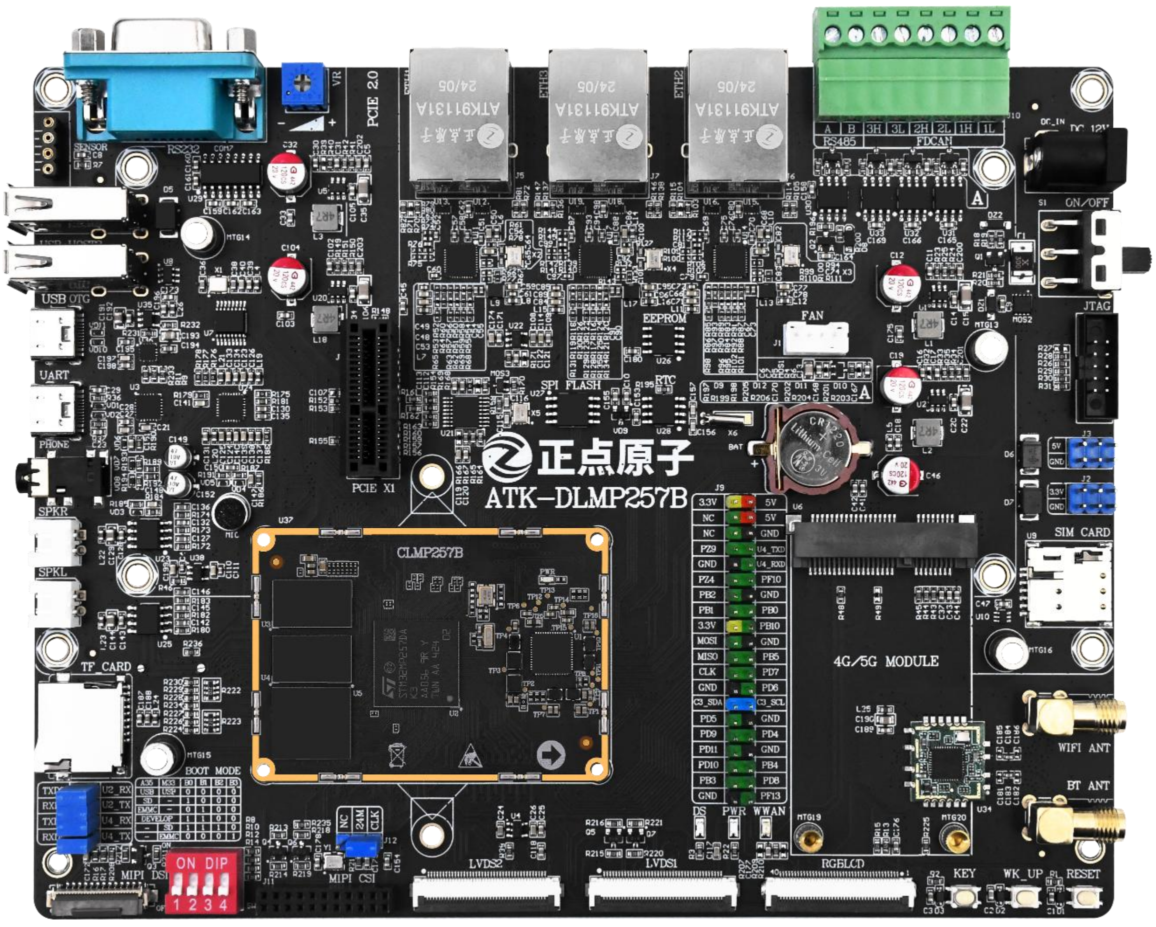
ATK-DLMP257B

Factory system source code use guide

V1.1



1. Shopping：

TMALL：https://zhengdianyuanzi.tmall.com

TAOBAO：https://openedv.taobao.com

2. Download

Address：http://www.openedv.com/docs/index.html

3. FAE

Website	：	www.alientek.com
Forum	：	http://www.openedv.com/forum.php
Videos	：	www.yuanzige.com
Fax	：	+86 - 20 - 36773971
Phone	：	+86 - 20 - 38271790

Disclaimer

Revision History:

Version	Version Update Notes	Responsible person	Proofreading	Date
V1.0	release officially	ALIENTEK	ALIENTEK	2025.04.01
V1.1	1. Fix clerical errors in Subsection 3.1	ALIENTEK	ALIENTEK	2025.4.21

Catalogue

Chapter 1. Describes the environment

This document mainly introduces the compilation process of the factory system source code of the ALIENTEK ATK-DLMP257B development board, and guides users to master the factory source code compilation method of the development board, so as to facilitate the subsequent secondary development. This document is not intended to explain how each part of compilation works.

The environment used in this document:

- Windows 10 64bits. It is not recommended to use Windows 32 bit to develop, Windows 32 bit support memory size is limited, the system performance is limited. This document is a 64-bit operating system for the introduction.
- Ubuntu24.04. Ubuntu recommends using 24.04, otherwise it may cause errors due to different installation environments.
- The reader is required to use FileZilla or WinSCP to transfer files between Ubuntu and Windows.
- This document is written for the default factory system source code of the ATK-DLMP257B development board. If the user uses other source code versions or other versions of the cross compiler, it may be due to the source code configuration, compiler support instructions and other different, resulting in compilation errors, you need to solve it by yourself, this document compilation instructions are for reference only. For the convenience of development, please use the default factory system source code.

1.1 Software Versions

Source code/tools	Version	Remark
TF-A	2.10	
OPTEE	4.0	
U-Boot	2023.10	
Linux kernel	6.6.48	
Qt	5.15	
aarch64-ostl-linux-gcc	13.3.0	Cross-compilation toolchain
STM32CubeProgrammer		ST burn tool
development environment	Ubuntu24.04	Virtual machine image provided by ALIENTE

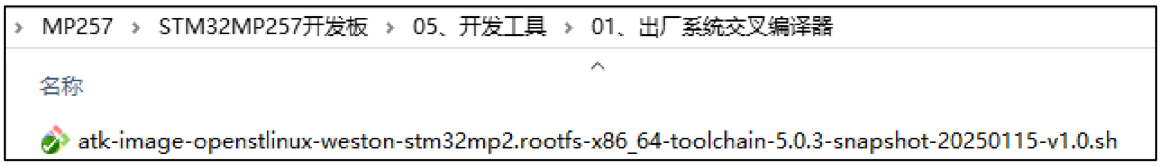
Chapter 2. Installing the ARM cross-compiler toolchain

2.1 Description of the cross-compilation toolchain

The factory system source code of ATK-DLMP257B development board needs to be compiled by ARM cross-compilation tool chain, and finally executable binary files are generated and burned to the development board. This section describes how to install the cross-compilation toolchain.

The ARM cross-compilation tool chain provided by the ATK-DLMP257B development board is located in the data path of the network disk:

[ALIENTEK] STM32MP257 Development board (disk A) - Basic information \5_ tools \1_Factory_system_cross_compiler


MP257 › STM32MP257开发板 › 05、开发工具 › 01、出厂系统交叉编译器

名称

atk-image-openstlinux-weston-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot-20250115-v1.0.sh

Figure 2.1-1 The ATK-DLMP257B cross-compilation toolchain

The atk-image-openstlinux-weston on-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot.sh toolchain is a cross-compilation toolchain generated by ALIENTEK based on the ST official yocto file system. It is mainly used to compile filesystem related commands and programs (including Qt and AI related programs). The suffix 20250115-v1.0 is the ALIENTEK version date and version number, which is only used to record the version. The general name atk-image-openstlinux-weston-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot.sh is used for demonstration throughout this document, which does not affect installation and use.

The size of the toolchain installation package is about 3GB, and the installation space is about 16GB. Please ensure that the environment storage space is sufficient before installation.

2.2 Source cross-compilation toolchain installation

Before installing, first copy it to the Ubuntu virtual machine. The ATK-DLMP257B development board is developed using Ubuntu24.04. It is recommended that users use Ubuntu24.04 in a unified way. The user environment is consistent with the author's environment, which can facilitate the use of problems in the future.

atk-image-openstlinux-weston-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot.sh is the cross-compile toolchain installation package, copy this file to the virtual machine Ubuntu24.04 system, This article has copied the cross-compiler tool to the Ubuntu virtual machine.

