 pango-config 的命令操作讲解。

3.3.1.pango-config -hw

导入 hpc 文件，获取硬件平台信息。该命令需要传入 hpc 文件（请将 hpc 文件存放在 edk_hw 目录），命令操作（pango-config -hw edk_hw/pg2k400_676_demo_wrapper.hp）如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-config -hw edk_hw/pg2k400_676_demo_wrapper.hpc
This is pango config,Verion:1.5
pango config -hw edk_hw/pg2k400_676_demo_wrapper.hpc configureation.
#
#DEVICE="PG2K400" PACKAGE="FFBG676"
#
#The edk.tar.gz file path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/edk_tool/kosmo2-pg2k400/edk.tar.gz
#
#Wait about 20s to unpack edk.tar.gz for the first time
#
#Successfully obtain product hardware information
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-build
```

获取芯片的 device type 和硬件信息文件（ *.hdd），其中 device type 用于自动生成 Kconfig.autoset 和 Kconfig.syshw 配置文件、 匹配 uboot 和 kernel 的 deconfig 文件以 kernel 的平台设备树文件； *.hdd 文件用于自动设备树生成和 fsbl 编译。

注意：

1.pango-config -c edk_hw/xxx.hpc 命令操作是继 source settings.sh 之后的第二步操作， 执行完该步骤，才能执行接下来的 pango-build 、 pango-config 相关命令操作（主要针对 cfg、 fsbl、 atf、 uboot、kernel）。

2.若是第一次执行该命令，需要解压 edk.tar.gz 文件，大约等待 20s，后续再次导入 hpc 文件则无需等待。

3.3.2.pango-config -d

执行 pango-config -d 命令，能默认配置，相当于 make xxx_deconfig。

pango-config -d xxx（xxx 可以是： cfg、 uboot、 kernel、 rootfs）命令是将使能相应源码的 deconfig 文件，将默认配置写入.config 文件。以 ango-config -d cfg 为例讲解。

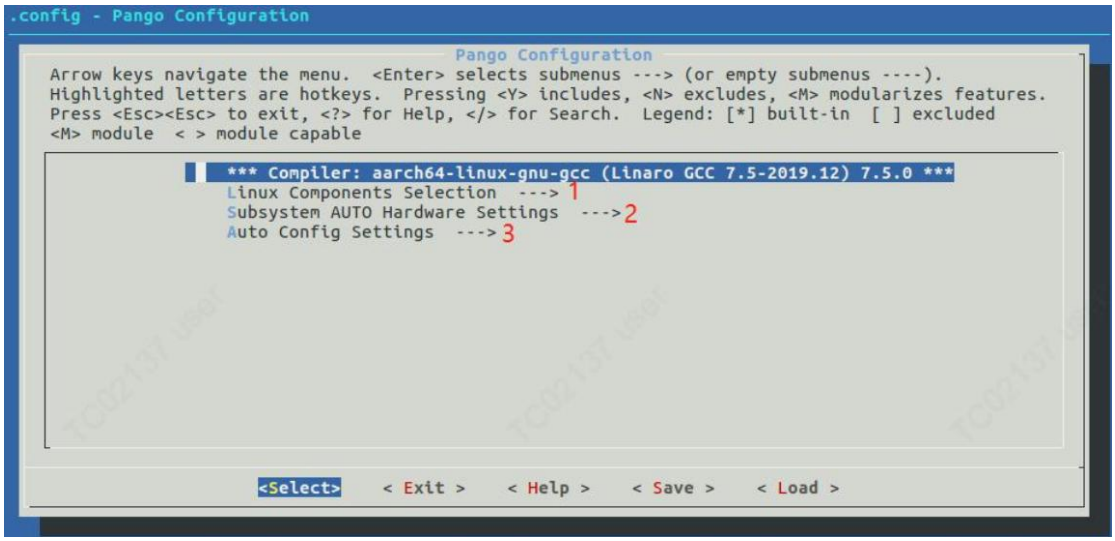
执行 pango-config -d cfg 命令，如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-config -d cfg
This is pango config,Verion:1.5
pango config -d cfg configureation.
#
#pango-config compilation.
#source_path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/cfg
#
do make pango_kosmo_cfg_defconfig --> cfg
  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
  Find default configuration "configs/pango_kosmo_cfg_defconfig"
#
# configuration written to .config
#
```

3.3.3.pango-config -c

执行 pango-config -c 命令（默认会执行 make xxx_deconfig 操作，但仅是第一次执行 pango-config -c 命令的时候执行 make xxx_deconfig 操作），可显示当前最新配置信息。pango-config -c xxx（xxx 可以是：cfg、uboot、kernel、rootfs、auto-dts，其中 pango-config-c auto-dts 是用于调试自动设备树生成的命令，仅是生成 kosmo-cfg.dtsi 和 kosmo-macro.h 文件，不将文件拷贝到内核设备树目录下）命令主要是做源码编译前的配置，以 pango-config -ccfg 为例讲解。

执行 pango-config -c cfg 命令，打开 cfg 配置界面，如下图所示：



说明：

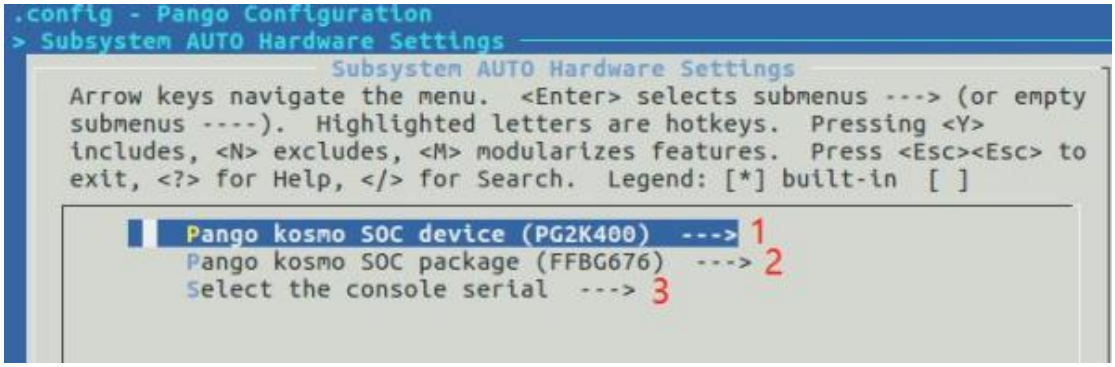
1. 如果关闭（atf、uboot、kernel、rootfs）源码配置选项，则必须保证源码的文件夹和 pango_config 文件夹在同一级目录并且源码文件夹名字不被修改（atf 源码文件夹：pango_atf；uboot 源码文件夹：pango_uboot；kernel 源码文件夹：pango_kerne