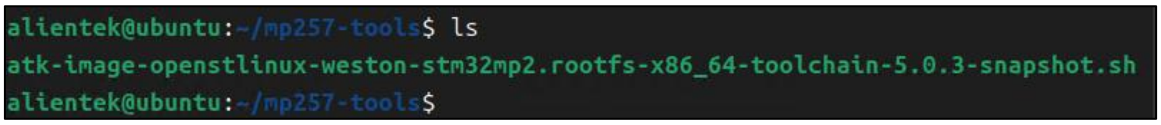

```
alientek@ubuntu:~/mp257-tools$ ls
atk-image-openstlinux-weston-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot.sh
alientek@ubuntu:~/mp257-tools$
```

Figure 2.2-1 Copy the cross-build toolchain installation package to your Ubuntu system

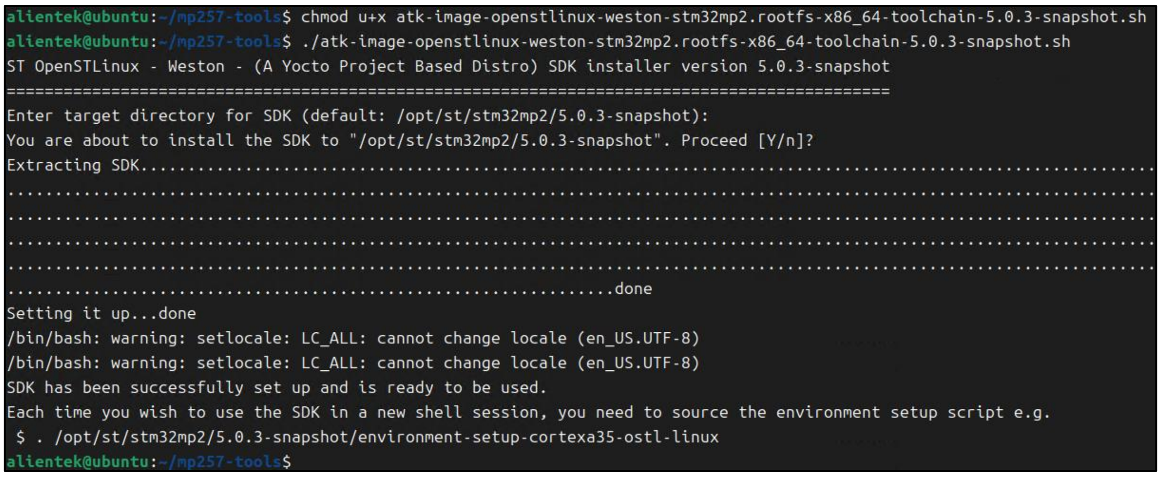
Execute the following command to change the script permissions.

```
chmod u+x atk-image-openstlinux-weston-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot.sh
```

Directly execute the script to install the cross-compiler tool, press the Enter key twice in succession to confirm, and then enter the user password. The directory of this installation is the default installation directory specified by the script, and the cross-compilation of the following kernel compilation environment is operated according to this installation directory, so it is recommended that users also install the default directory /opt/st/stm32mp2/5.0.3-snapshot.

Run the following command to install the cross-compilation toolchain, with the Enter key installed to the default path.

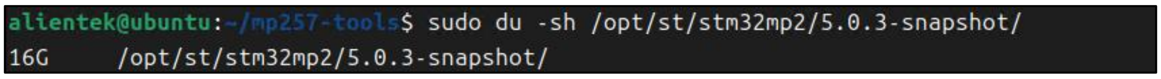
```
. / atk - image - openstlinux - weston - stm32mp2. Rootfs x86_64 - toolchain - 5.0.3 - the snapshot. Sh
```



```
alientek@ubuntu:~/mp257-tools$ chmod u+x atk-image-openstlinux-weston-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot.sh
alientek@ubuntu:~/mp257-tools$ ./atk-image-openstlinux-weston-stm32mp2.rootfs-x86_64-toolchain-5.0.3-snapshot.sh
ST OpenSTLinux - Weston - (A Yocto Project Based Distro) SDK installer version 5.0.3-snapshot
===========================================================================================
Enter target directory for SDK (default: /opt/st/stm32mp2/5.0.3-snapshot):
You are about to install the SDK to "/opt/st/stm32mp2/5.0.3-snapshot". Proceed [Y/n]?
Extracting SDK.......................................................................................................
.....................................................................................................................
.....................................................................................................................
.....................................................................................................................
.....................................................................................................................
...............................................................done
Setting it up...done
/bin/bash: warning: setlocale: LC_ALL: cannot change locale (en_US.UTF-8)
/bin/bash: warning: setlocale: LC_ALL: cannot change locale (en_US.UTF-8)
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/st/stm32mp2/5.0.3-snapshot/environment-setup-cortexa35-ostl-linux
alientek@ubuntu:~/mp257-tools$
```

Figure 2.2-2 Install the cross-compile toolchain to the default directory

The installed toolchain directory size is 16GB. This toolchain contains Qt, AI and other related libraries, so it needs enough space

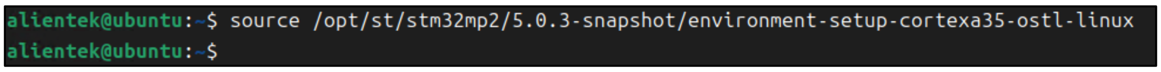


```
alientek@ubuntu:~/mp257-tools$ sudo du -sh /opt/st/stm32mp2/5.0.3-snapshot/
16G     /opt/st/stm32mp2/5.0.3-snapshot/
```

Figure 2.2-3 Total toolchain size

It's also very easy to use, following the instructions printed above, just enable the environment variables. But in different terminals or switch users need to re-enable the environment variable can be used.

```
source /opt/st/stm32mp2/5.0.3-snapshot/environment-setup-cortexa35-ostl-linux
```

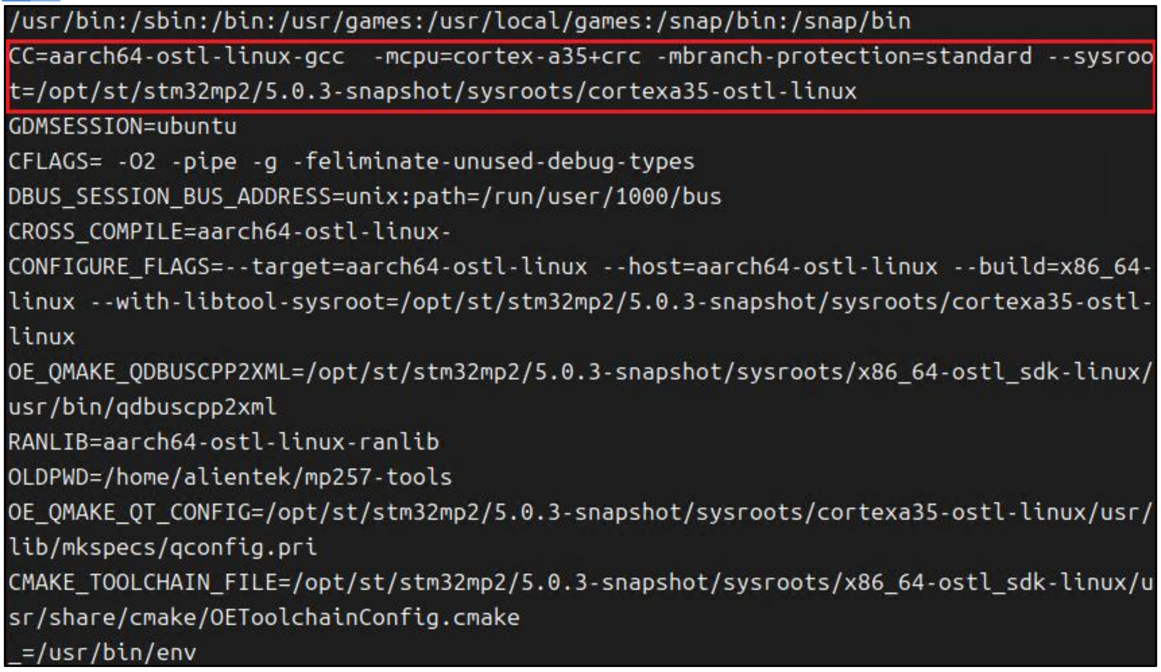


```
alientek@ubuntu:~$ source /opt/st/stm32mp2/5.0.3-snapshot/environment-setup-cortexa35-ostl-linux
alientek@ubuntu:~$
```

Figure 2.2-4 Enable environment variables

After enabling environment variables, you can use the env directive to see which environment variables are in effect. The following screenshot shows that gcc has configured the parameters for compilation with this environment variable enabled. Below is a screenshot of the environment variables section.

```
env
```



```
/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin
CC=aarch64-ostl-linux-gcc  -mcpu=cortex-a35+crc -mbranch-protection=standard --sysroo
t=/opt/st/stm32mp2/5.0.3-snapshot/sysroots/cortexa35-ostl-linux
GDMSESSION=ubuntu
CFLAGS= -O2 -pipe -g -feliminate-unused-debug-types
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
CROSS_COMPILE=aarch64-ostl-linux-
CONFIGURE_FLAGS=--target=aarch64-ostl-linux --host=aarch64-ostl-linux --build=x86_64-
linux --with-libtool-sysroot=/opt/st/stm32mp2/5.0.3-snapshot/sysroots/cortexa35-ostl-
linux
OE_QMAKE_QDBUSCPP2XML=/opt/st/stm32mp2/5.0.3-snapshot/sysroots/x86_64-ostl_sdk-linux/
usr/bin/qdbuscpp2xml
RANLIB=aarch64-ostl-linux-ranlib
OLDPWD=/home/alientek/mp257-tools
OE_QMAKE_QT_CONFIG=/opt/st/stm32mp2/5.0.3-snapshot/sysroots/cortexa35-ostl-linux/usr/
lib/mkspecs/qconfig.pri
CMAKE_TOOLCHAIN_FILE=/opt/st/stm32mp2/5.0.3-snapshot/sysroots/x86_64-ostl_sdk-linux/u
sr/share/cmake/OEToolchainConfig.cmake
_=/usr/bin/env
```

Figure 2.2-5 Look at the enabled environment variables

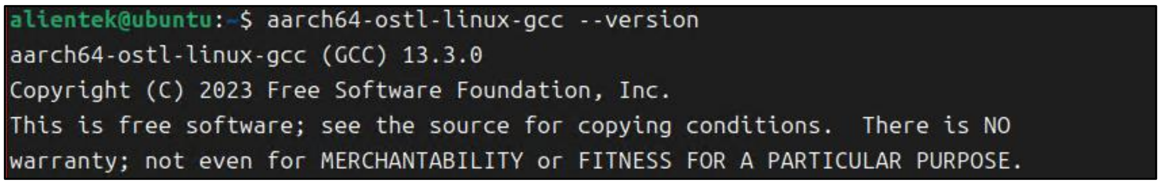
2.3 Query cross compiler

Now that you have the cross-compilation toolchain installed, you can verify that it was installed by querying the GCC version that comes with the toolchain.

Query the GCC version of the cross compiler.

```
aarch64-ostl-linux-gcc –version
```

The result is shown in the following screenshot:



```
alientek@ubuntu:~$ aarch64-ostl-linux-gcc --version
aarch64-ostl-linux-gcc (GCC) 13.3.0
Copyright (C) 2023 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.
```

Figure 2.3-1 Query cross compiler version

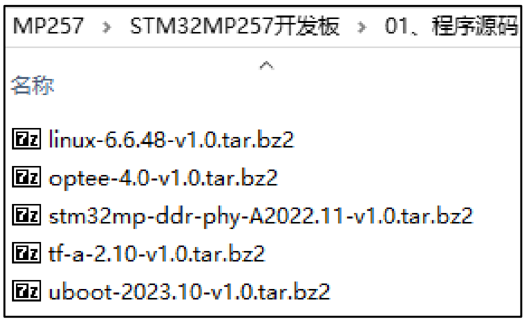
It can be seen from the above figure that the GCC version of the cross-compiler is 13.3.0, which proves that the ARM cross-compilation tool chain of the development board has been installed successfully.

Chapter 3. Factory system source code compilation and image construction

The factory system source code is transplanted and modified on the original source code version provided by ST by ALIENTEK to adapt to the hardware resources of ATK-DLMP257B development board. Users can carry out secondary development and debugging based on the factory system source code. For example, if the hardware resources of the design base board and the development board are different, they can modify a part of the factory system source code and directly compile and debug.

The factory system source code is located in the network disk data path:

[ALIENTEK] STM32MP257 development board (disk A) - Basic information \1_codes \1_ ALIENTEK_Linux_factory_system_source_code



MP257 › STM32MP257开发板 › 01、程序源码

名称

linux-6.6.48-v1.0.tar.bz2
optee-4.0-v1.0.tar.bz2
stm32mp-ddr-phy-A2022.11-v1.0.tar.bz2
tf-a-2.10-v1.0.tar.bz2
uboot-2023.10-v1.0.tar.bz2

As shown in the figure above, the factory source code will be iterated if necessary, and information such as version number will change from the example in the figure, so users can use it directly. For the convenience of driver development learning, Linux kernel source code does not do Git management, only retain a release number. In order to facilitate the explanation of this document, the v1.0 version is taken as an example to write here, and the specific source code version is subject to the latest release. Pay attention to change the following decompress source code instructions related to the source code version.

The following sections describe how the source code is compiled.

Before compiling, please create a directory named "ATK-DLMP257B" in any directory of Ubuntu environment. All compilation operations of the factory source code below will be carried out in the ATK-DLMP257B directory.

In addition, install the following tools on ubuntu24.04 to avoid the lack of libraries at compile time:

```
sudo apt-get install make gcc libssl-dev g++ git libncurses5-dev libncursesw5-dev libyaml-dev
sudo apt-get install u-boot-tools python3-pyelftools device-tree-compiler bison flex expect bzip2
```

3.1 Compile U-boot related firmware

To build the U-boot firmware, you need to install the library by entering the following:

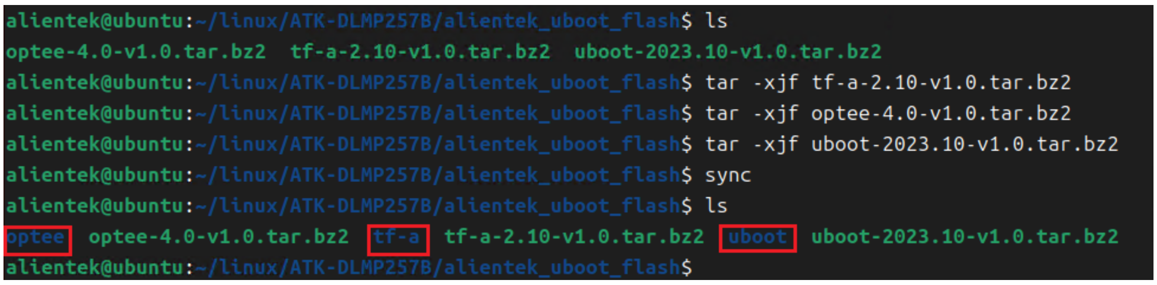
```
sudo apt-get install bison flex
```

The uboot firmware of ATK-DLMP257B development board involves four parts: tf-a, optee, ddr firmware and uboot, and the source code of tf-a, optee and uboot needs to be compiled in turn.

In Ubuntu, create a directory called "alientek_uboot_flash" in the ATK-DLMP257B directory, Then copy the positive Atom factory tf-a source code package, the ALIENTEK factory optee source code package and the ALIENTEK factory uboot source code package to the alientek_uboot_flash directory. The unzip command is as follows: (Please confirm the source code version you downloaded, do not directly copy the unzip command below)

```
tar -xjf tf-a-2.10-v1.0.tar.bz2
tar -xjf optee-4.0-v1.0.tar.bz2
tar -xjf uboot-2023.10-v1.0.tar.bz2
sync
```

Make sure that the tf-a, optee, ddr firmware, and uboot folders are in the same directory, because when compiling the firmware, the related file paths will be linked, and the wrong path may cause the compilation failure:



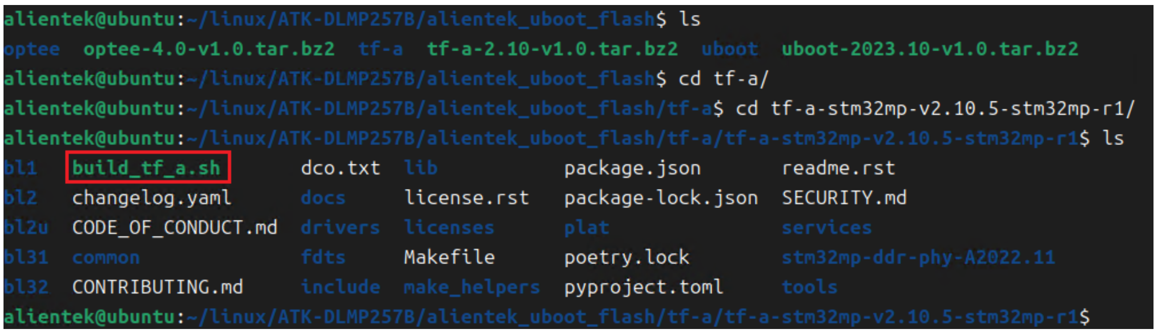
```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ ls
optee-4.0-v1.0.tar.bz2  tf-a-2.10-v1.0.tar.bz2  uboot-2023.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ tar -xjf tf-a-2.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ tar -xjf optee-4.0-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ tar -xjf uboot-2023.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ sync
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ ls
optee  optee-4.0-v1.0.tar.bz2  tf-a  tf-a-2.10-v1.0.tar.bz2  uboot  uboot-2023.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$
```

Figure 3.1-1 The four folders tf-a, optee, ddr firmware, and uboot are in the same directory

The ATK-DLMP257B development board \ core board has two versions of 1GB DDR and 2GB DDR. The configuration of different versions is not the same, and users need to compile according to the development board \ core board they use. Because there are many steps to compile UBoot-related firmware, tf-a, optee and uboot need to be compiled in turn. Users need to compile according to the only version they use in the compilation process, and the source code \ firmware of 1GB and 2GB versions can not be used interactively.

3.1.1 Compiling tf-a

When the factory source code is compiled, the user does not have to modify anything. To facilitate your compilation, ALIENTEK provides a key compilation script, the first time you can directly run the script compilation, tf-a compilation script build_tf_a.sh path is as follows:



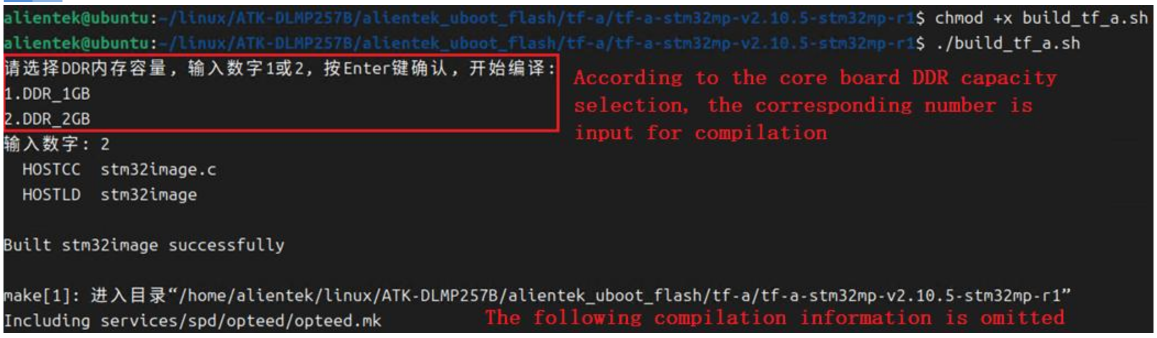
```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ ls
optee  optee-4.0-v1.0.tar.bz2  tf-a  tf-a-2.10-v1.0.tar.bz2  uboot  uboot-2023.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ cd tf-a/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a$ cd tf-a-stm32mp-v2.10.5-stm32mp-r1/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1$ ls
bl1   build_tf_a.sh       dco.txt  lib           package.json       readme.rst
bl2   changelog.yaml      docs     license.rst   package-lock.json  SECURITY.md
bl2u  CODE_OF_CONDUCT.md  drivers  licenses      plat               services
bl31  common              fdts     Makefile      poetry.lock        stm32mp-ddr-phy-A2022.11
bl32  CONTRIBUTING.md     include  make_helpers  pyproject.toml     tools
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1$
```