l； rootfs 源码文件夹： pango_rootfs） ， 如果路径或文件夹名不一致， 编译会发生未知错误。

2. 如果打开了源码配置选项， 并且手动配置了源码路径， 则没有该限制。

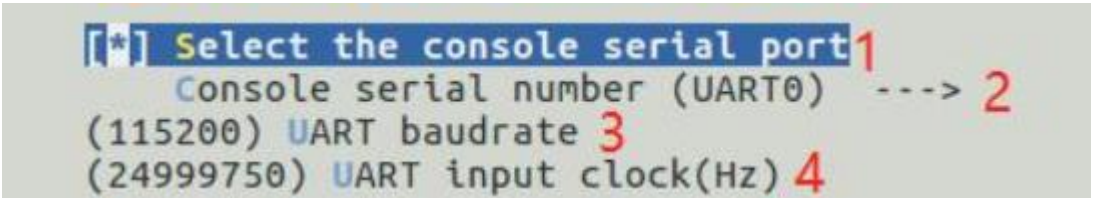
Subsystem AUTO Hardware Settings

进入系统自动硬件配置选项， 如下图所示：



1. SOC device， 该选项不可修改， 自动获取
2. SOC package， 该选项不可修改， 自动获取
3. 串口控制台配置选项

其中 Select the console serial 为串口控制台配置选项， 主要是 PU 侧串口配置，如下图所示



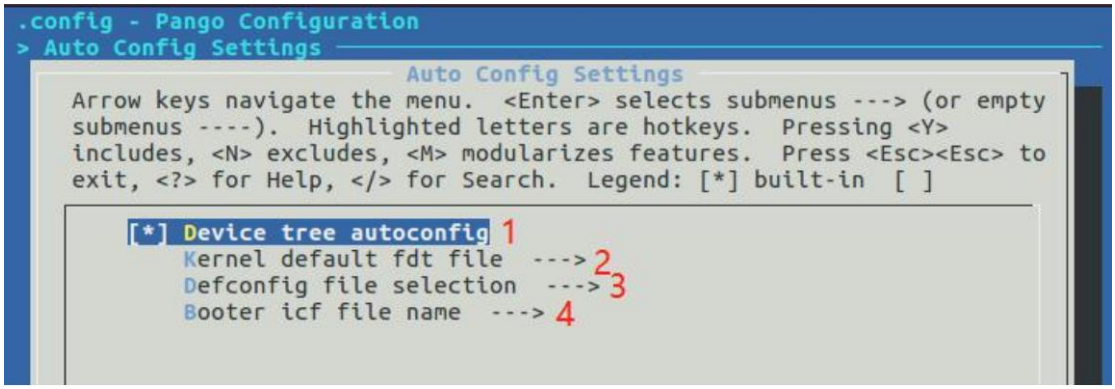
1. 串口控制台选择使能开关， 默认打开， 表示使用 cfg 自动配置
2. 串口编号配置
3. 串口波特率配置
4. 串口控制器时钟配置

自动获取串口编号、 串口波特率、 串口控制器时钟， 如果 PU 同时使能了串口 0 和串口 1，默认以串口 0 作为控制台， 若需要切换控制台串口编号， 可手动修改， 进入 Console serialnumber 配置项进行选择， 如下图所示：

串口波特率或串口控制器时钟的修改可参考串口编号的修改方法。 进入配置选项，直接修改数值。

Auto Config Settings

进入自动配置选项，如下图所示：

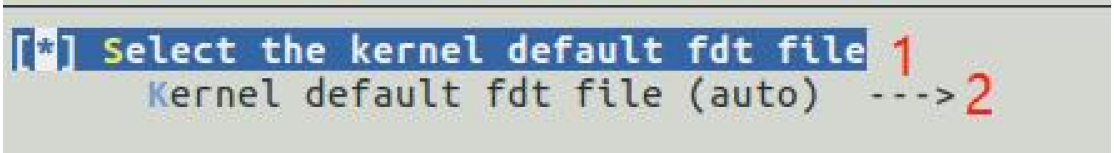


1. 自动设备树生成配置选项， 默认打开
2. kernel 设备树文件配置选项
3. uboot、 kernel 默认 deconfig 文件配置选项
4. 打包工具默认 icf 文件配置选项

Device tree autoconfig 选项为自动设备树生成使能开关， 默认打开， 即每次编译内核都会自动生成 kosmo-cfg.dtsi 和 kosmo-macro.h， 并且拷贝到 kernel 的/arch/arm64/boot/dts/pango 目录下。 若不需要自动生成， 关闭该配置项即可。

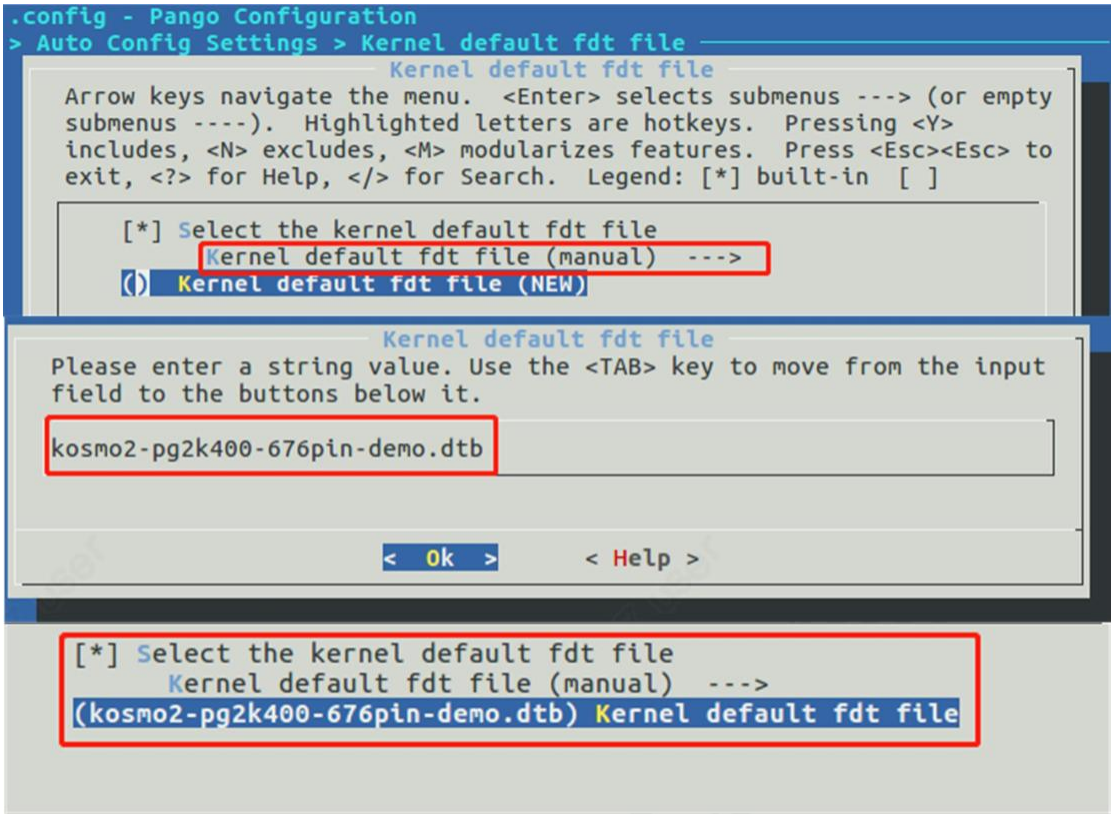
Kernel default fdt file、 Defconfig file selection 和 Booter icf file name 配置项分别是： kernel 平台设备树文件配置、uboot 和 kernel 的 deconfig 文件配置以及位流打包 icf 文件配置。以 Kerneldefault fdt file 选项为例讲解， 剩余两个选项配置方法类似。