Figure 3.1.1-1 The build_tf_a.sh script

Give the script permissions and execute the script:

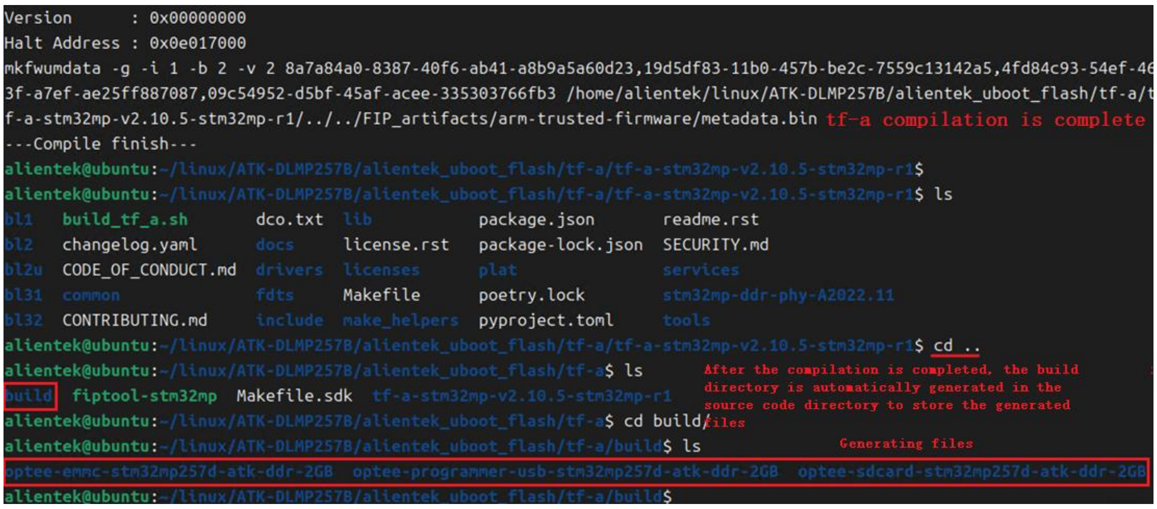
```
chmod u+x build_tf_a.sh
./build_tf_a.sh
```

The ATK-DLMP257B development board \ core board has two versions of 1GB DDR and 2GB DDR. The configuration of different versions is not the same, and users need to compile according to the development board \ core board they use. For example, the 2GB version:


According to the core board DDR capacity selection, the corresponding number is input for compilation
The following compilation information is omitted

```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1$ chmod +x build_tf_a.sh
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1$ ./build_tf_a.sh
请选择DDR内存容量，输入数字1或2，按Enter键确认，开始编译：
1.DDR_1GB
2.DDR_2GB
输入数字：2
  HOSTCC  stm32image.c
  HOSTLD  stm32image

Built stm32image successfully

make[1]: 进入目录“/home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1”
Including services/spd/opteed/opteed.mk
```

Figure 3.1.1-2 Compile tf-a


tf-a compilation is complete
After the compilation is completed, the build directory is automatically generated in the source code directory to store the generated files
Generating files

```
Version      : 0x00000000
Halt Address : 0x0e017000
mkfwumdata -g -i 1 -b 2 -v 2 8a7a84a0-8387-40f6-ab41-a8b9a5a60d23,19d5df83-11b0-457b-be2c-7559c13142a5,4fd84c93-54ef-46
3f-a7ef-ae25ff887087,09c54952-d5bf-45af-acee-335303766fb3 /home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/t
f-a-stm32mp-v2.10.5-stm32mp-r1/../../FIP_artifacts/arm-trusted-firmware/metadata.bin
---Compile finish---
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1$
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1$ ls
bl1   build_tf_a.sh       dco.txt  lib           package.json       readme.rst
bl2   changelog.yaml      docs     license.rst   package-lock.json  SECURITY.md
bl2u  CODE_OF_CONDUCT.md  drivers  licenses      plat               services
bl31  common              fdts     Makefile      poetry.lock        stm32mp-ddr-phy-A2022.11
bl32  CONTRIBUTING.md     include  make_helpers  pyproject.toml     tools
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/tf-a-stm32mp-v2.10.5-stm32mp-r1$ cd ..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a$ ls
build  fiptool-stm32mp  Makefile.sdk  tf-a-stm32mp-v2.10.5-stm32mp-r1
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a$ cd build/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/build$ ls
optee-emmc-stm32mp257d-atk-ddr-2GB  optee-programmer-usb-stm32mp257d-atk-ddr-2GB  optee-sdcard-stm32mp257d-atk-ddr-2GB
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/build$
```

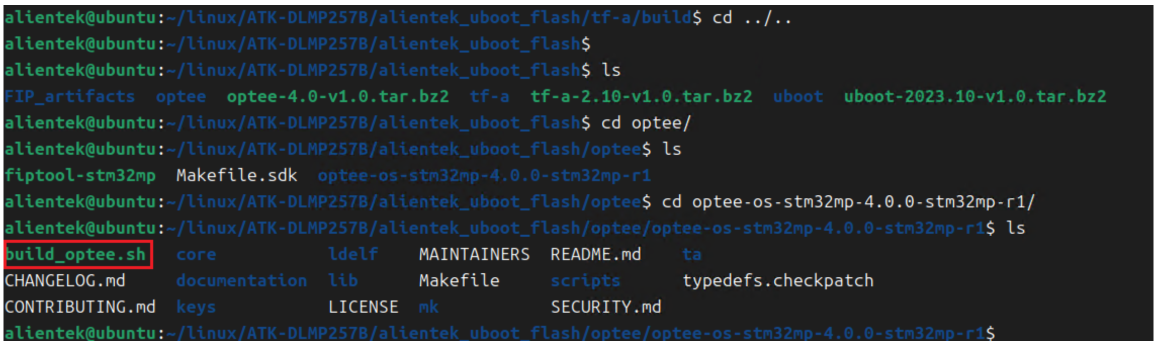
Figure 3.1.1-3 tf-a compilation is complete

After executing the build_tf_a.sh script and compiling, a build directory will be generated in the upper directory to store the files generated by compilation, including the files required by optee compilation.

3.1.2 Compiling optee

When the factory source code is compiled, the user does not have to modify anything. In order to facilitate your compilation, ALIENTEK provides a key compilation script, the first time you can directly run the script compilation, optee compilation script build_optee.sh path is as follows:

```
cd ../..
cd optee
cd optee-os-stm32mp-4.0.0-stm32mp-r1
```

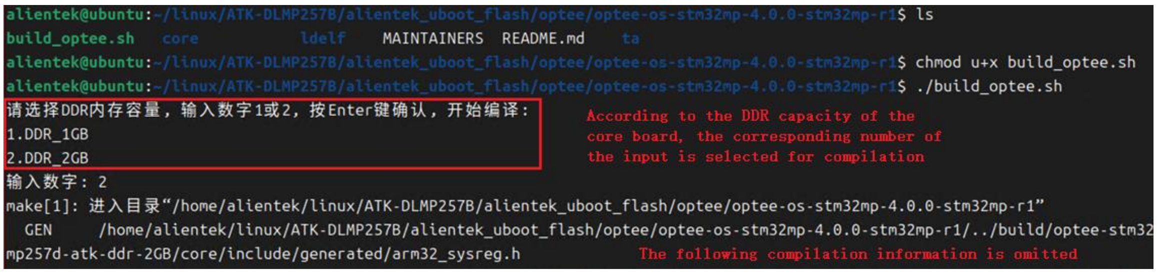


```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/tf-a/build$ cd ../..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ ls
FIP_artifacts  optee  optee-4.0-v1.0.tar.bz2  tf-a  tf-a-2.10-v1.0.tar.bz2  uboot  uboot-2023.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ cd optee/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee$ ls
fiptool-stm32mp  Makefile.sdk  optee-os-stm32mp-4.0.0-stm32mp-r1
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee$ cd optee-os-stm32mp-4.0.0-stm32mp-r1/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$ ls
build_optee.sh   core           ldelf     MAINTAINERS  README.md    ta
CHANGELOG.md     documentation  lib       Makefile     scripts      typedefs.checkpatch
CONTRIBUTING.md  keys           LICENSE   mk           SECURITY.md
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$
```

Figure 3.1.2-1 The build_optee.sh script

After compiling tf-a, optee can be compiled. After giving build_optee.sh executable permission, execute the build_optee.sh script and select the corresponding DDR version for compilation. The author used the 2GB configuration before, but also need to choose the 2GB configuration here, and can not choose the 1GB compilation configuration.

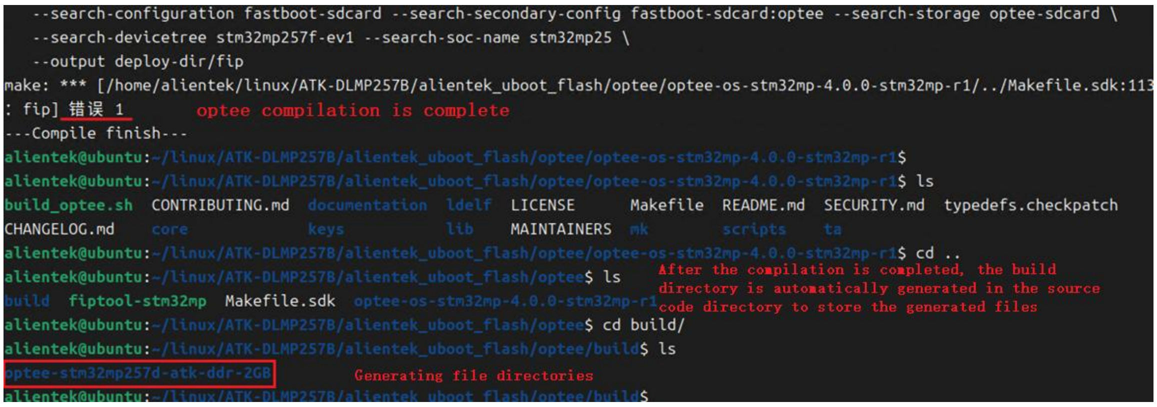
```
chmod u+x build_optee.sh
./build_optee.sh
```



```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$ ls
build_optee.sh   core           ldelf     MAINTAINERS  README.md    ta
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$ chmod u+x build_optee.sh
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$ ./build_optee.sh
请选择DDR内存容量，输入数字1或2，按Enter键确认，开始编译：
1.DDR_1GB
2.DDR_2GB
输入数字：2
make[1]: 进入目录“/home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1”
  GEN     /home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1/../build/optee-stm32
mp257d-atk-ddr-2GB/core/include/generated/arm32_sysreg.h
```

According to the DDR capacity of the core board, the corresponding number of the input is selected for compilation
The following compilation information is omitted

Figure 3.1.2-2 Compiling optee

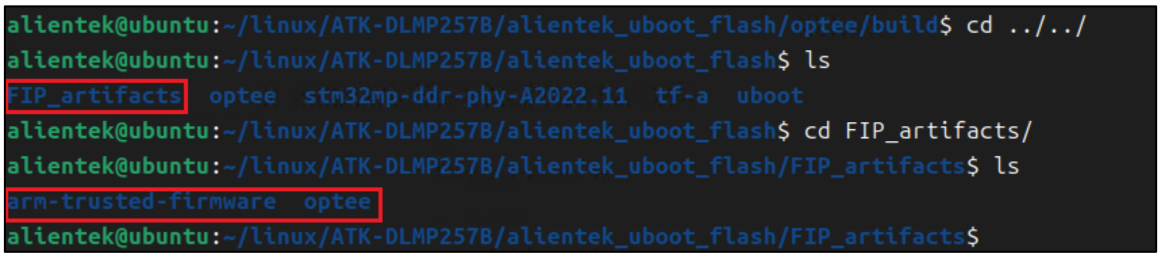


```
   --search-configuration fastboot-sdcard --search-secondary-config fastboot-sdcard:optee --search-storage optee-sdcard \
   --search-devicetree stm32mp257f-ev1 --search-soc-name stm32mp25 \
   --output deploy-dir/fip
make: *** [/home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1/../Makefile.sdk:113
: fip] 错误 1
---Compile finish---
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$ ls
build_optee.sh  CONTRIBUTING.md  documentation  ldelf  LICENSE      Makefile  README.md  SECURITY.md  typedefs.checkpatch
CHANGELOG.md    core             keys           lib    MAINTAINERS  mk        scripts    ta
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/optee-os-stm32mp-4.0.0-stm32mp-r1$ cd ..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee$ ls
build  fiptool-stm32mp  Makefile.sdk  optee-os-stm32mp-4.0.0-stm32mp-r1
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee$ cd build/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/build$ ls
optee-stm32mp257d-atk-ddr-2GB
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/build$
```

optee compilation is complete
After the compilation is completed, the build directory is automatically generated in the source code directory to store the generated files
Generating file directories

Figure 3.1.2-3 optee compilation is complete

After executing the build_optee.sh script and compiling, it will generate a build directory in the upper directory to store the generated files. After compiling, you can see the fip error message, which does not affect the compilation of optee. After compiling uboot, you will not get an error when compiling optee. After compiling, you can see the FIP_artifacts directory generated in the upper directory of the optee source code, which stores the arm-trusted-firmware and optee related files.

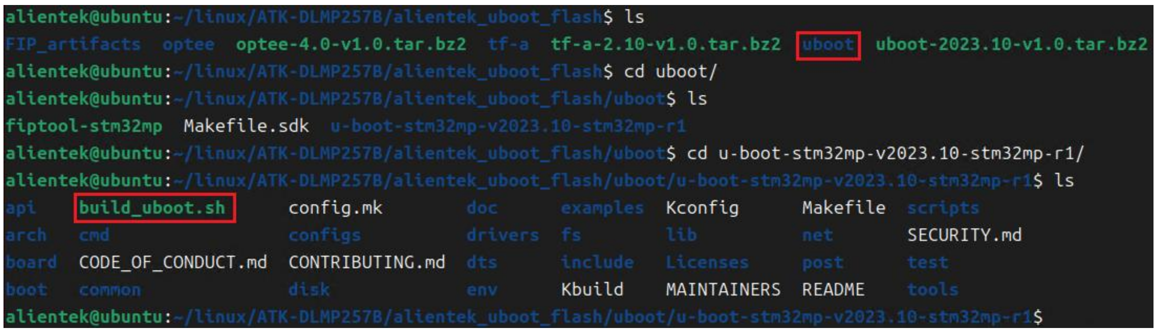


```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/optee/build$ cd ../../
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ ls
FIP_artifacts  optee  stm32mp-ddr-phy-A2022.11  tf-a  uboot
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ cd FIP_artifacts/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$ ls
arm-trusted-firmware  optee
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$
```

Figure 3.1.2-4 The FIP_artifacts directory is generated when compilation is complete

3.1.3 Compiling uboot

When the factory source code is compiled, the user does not have to modify anything. To facilitate your compilation, ALIENTEK provides a key compilation script, you can directly run the script compilation when compiling for the first time, uboot compilation script build_uboot.sh path is as follows:

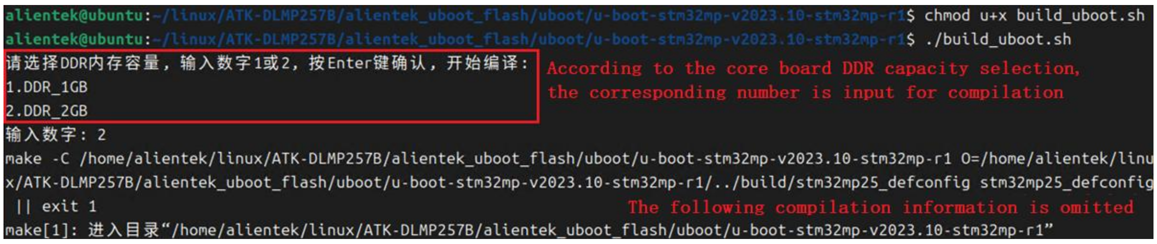


```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ ls
FIP_artifacts  optee  optee-4.0-v1.0.tar.bz2  tf-a  tf-a-2.10-v1.0.tar.bz2  uboot  uboot-2023.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ cd uboot/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot$ ls
fiptool-stm32mp  Makefile.sdk  u-boot-stm32mp-v2023.10-stm32mp-r1
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot$ cd u-boot-stm32mp-v2023.10-stm32mp-r1/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1$ ls
api    build_uboot.sh      config.mk        doc      examples  Kconfig      Makefile  scripts
arch   cmd                 configs          drivers  fs        lib          net       SECURITY.md
board  CODE_OF_CONDUCT.md  CONTRIBUTING.md  dts      include   Licenses     post      test
boot   common              disk             env      Kbuild    MAINTAINERS  README    tools
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1$
```

Figure 3.1.3-1 build_uboot.sh script

After compiling tf-a and optee, compile uboot. After giving build_uboot.sh executable permission, execute the build_uboot.sh script and select the corresponding DDR version for compilation. The author used the 2GB configuration before, and also need to choose the 2GB configuration here, and can not choose the 1GB compilation configuration.

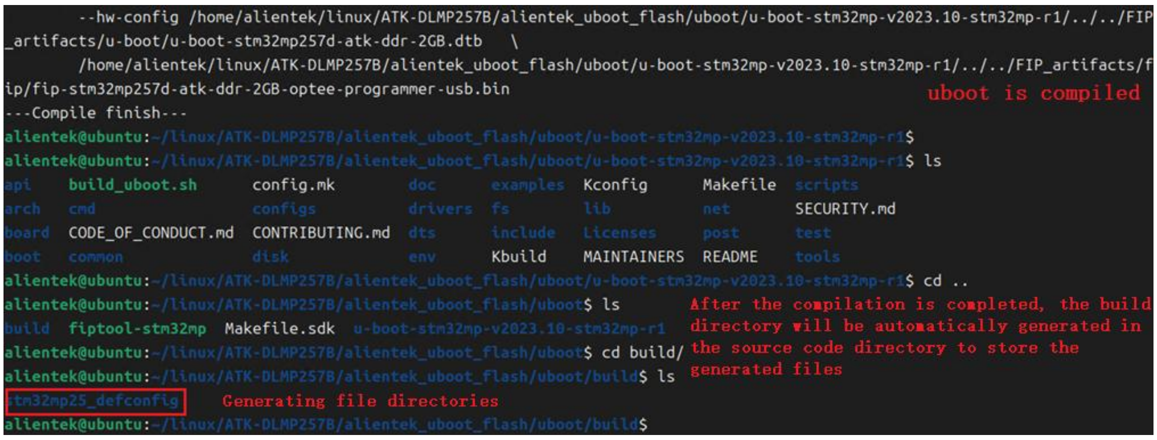
```
chmod u+x build_uboot.sh
./build_uboot.sh
```



```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1$ chmod u+x build_uboot.sh
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1$ ./build_uboot.sh
请选择DDR内存容量，输入数字1或2，按Enter键确认，开始编译：
1.DDR_1GB
2.DDR_2GB
输入数字：2
make -C /home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1 O=/home/alientek/linu
x/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1/../build/stm32mp25_defconfig stm32mp25_defconfig
 || exit 1
make[1]: 进入目录"/home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1"
```