进入 Kernel default fdt file 配置选项， 如下图所示：



1. kernel 平台设备树文件配置使能开关， 默认打开
2. kernel 平台设备树文件配置模式选择(auto/manual)

Kernel default fdt file 配置选项默认使用 auto 模式， 根据 SOC device type 自动匹配平台设备树文件。 如果需要使用自己定义的平台设备树文件， 该配置选项改为手动，并且需要配置平台设备树文件名， 如下图所示：



3.3.4.pango-config -s

执行 pango-config -c 命令后， 对配置有更改， 如果需要将修改项保存到 xxx_deconfig 文件， 执行 pango-config -s 命令。

pango-config -s xxx （ xxx 可以是： cfg、 uboot、 kernel、 rootfs） 命令是将修改项保存到 xxx_deconfig 文件， 以 pango-config -s cfg 为例讲解。

执行 pango-config -s cfg 命令， 如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-config -s cfg
This is pango config,Verion:1.5
pango config -s cfg configureation.
#
#pango-config compilation.
#source_path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/cfg
#
do make savedefconfig --> cfg
Save defconfig to pango_kosmo_cfg_defconfig file
scripts/kconfig/conf  --savedefconfig=defconfig Kconfig
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

3.3.5.pango-build

编译源码命令 pango-build， 可单独使用也可 pango-build -c 一起使用。

执行 pango-build 命令， 编译全部（cfg、 fsbl、 atf、 uboot、 kernel， 不包含 root fs） 源码。

执行 pango-build -c rootfs 命令， 编译 rootfs（因 rootfs 编译耗时较久， 单独编译 rootfs） 。

pango-build -c xxx（xxx 可以是： cfg、 fsbl、 atf、 uboot、 kernel、 rootfs） 命令主要是指定需要编译的源码， 以 pango-build -c cfg 为例进行讲解。

执行 pango-build -c cfg 命令， 如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-build -c cfg
This is pango build,Verion:1.4
pango build -c cfg configureation.
#
#pango-config compilation.
#source_path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/cfg
#
do make all --> cfg
scripts/kconfig/conf  --syncconfig Kconfig
  GEN     include/autoconf.mk.dep
  CFG     pango_cfg.h
  GEN     include/autoconf.mk
make: Nothing to be done for 'all'.
#
#Use current atf source path:/home/pango/Desktop/kosmo/kosmo2/pango_atf
#Use current uboot source path:/home/pango/Desktop/kosmo/kosmo2/pango_uboot
#Use current kernel source path:/home/pango/Desktop/kosmo/kosmo2/pango_kernel
#Use current rootfs source path:/home/pango/Desktop/kosmo/kosmo2/pango_rootfs
#
#
#Automatic generation device tree configuration option enabled
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

该命令执行完成后， 可获取到当前 atf、 uboot、 kernel、 rootfs 的源码路径。

注意：

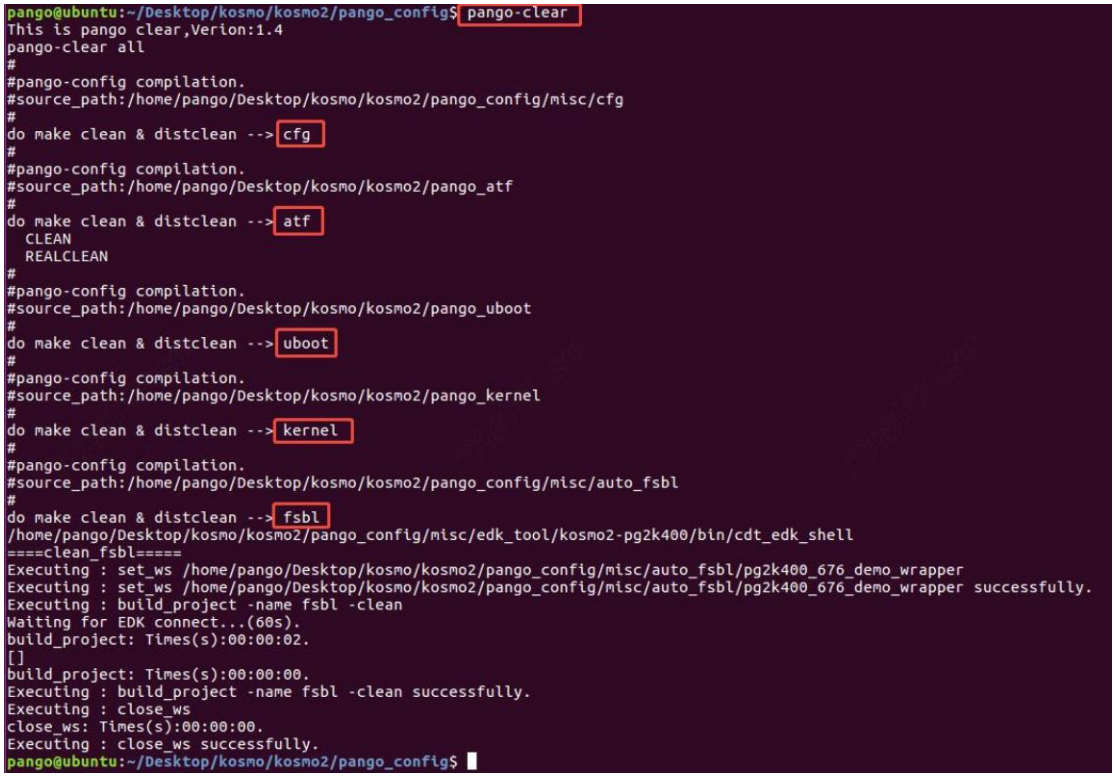
pango-build -c cfg 命令操作是继 pango-config -c edk_hw/xxx.hpc 命令之后的第三步操作， 执行完该步骤， 才能执行接下来的 pango-build 、 pango-config、 pango-info 相关命令操作（主要针对 fsbl、 atf、uboot、 kernel） 。

fsbl、 atf、 uboot、 kernel 编译生产的目标文件会自动拷贝到 pango_config/images 目录； rootfs 编译生成的目标文件会自动拷贝到 pango_config/rootfs 目录。