According to the core board DDR capacity selection, the corresponding number is input for compilation
The following compilation information is omitted

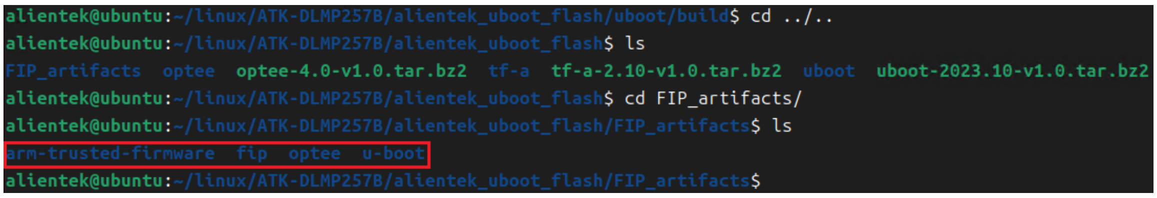
Figure 3.1.3-2 Compiling uboot



```
        --hw-config /home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1/../../FIP
_artifacts/u-boot/u-boot-stm32mp257d-atk-ddr-2GB.dtb    \
        /home/alientek/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1/../../FIP_artifacts/f
ip/fip-stm32mp257d-atk-ddr-2GB-optee-programmer-usb.bin                                         uboot is compiled
---Compile finish---
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1$
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1$ ls
api    build_uboot.sh      config.mk          doc       examples  Kconfig      Makefile  scripts
arch   cmd                 configs            drivers   fs        lib          net       SECURITY.md
board  CODE_OF_CONDUCT.md  CONTRIBUTING.md    dts       include   Licenses     post      test
boot   common              disk               env       Kbuild    MAINTAINERS  README    tools
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/u-boot-stm32mp-v2023.10-stm32mp-r1$ cd ..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot$ ls        After the compilation is completed, the build
build  fiptool-stm32mp  Makefile.sdk  u-boot-stm32mp-v2023.10-stm32mp-r1    directory will be automatically generated in
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot$ cd build/ the source code directory to store the
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/build$ ls   generated files
stm32mp25_defconfig     Generating file directories
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/build$
```

Figure 3.1.3-3 uboot is compiled

After executing the build_uboot.sh script and compiling, it will generate a build directory in the upper directory to store the generated files. After the compilation is complete, the required uboot files will be automatically copied to the FIP_artifacts directory in the uboot source directory, with the fip and uboot directories added.

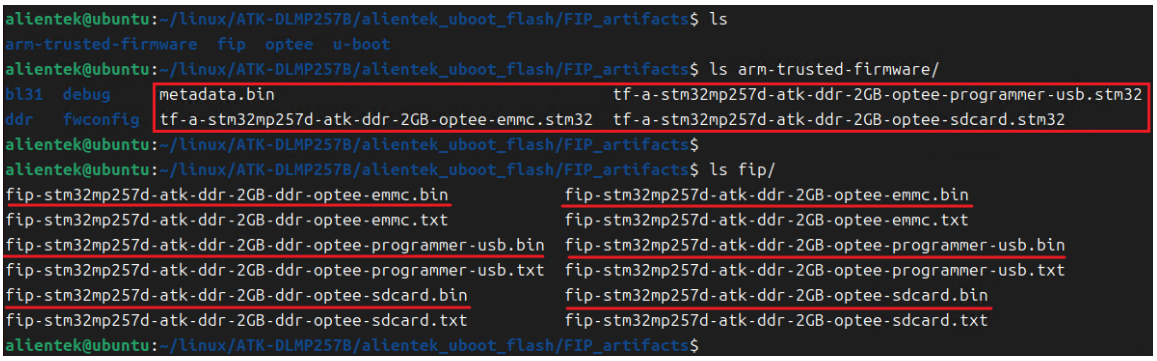


```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/uboot/build$ cd ../..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ ls
FIP_artifacts  optee  optee-4.0-v1.0.tar.bz2  tf-a  tf-a-2.10-v1.0.tar.bz2  uboot  uboot-2023.10-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash$ cd FIP_artifacts/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$ ls
arm-trusted-firmware  fip  optee  u-boot
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$
```

Figure 3.1.3-4 FIP_artifacts directory

3.1.4 The FIP_artifacts directory

After compiling tf-a, optee, and uboot in turn, the arm-trusted-firmware, fip, optee, and u-boot files are generated in the FIP_artifacts directory. The arm-trusted-firmware and fip are the files we can burn to the ATK-DLMP257B development board, and the content is as follows:



```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$ ls
arm-trusted-firmware  fip  optee  u-boot
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$ ls arm-trusted-firmware/
bl31  debug    metadata.bin                                 tf-a-stm32mp257d-atk-ddr-2GB-optee-programmer-usb.stm32
ddr   fwconfig tf-a-stm32mp257d-atk-ddr-2GB-optee-emmc.stm32  tf-a-stm32mp257d-atk-ddr-2GB-optee-sdcard.stm32
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$ ls fip/
fip-stm32mp257d-atk-ddr-2GB-ddr-optee-emmc.bin              fip-stm32mp257d-atk-ddr-2GB-optee-emmc.bin
fip-stm32mp257d-atk-ddr-2GB-ddr-optee-emmc.txt              fip-stm32mp257d-atk-ddr-2GB-optee-emmc.txt
fip-stm32mp257d-atk-ddr-2GB-ddr-optee-programmer-usb.bin    fip-stm32mp257d-atk-ddr-2GB-optee-programmer-usb.bin
fip-stm32mp257d-atk-ddr-2GB-ddr-optee-programmer-usb.txt    fip-stm32mp257d-atk-ddr-2GB-optee-programmer-usb.txt
fip-stm32mp257d-atk-ddr-2GB-ddr-optee-sdcard.bin            fip-stm32mp257d-atk-ddr-2GB-optee-sdcard.bin
fip-stm32mp257d-atk-ddr-2GB-ddr-optee-sdcard.txt            fip-stm32mp257d-atk-ddr-2GB-optee-sdcard.txt
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_uboot_flash/FIP_artifacts$
```

Figure 3.1-2 arm-trusted-firmware and fip files

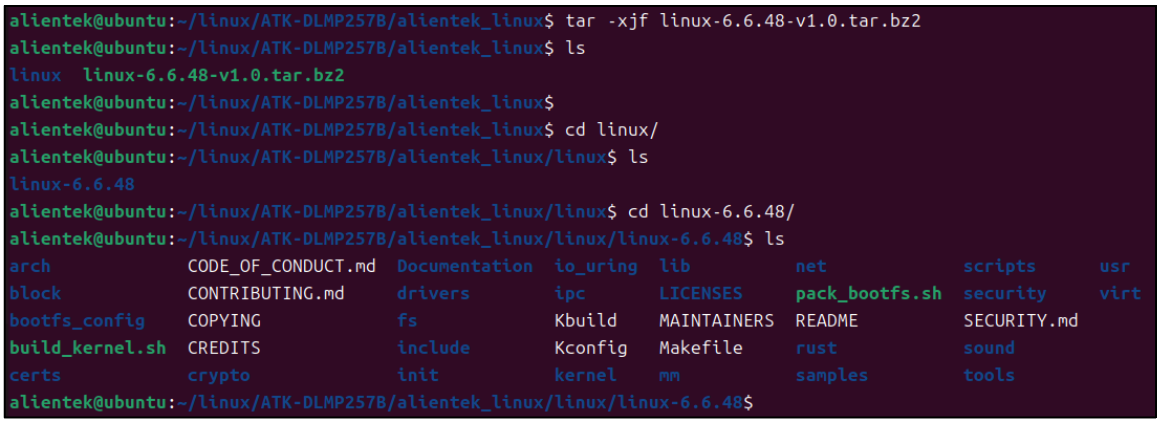
This is the 2GB build because I'm using the 2GB build. If you're using the 1GB build configuration, this is the 1GB build. The files in the arm-trusted-firmware and fip directories correspond to 8_system_image \01, factory system image \01 in the network disk data of the development board, and the arm-trusted-firmware and fip directories of the firmware package burned by STM32CubeProg. You can directly replace these two directories and use STM32CubeProgrammer for burning.

3.2 Compile factory kernel sources and modules

Create a directory called "alientek_linux" in the Ubuntu ATK-DLMP257B directory, and copy the factory linux source package of ALIENTEK to this directory. Execute the unzip command and navigate to the kernel source directory. (Please confirm the source code version you downloaded, do not directly copy the unzip command below)

```
tar-xjf linux-6.6.48-v1.0.tar.bz2 // Add the -v argument to see the unpacking details
sync
cd linux/linux-6.6.48
```

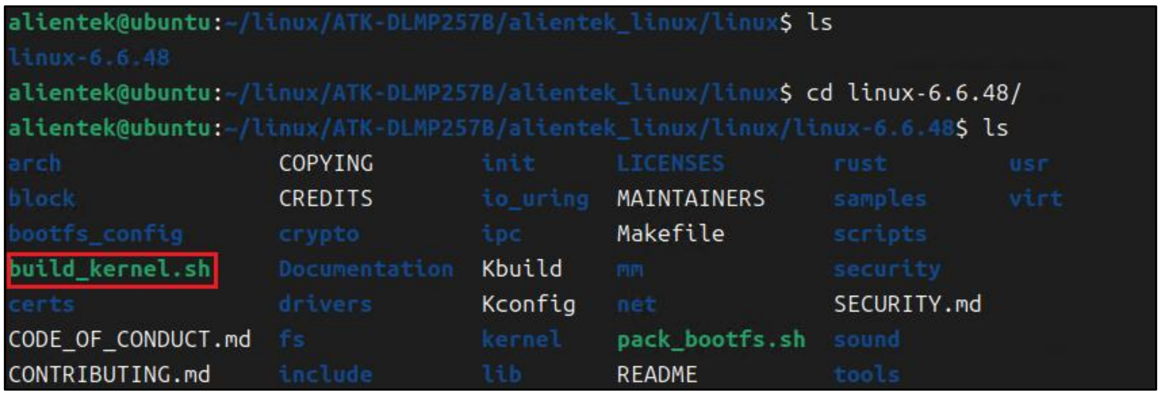
The result is shown in the following screenshot:



```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux$ tar -xjf linux-6.6.48-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux$ ls
linux  linux-6.6.48-v1.0.tar.bz2
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux$
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux$ cd linux/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ ls
linux-6.6.48
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ cd linux-6.6.48/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ ls
arch              CODE_OF_CONDUCT.md  Documentation  io_uring  lib          net             scripts      usr
block             CONTRIBUTING.md     drivers        ipc       LICENSES     pack_bootfs.sh  security     virt
bootfs_config     COPYING             fs             Kbuild    MAINTAINERS  README          SECURITY.md
build_kernel.sh   CREDITS             include        Kconfig   Makefile     rust            sound
certs             crypto              init           kernel    mm           samples         tools
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$
```

Figure 3.2-1 Unzip the kernel sources

When the factory source code is compiled, the user does not have to modify anything. To facilitate your compilation, ALIENTEK provides a one-button compilation script build_kernel.sh, you can directly run the script to compile the first time.



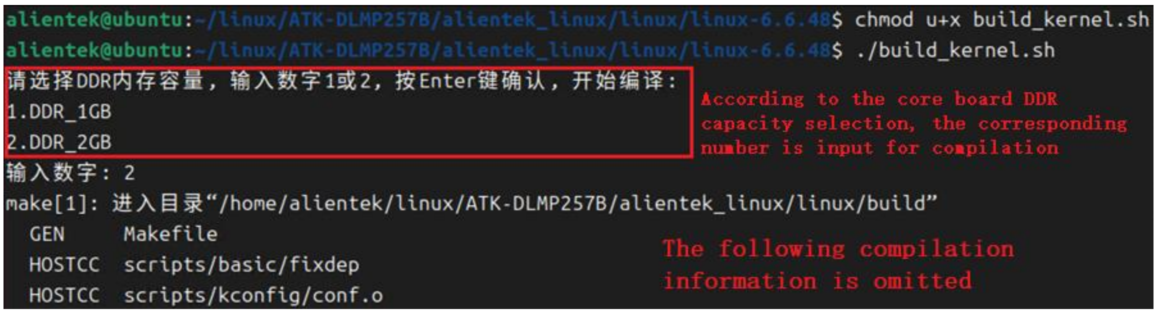
```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ ls
linux-6.6.48
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ cd linux-6.6.48/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ ls
arch                COPYING        init      LICENSES        rust         usr
block               CREDITS        io_uring  MAINTAINERS     samples      virt
bootfs_config       crypto         ipc       Makefile        scripts
build_kernel.sh     Documentation  Kbuild    mm              security
certs               drivers        Kconfig   net             SECURITY.md
CODE_OF_CONDUCT.md  fs             kernel    pack_bootfs.sh  sound
CONTRIBUTING.md     include        lib       README          tools
```

Figure 3.2-2 build_kernel.sh script

After giving the script executable permission, run the compile command:

```
chmod u+x build_kernel.sh
./build_kernel.sh
```

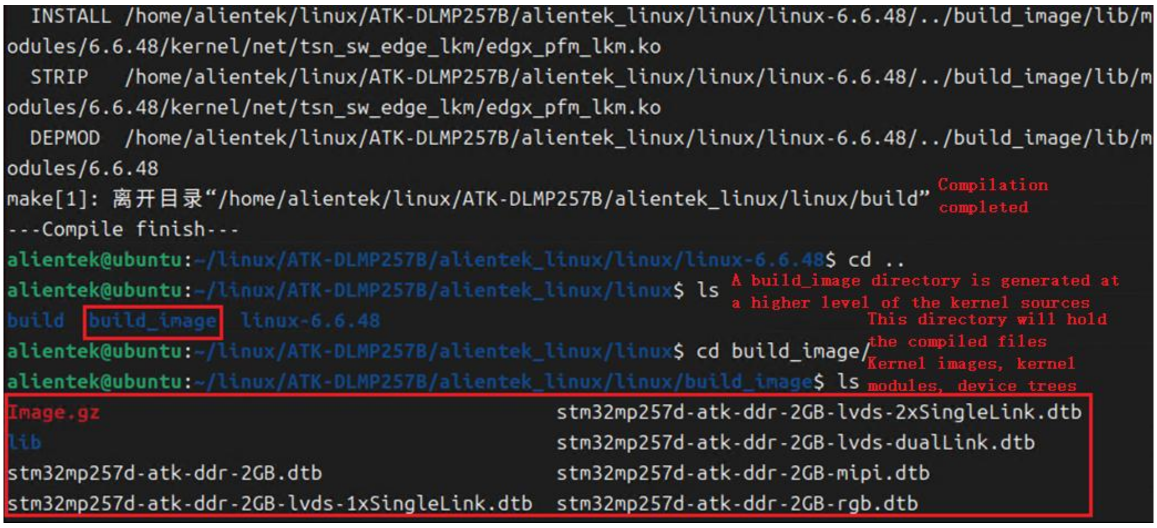
The ATK-DLMP257B development board \ core board has two versions of 1GB DDR and 2GB DDR. The configuration of different versions is not the same, and users need to compile according to the development board \ core board they use. For example, the 2GB version:



```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ chmod u+x build_kernel.sh
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ ./build_kernel.sh
请选择DDR内存容量，输入数字1或2，按Enter键确认，开始编译：
1.DDR_1GB
2.DDR_2GB
输入数字：2
make[1]: 进入目录“/home/alientek/linux/ATK-DLMP257B/alientek_linux/linux/build”
  GEN     Makefile
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/conf.o
```

According to the core board DDR capacity selection, the corresponding number is input for compilation
The following compilation information is omitted

Figure 3.2-3 Compile the kernel sources



```
  INSTALL /home/alientek/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48/../build_image/lib/m
odules/6.6.48/kernel/net/tsn_sw_edge_lkm/edgx_pfm_lkm.ko
  STRIP   /home/alientek/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48/../build_image/lib/m
odules/6.6.48/kernel/net/tsn_sw_edge_lkm/edgx_pfm_lkm.ko
  DEPMOD  /home/alientek/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48/../build_image/lib/m
odules/6.6.48
make[1]: 离开目录“/home/alientek/linux/ATK-DLMP257B/alientek_linux/linux/build”
---Compile finish---
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ cd ..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ ls
build  build_image  linux-6.6.48
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ cd build_image/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image$ ls
Image.gz                                       stm32mp257d-atk-ddr-2GB-lvds-2xSingleLink.dtb
lib                                            stm32mp257d-atk-ddr-2GB-lvds-dualLink.dtb
stm32mp257d-atk-ddr-2GB.dtb                    stm32mp257d-atk-ddr-2GB-mipi.dtb
stm32mp257d-atk-ddr-2GB-lvds-1xSingleLink.dtb  stm32mp257d-atk-ddr-2GB-rgb.dtb
```

Compilation completed
A build_image directory is generated at a higher level of the kernel sources
This directory will hold the compiled files Kernel images, kernel modules, device trees

Figure 3.2-4 Compilation completed

After compilation, a build_image directory is generated in the upper directory of the kernel sources to store the generated kernel Image file image.gz, the kernel module package lib, and the device tree file dtb. You can also see a build directory, which is used to hold the intermediate files generated during compilation.