执行 pango- build -h 命令， 查看 pango- build 的命令操作讲解。

3.3.6.pango-clear

执行 pango-clear 命令： 清除所有源码（cfg、 atf、 uboot、 kernel、 fsbl， 不含 rootfs） 编译生成的临时文件， 如下图所示：



执行 pango-clear -c cfg 命令， 清除 cfg 编译生成的临时文件。

执行 pango-clear -c fsbl 命令， 清除 fsbl 编译生成的临时文件。

执行 pango-clear -c atf 命令， 清除 atf 编译生成的临时文件。

执行 pango-clear -c uboot 命令， 清除 uboot 编译生成的临时文件。

执行 pango-clear -c kernel 命令， 清除 kernel 编译生成的临时文件。

执行 pango-clear -c rootfs 命令， 清除 rootfs 编译生成的临时文件。

执行 pango-clear -h 命令， 查看 pango-clear 的命令操作讲解。

3.3.7.pango-package

pango-package 命令是把 fsbl.elf、 *.sbit（FPGA 位流文件） 以及其他类型文件打包合并生成 BOOT_PG.bin， 其中 fsbl.elf 有三种方式产生：

第一种： 手动搭建 EDK FSBL 工程并编译， 将生成的 elf 文件拷贝到 pango_config/images 目录下， 并命名为： fsbl.elf。

第二种： 执行 pango-build -c fsbl 命令， 创建 fsbl 工程并编译， 自动将 fsbl.elf 拷贝到 pango_config/images 目录下。

第三种： 使能 Auto Compile First Stage Bootloader(FSBL)配置选项， 执行 pango-build 全编译命令， 会自动创建 fsbl 工程并编译（该功能已默认打开） ， 编译完成， 自动将 fsbl.elf 拷贝到 pango_config/images 目录下。

执行该命令之前， 需要确认 atf、 uboot、 kernel、 fsbl 已经正确编译完成， 否则会找不到相关 elf 文件。

pango-package 命令有两种用法：

第一种： 直接输入 pango-package

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-package
This is pango package,Verion:1.2
#
#Use the entered icf file:./icf/kosmo.icf 1
#
#pango-config compilation.
#
[Booter]Work path: /home/pango/Desktop/kosmo/kosmo2/pango_config/booter/bin
[Booter]Finish. Generated image file in: /home/pango/Desktop/kosmo/kosmo2/pango_config/images/BOOT_PG.bin. 2

pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

直接输入 pango-package 命令， 这里默认使用 cfg 里面配置的 icf 文件名称。

1： icf 文件路径（kosmo.ifc 文件自动生成）

2： BOOT_PG.bin 文件输出路径

第二种： pango-package kosmo.icf， pango-package 加 icf 文件或文件路径（即： pango-packageicf/kosmo.icf） 的命令方式， 在当前 pango_config 目录下的所有文件中查找 kosmo.icf 文件

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-package kosmo.icf
This is pango package,Verion:1.2
#
#Use the entered icf file:./icf/kosmo.icf 1
#
#pango-config compilation.
#
[Booter]Work path: /home/pango/Desktop/kosmo/kosmo2/pango_config/booter/bin
[Booter]Finish. Generated image file in: /home/pango/Desktop/kosmo/kosmo2/pango_config/images/BOOT_PG.bin.

pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

1：查找到的 kosmo.icf 文件路径

执行 pango- package -h 命令， 查看 pango- package 的命令操作讲解。

```
1 //arch = PG2K400; split = false; format = BIN
2 boot_image:
3 {
4     [mnft_sign] RSASSA_1024_PKCS_V21_SHA1
5     [puk_sign] SHA224
6     [puk_file] ../rsc/rsa_1024_pubkey.pem
7     [prk_file] ../rsc/rsa_1024_privkey.pem
8     [img_hash_sign] SHA224
9     [nky] ../rsc/aes.nky
10    [enc] false
11    [secure_boot] false
12    [prim:bootloader] ../../images/fsbl.elf 1
13    //[prim:datafile,space=pa] ../../fpga/TOP.sbit 2
14    [prim:datafile,space=pu,load=0x06000000,entry=0x06000000] ../../images/Image 3
15    [prim:datafile,space=pu,load=0x04000000,entry=0x04000000] ../../images/kosmo2-pg2k400-676pin-demo.dtb 4
16    //[prim:datafile,space=pu,load=0x04000000,entry=0x04000000] ../../images/kosmo2-pg2k400-900pin-demo.dtb
17    [prim:bootloader] ../../images/u-boot.elf 5
18    [prim:bootloader] ../../images/bl31.elf 6
19    [out_dir] ../../images/ 7
20 }
21
```

3.3.8.icf 简单说明

当前提供的 icf 模版文件为 2k400_all_in.icf。

2k400_all_in.icf：

1： fsbl.elf 文件路径（必须放在第一位， bootrom 默认引导第一位置的 elf 文件）；

2： fpga 的 sbit 文件路径；

3： kernel 镜像文件路径；

4： kernel 设备树文件路径；

5： u-boot.elf 文件路径；

6： bl31.elf 文件路径（必须放在最后一位， fsbl 默认跳转最后一个应用程序）；

7： BOOT_PG.bin 文件输出路径；

icf 文件结构说明可以参考/booter/doc/Booter_User_Guide.pdf 文档。

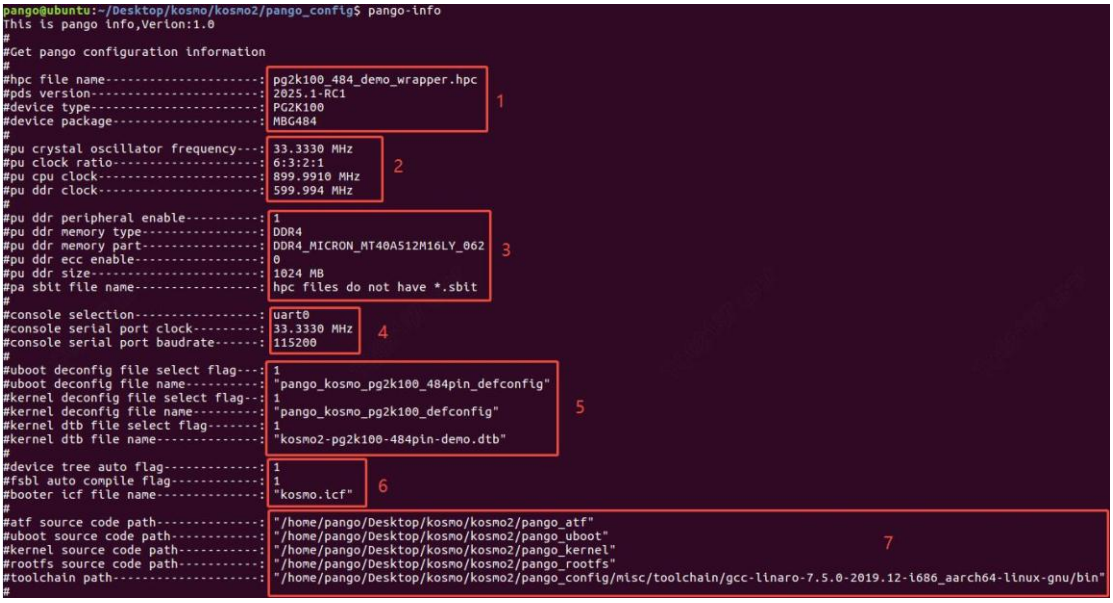
注意：

以上涉及到的 elf 文件、 sbit 文件均为测试用， 不匹配我司提供的 Demo 板， 需要根据自己实际的工程替换。

3.4.pango-info

pango-info 命令是能够方便的获取当前的配置信息， 包括硬件配置和 cfg 的配置，执行该命令之前， 确保已经执行 pango-build -c cfg 命令。

pango-info 命令执行结果如下图所示：



1： 硬件配置信息

2： 晶振频率、 时钟比、 CPU 主频、 DDR 主频

3： DDR 配置信息、 是否包含 PA 位流

4： 串口配置信息

5： uboot 和 kernel 的相关配置信息

6： 自动设备树生成和 fsbl 自动编译标记

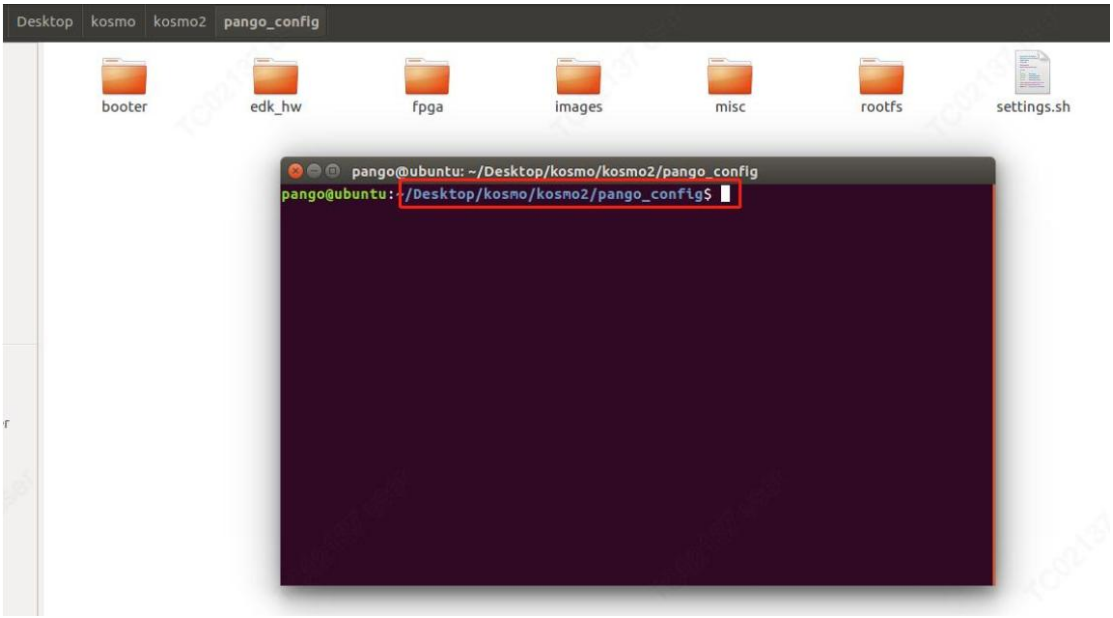
7： 源码路径以及交叉编译工具链路径

4.定制 Linux 系统

4.1.实验简介

资料包已经提供了相关源码， 可参考第 2 节内容。 下面讲解如何生成 atf 的 elf、uboot 的 elf、 内核 Image、 内核设备树。

Step1： 进入 pango_config 目录， 打开控制台。



普通用户权限和 root 权限（密码： root） 均支持编译， 这里以普通用户权限为例。

Step2：执行 setting.sh 脚本 source settings.sh。

```
pango@ubuntu: ~/Desktop/kosmo/kosmo2/pango_config
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ source settings.sh
Pango KOSMO project setttings...
pango encode aes program V2.3
############################
indexnumber=3
############################
indexnumber=3
############################
indexnumber=3
############################
indexnumber=3
############################

child SHELL
expand_aliases  on
expand_aliases  on
#
#Configure environment variables for the toolchain
#toolchain path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/toolchain/gcc-linaro-7.5.0-2019.12-i686_aarch64-linux-gnu/bin/
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

Step3：导入 hpc 文件， pango-config -hw edk_hw/pg2k400_676_demo_wrapper.hpc。

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-config -hw edk_hw/pg2k400_676_demo_wrapper.hpc
This is pango config,Verion:1.5
pango config -hw edk_hw/pg2k400_676_demo_wrapper.hpc configureation.
#
#DEVICE="PG2K400" PACKAGE="FFBG676"
#
#The edk.tar.gz file path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/edk_tool/kosmo2-pg2k400/edk.tar.gz
#
#Wait about 20s to unpack edk.tar.gz for the first time
#
#Successfully obtain product hardware information
#
```

Step4：一键编译 atf、 uboot、 kernel、 fsbl，执行 pango-build 命令。

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-build
This is pango build,Verion:1.4
pango build all..
#
#pango-config compilation.
#source_path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/cfg
#
do make all --> cfg
scripts/kconfig/conf  --syncconfig Kconfig
  CFG     pango_cfg.h
  GEN     include/autoconf.mk.dep
  GEN     include/autoconf.mk
make: Nothing to be done for 'all'.
#
#Use current atf source path:/home/pango/Desktop/kosmo/kosmo2/pango_atf
#Use current uboot source path:/home/pango/Desktop/kosmo/kosmo2/pango_uboot
#Use current kernel source path:/home/pango/Desktop/kosmo/kosmo2/pango_kernel
#Use current rootfs source path:/home/pango/Desktop/kosmo/kosmo2/pango_rootfs
#
#
#Automatic generation device tree configuration option enabled
#
#
```

接下来就会默认全编译（cfg、atf、uboot、kernel、fsbl），整个编译过程比较久。

成功编译最后会停留在 fsbl 编译完成的状态，控制台界面如下：

```
  LD [M]  drivers/mtd/tests/mtd_speedtest.ko
  LD [M]  drivers/mtd/tests/mtd_oobtest.ko
  LD [M]  drivers/usb/gadget/legacy/g_mass_storage.ko
  LD [M]  net/ipv6/sit.ko
  LD [M]  drivers/mtd/tests/mtd_stresstest.ko
  LD [M]  drivers/mtd/tests/mtd_subpagetest.ko
  LD [M]  drivers/mtd/tests/mtd_readtest.ko
  LD [M]  drivers/mtd/tests/mtd_torturetest.ko
  LD [M]  drivers/remoteproc/kosmo_remoteproc.ko
  LD [M]  drivers/rpmsg/rpmsg_user_dev_driver.ko
  LD [M]  drivers/rpmsg/virtio_rpmsg_bus.ko
  LD [M]  drivers/rpmsg/rpmsg_core.ko
  LD [M]  net/bridge/br_netfilter.ko
  LD [M]  net/ipv4/tunnel4.ko
  LD [M]  drivers/usb/gadget/libcomposite.ko
  LD [M]  drivers/usb/gadget/function/usb_f_mass_storage.ko
#
#pango-config compilation.
#source_path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/auto_fsbl
#
do make all --> fsbl
/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/edk_tool/kosmo2-pg2k400/bin/cdt_edk_shell
====create_fsbl_and_build=====
Executing : set_ws /home/pango/Desktop/kosmo/kosmo2/pango_config/misc/auto_fsbl/pg2k400_676_demo_wrapper
Executing : set_ws /home/pango/Desktop/kosmo/kosmo2/pango_config/misc/auto_fsbl/pg2k400_676_demo_wrapper successfully.
Executing : create_hdd -path /home/pango/Desktop/kosmo/kosmo2/pango_config/edk_hw/pg2k400_676_demo_wrapper.hpc
Waiting for EDK connect...(60s).
create_hdd: Times(s):00:00:03.
[]
create_hdd: Times(s):00:00:00.
Executing : create_hdd -path /home/pango/Desktop/kosmo/kosmo2/pango_config/edk_hw/pg2k400_676_demo_wrapper.hpc successfully.
Executing : create_app -app fsbl -name fsbl -hdd pg2k400_676_demo_wrapper_hw_platform
[]
create_app: Times(s):00:00:03.
Executing : create_app -app fsbl -name fsbl -hdd pg2k400_676_demo_wrapper_hw_platform successfully.
Executing : build_project -name fsbl -build
[]
build_project: Times(s):00:00:09.
Executing : build_project -name fsbl -build successfully.
Executing : close_ws
close_ws: Times(s):00:00:00.
Executing : close_ws successfully.
```

Step5：检查镜像是否编译成功

最后生成的镜像文件会被拷贝到*/pango_config/images 目录下，如下图所示：

Step6：编译文件系统（若有需要，选择编译，耗时比较久） pango-build -c rootfs

5.SD 卡与 Linux 启动

启动 Linux 的方式有 SD 卡启动、 QSPI 启动、 NAND 启动， 这里以 SD 卡启动为例，说明 Linux 启动过程。

5.1.打包位流

将 fsbl、atf、uboot 编译生成的 elf 文件打包成 BOOT_PG.bin 文件，参考 3.3.7 小节，或参考《TS130002_Kosmo 系列 EDK 实验手册》 BOOT_PG.bin 的生成流程。

5.2.SD 卡分区

本小节主要说明全新 SD 卡如何分区，如果 SD 卡已经分好区的可跳过该小节。

Step1：解压分区软件

这里推荐 DiskGenius，这是一款免安装的免费软件。 安装包位于资料包中，如下图所示：

解压后进入到解压目录，双击 DiskGenius.exe 就可以打开使用。

Step2： 删除分区

插上读卡器，分区软件就会识别 SD 卡，如下图所示：

删除分区，操作步骤如下图所示

Step3：新建 BOOT 分区

选中磁盘，鼠标右键选择新建分区，弹出新建分区界面，如下图所示：

按照上图配置， BOOT 分区为 FAT32 文件系统类型， 配置完成点击确认按钮。

Step4：新建 ROOTFS 分区

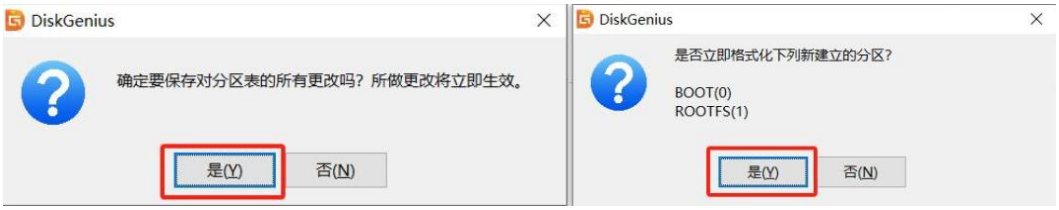
选中未分区的磁盘空间（灰色区域），鼠标右键选择新建分区，弹出新建分区界面，如下图所示：

按照上图配置，ROOTFS 分区为 Ext4 文件系统类型，配置完成点击确认按钮。

Step5： 保存更改

点击分区软件左上角的保存更改按钮， 如下图所示：

保存过程中会弹出如下图的两个界面， 全部选是。



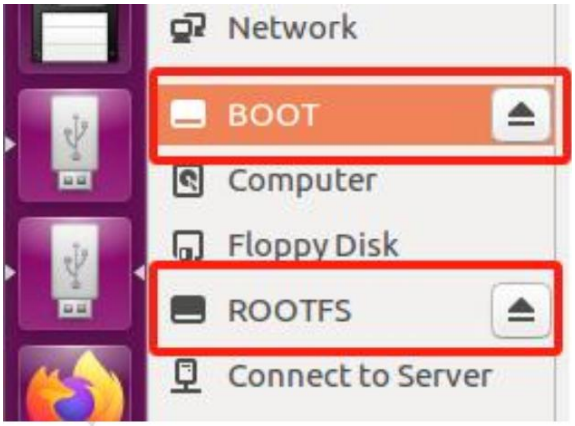
保存完成后，界面如下图所示：

关闭分区软件，将 SD 卡连接到虚拟机，在虚拟机右下角，有个 U 盘形状的小图标，

如下图所示：

点击，选择连接。

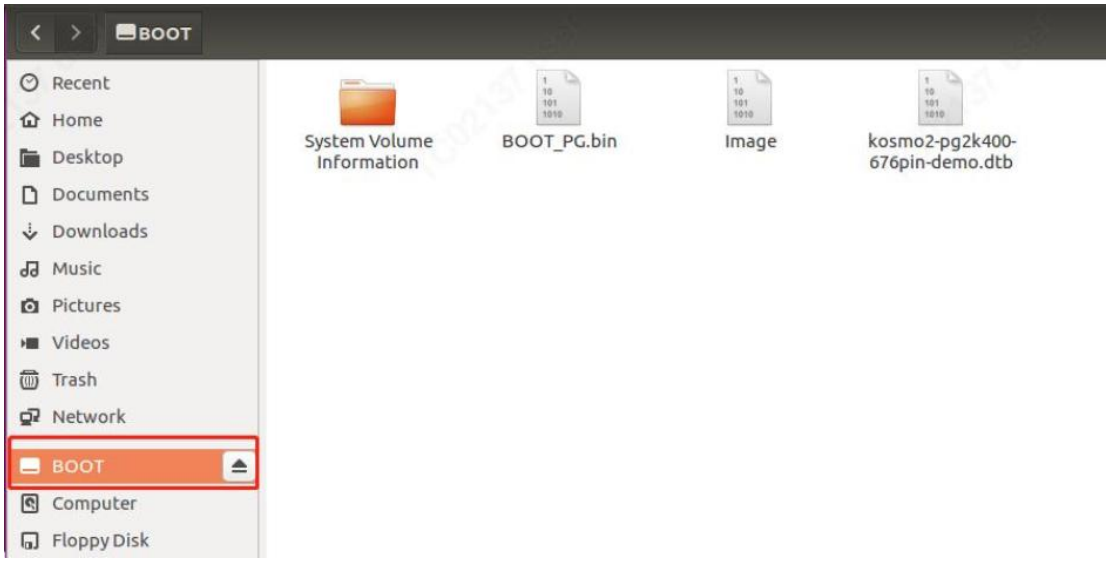
识别到 SD 卡分区，如下图所示：



SD 卡分区的操作流程到此结束。

5.3.启动文件拷贝到 BOOT 分区

将 BOOT_PG.bin、 Image、 kosmo2-pg2k400-676pin-demo.dtb 这三个文件都拷贝到 BOOT 分区， 如下图所示：



5.4.文件系统拷贝到 ROOTFS 分区

将/home/pango/Desktop/kosmo/kosmo2/pango_config/rootfs 文件夹里面的所有文件拷贝到 ROOTFS 分区， 打开控制台， 进入 root 权限， 执行如下命令：

```
cd /home/pango/Desktop/kosmo/kosmo2/pango_config/rootfs
cp -rf * /media/pango/ROOTFS
cd /media/pango/ROOTFS
chmod -R 777 *
sync
```

ROOTFS 分区里面的内容如下图所示：

最后点击 BOOT/ROOTFS 分区右边的三角， 弹出读卡器（重要步骤） ， 如下图所示：

6.上电启动 Demo 板

将 SD 卡插入开发板， 接好电源线、 USB-Type-C 数据线接入 PU_UART、 配置启动跳帽 ， 如下图所示：

```
[    1.142578] of_cfs_init
[    1.149197] of_cfs_init: OK
[    1.151657]  mmcblk0: p1 p2
[    1.155189] dw-apb-uart e5000000.serial: forbid DMA for kernel console
[    1.551782] EXT4-fs (mmcblk0p2): warning: mounting fs with errors, running e2
fsck is recommended
[    1.606237] EXT4-fs (mmcblk0p2): recovery complete
[    1.656643] EXT4-fs (mmcblk0p2): mounted filesystem with ordered data mode. O
pts: (null)
[    1.664521] VFS: Mounted root (ext4 filesystem) on device 179:2.
[    1.672356] devtmpfs: mounted
[    1.676374] Freeing unused kernel memory: 1472K
[    1.680962] Run /sbin/init as init process
[    2.379495] EXT4-fs (mmcblk0p2): re-mounted. Opts: (null)
[    2.460099] random: fast init done
Starting syslogd: OK
Starting klogd: OK
Running sysctl: OK
Initializing random number generator: OK
Saving random seed: OK
Starting rpcbind: OK
Starting network: OK
Starting NFS statd: OK
Starting NFS services: OK
Starting NFS daemon: rpc.nfsd: Unable to access /proc/fs/nfsd errno 2 (No such file or directory).
Please try, as root, 'mount -t nfsd nfsd /proc/fs/nfsd' and then restart rpc.nfsd to correct the problem
FAIL
Starting NFS mountd: OK

Welcome to pango root
buildroot login:
```

登录系统， 登录名： root， 密码： root。

```
Welcome to pango root
buildroot login: root
Password:
# cd /
# ls
bin         lib         media       root        tmp
dev         lib64       mnt         run         usr
etc         linuxrc     opt         sbin        var
home        lost+found  proc        sys
#
```

7.常见问题

7.1.未导入 hpc 文件提示

执行 source settings.sh 过后， 若直接执行 pango-build 或 pango-config 相关操作，会提示

先执行 hpc 文件导入命令， 如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-build -c cfg
This is pango build,Verion:1.4
pango build -c cfg configureation.
#
#Please execute the command first: pango-config -hw edk_hw/xxx.hpc
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

图 8- 1 hpc 文件导入命令提示

按照提示执行命令即可。

7.2.未执行 pango-build -c cfg 命令提示

执行 hpc 文件导入命令、 pango-cler -c cfg 命令或 pango-cler 命令后， 若直接执行 atf、 uboot 或 kernel 的编译或配置操作， 会提示优先执行 pango-build -c cfg， 如下图所示：

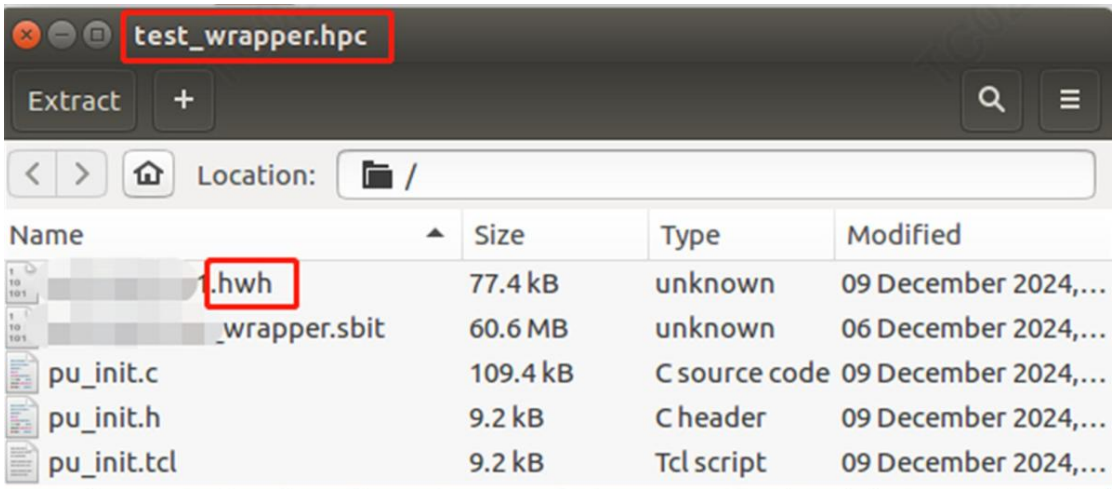
```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-build -c atf
This is pango build,Verion:1.4
pango build -c atf configureation.
#
#Hardware information has been updated
#Please execute the command first: pango-build -c cfg
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

图 8- 2 执行 pango-build -c cfg 命令提示

按照提示执行命令即可。

7.3.hdd 文件不存在提示

旧版本的 PDS 生成的 hpc 文件里面包含的硬件配置信息文件后缀名为.hwh， 不支持 hpc 文件导入功能， 如下图所示：




```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-config -hw edk_hw/test_wrapper.hpc
This is pango config,Verion:1.5
pango config -hw edk_hw/test_wrapper.hpc configureation.
#
#No files with '*.hdd' suffix found.
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

支持 hpc 导入功能的 PDS 版本为 PDS_2024.2 及其以上。

7.4.PU 串口 0 和串口 1 均没使能提示

在生成 HPC 文件里面如果 PU 的串口 0 和串口 1 均没有使能， 会有警告信息提示， 如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-config -hw edk_hw/test_900_wrapper.hpc
This is pango config,Verion:1.5
pango config -hw edk_hw/test_900_wrapper.hpc configureation.
#
#DEVICE="PG2K400" PACKAGE="FFBG900"
#
#The edk.tar.gz file path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/edk_tool/kosmo2-pg2k400/edk.tar.gz
#
#Successfully obtain product hardware information
#
#WARNING: Either serial port 0 nor serial port 1 of the PU is enabled.The serial console may not print anything!!!
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

图 8- 4 PU 串口 0 和串口 1 均没使能

出现这种情况， 可能会导致串口控制台没有输出。

7.5.解压错误提示

解压 hpc 文件提示解压错误， 如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-config -hw edk_hw/test02_900_wrapper.hpc
This is pango config,Verion:1.5
pango config -hw edk hw/test02 900 wrapper.hpc configureation.
  End-of-central-directory signature not found.  Either this file is not
  a zipfile, or it constitutes one disk of a multi-part archive.  In the
  latter case the central directory and zipfile comment will be found on
  the last disk(s) of this archive.
unzip:  cannot find zipfile directory in one of temp_hpc/test02_900_wrapper.hpc or
        temp_hpc/test02_900_wrapper.hpc.zip, and cannot find temp_hpc/test02_900_wrapper.hpc.ZIP, period.
#
#No files with '*.hdd' suffix found.
#
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

图 8- 5 解压错误提示

大概率是压缩包有问题， 导致解压失败， 未找到 hdd 文件。

7.6.Waiting for EDK connect...(60s).多次提示

执行 pango-build -c fsbl 或 pango-clear -c fsbl 命令时， 若多次打印 Waiting for EDKconnect...(60s)， 将显示等待连接、 超时等错误信息， 如下图所示：

```
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$ pango-clear -c fsbl
This is pango clear,Verion:1.4
pango-clear -c fsbl
#
#pango-config compilation.
#source_path:/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/auto_fsbl
#
do make clean & distclean --> fsbl
/home/pango/Desktop/kosmo/kosmo2/pango_config/misc/edk_tool/kosmo2-pg2k400/bin/cdt_edk_shell
====clean_fsbl=====
Executing : set_ws /home/pango/Desktop/kosmo/kosmo2/pango_config/misc/auto_fsbl/676_demo_wrapper
Executing : set_ws /home/pango/Desktop/kosmo/kosmo2/pango_config/misc/auto_fsbl/676_demo_wrapper successfully.
Executing : build project -name fsbl -clean
Waiting for EDK connect...(60s).
Waiting for EDK connect...(60s).
Waiting for EDK connect...(60s).
Error connect with EDK font.
build_project: Times(s):00:01:00.
pango@ubuntu:~/Desktop/kosmo/kosmo2/pango_config$
```

图 8- 6 Waiting for EDK connect...(60s).多次提示

建议删除 auto_fabl 目录下的*_wrapper 文件夹， 如下图所示：

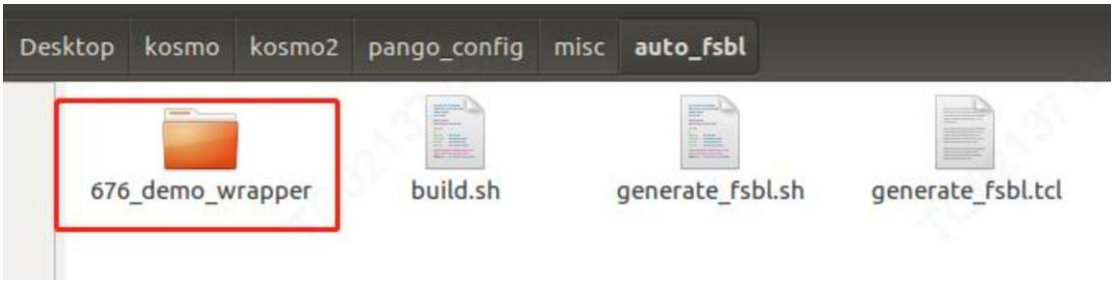


图 8- 7 删除*_wrapper 文件夹

删除*_wrapper 文件夹后， 再重新执行 pango-build -c fsbl 命