Looking at the files in the build_image directory, taking the 2GB version as an example, the compiled kernel image.gz Image and device tree file are as follows:

Image name	Mirror image effect	Image file and path
kernel	Linux image	Image.gz
Device tree	Development board basic peripheral functions, such as Ethernet, TF card, etc	stm32mp257d-atk-ddr-2GB.dtb
	Support positive ALIENTEK MIPI screen	stm32mp257d-atk-ddr-2GB-mipi.dtb
	Support positive ALIENTEK RGB screen	stm32mp257d-atk-ddr-2GB-rgb.dtb
	Support positive ALIENTEK LVDS screen (Interface 1)	stm32mp257d-atk-ddr-2GB-lvds-1xSingleLink.dtb
	Support positive ALIENTEK LVDS screen (Interface 2)	stm32mp257d-atk-ddr-2GB-lvds-2xSingleLink.dtb

	Supports dual LVDS screens	stm32mp257d-atk-ddr-2GB-lvds-dualLink.dtb
Kernel module	Modules that need to be loaded when the kernel is started are stored in this directory	lib

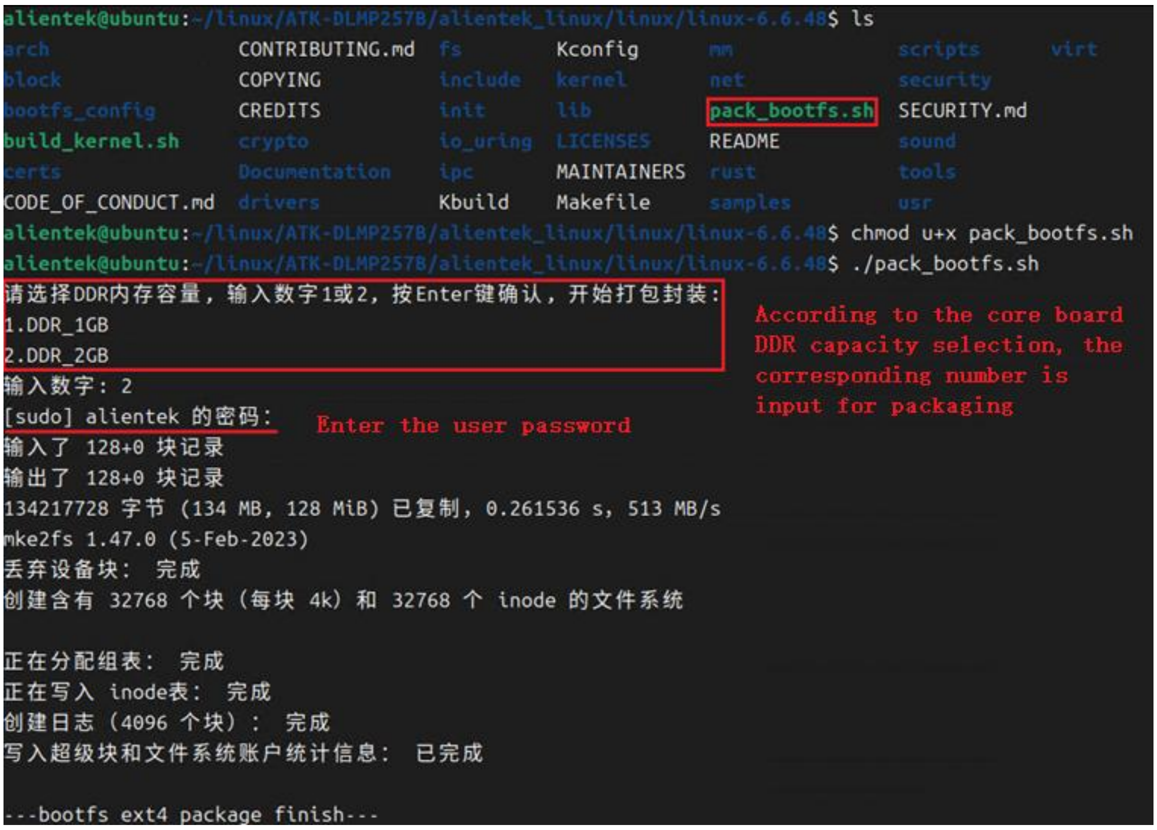
To verify the modified function, the compiled file can be copied to the corresponding directory on the development board system. The Image.gz and dtb files are located in the /boot directory of the factory system of the development board.

For more firmware update operations, please refer to [ALIENTEK] STM32MP257 development board (disk A) - Basic Information \10_user_manual \ [ALIENTEK] ATK-DLMP257B Firmware Update Reference Documentation V1.0.pdf

The 2GB version, for example, needs to be packaged into a bootfs-2GB.ext4 image if you want to use the host machine. There is also a pack_bootfs.sh script reserved in the factory kernel sources to package the image. After executing the build_kernel.sh script, the pack_bootfs.sh script can be used to generate the corresponding bootfs image.

```
Chmod u+x pack_bootfs.sh
./pack_bootfs.sh
```

To run this script, you need to use the sudo permission. Enter the user password.

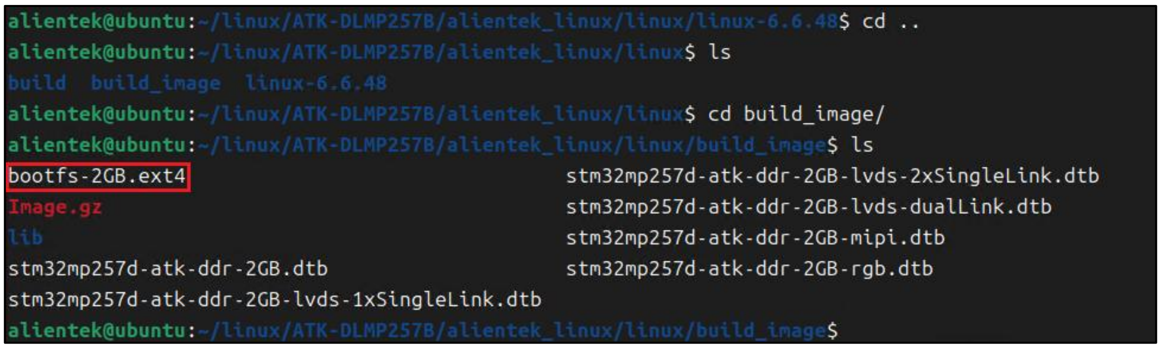


```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ ls
arch               CONTRIBUTING.md  fs        Kconfig      mm              scripts       virt
block              COPYING          include   kernel       net             security
bootfs_config      CREDITS          init      lib          pack_bootfs.sh  SECURITY.md
build_kernel.sh    crypto           io_uring  LICENSES     README          sound
certs              Documentation    ipc       MAINTAINERS  rust            tools
CODE_OF_CONDUCT.md drivers          Kbuild    Makefile     samples         usr
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ chmod u+x pack_bootfs.sh
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ ./pack_bootfs.sh
请选择DDR内存容量，输入数字1或2，按Enter键确认，开始打包封装:
1.DDR_1GB
2.DDR_2GB
输入数字: 2
[sudo] alientek 的密码:
输入了 128+0 块记录
输出了 128+0 块记录
134217728 字节 (134 MB, 128 MiB) 已复制, 0.261536 s, 513 MB/s
mke2fs 1.47.0 (5-Feb-2023)
丢弃设备块:  完成
创建含有 32768 个块 (每块 4k) 和 32768 个 inode 的文件系统

正在分配组表:  完成
正在写入 inode表:  完成
创建日志 (4096 个块) :  完成
写入超级块和文件系统账户统计信息:  已完成

---bootfs ext4 package finish---
```

According to the core board DDR capacity selection, the corresponding number is input for packaging
Enter the user password

Figure 3.2-5 Execute the pack_bootfs.sh script

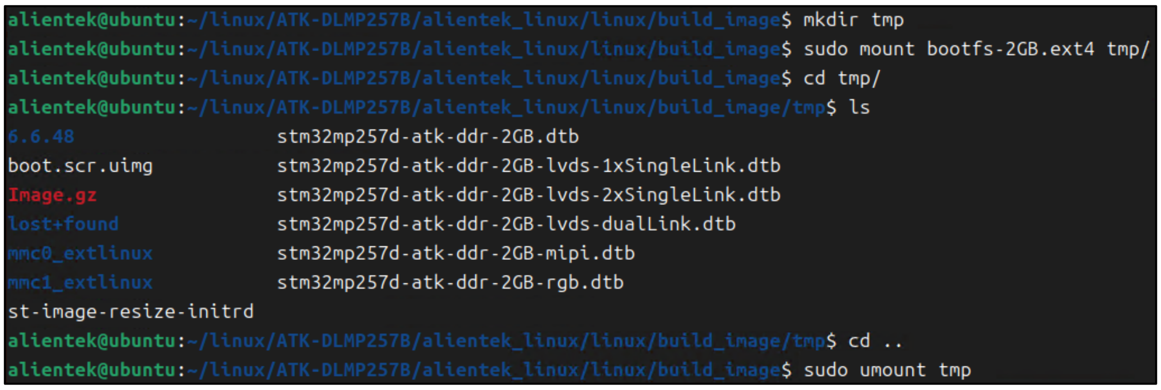
Once the package is complete, a bootfs image is generated under the build_image directory in the parent directory.



```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/linux-6.6.48$ cd ..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ ls
build  build_image  linux-6.6.48
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux$ cd build_image/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image$ ls
bootfs-2GB.ext4                                 stm32mp257d-atk-ddr-2GB-lvds-2xSingleLink.dtb
Image.gz                                        stm32mp257d-atk-ddr-2GB-lvds-dualLink.dtb
lib                                             stm32mp257d-atk-ddr-2GB-mipi.dtb
stm32mp257d-atk-ddr-2GB.dtb                     stm32mp257d-atk-ddr-2GB-rgb.dtb
stm32mp257d-atk-ddr-2GB-lvds-1xSingleLink.dtb
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image$
```

Figure 3.2-6 The corresponding bootfs image is generated under build_image

This Image is a bootfs file that contains the kernel image image.gz, kernel module lib, device tree dtb file, and added boot related files such as mmc_extlinux, boot.scr.uimg, st-image-resize-initrd. It can be used for STM32CubeProgrammer host computer burning or mass production burning.



```
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image$ mkdir tmp
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image$ sudo mount bootfs-2GB.ext4 tmp/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image$ cd tmp/
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image/tmp$ ls
6.6.48                  stm32mp257d-atk-ddr-2GB.dtb
boot.scr.uimg           stm32mp257d-atk-ddr-2GB-lvds-1xSingleLink.dtb
Image.gz                stm32mp257d-atk-ddr-2GB-lvds-2xSingleLink.dtb
lost+found              stm32mp257d-atk-ddr-2GB-lvds-dualLink.dtb
mmc0_extlinux           stm32mp257d-atk-ddr-2GB-mipi.dtb
mmc1_extlinux           stm32mp257d-atk-ddr-2GB-rgb.dtb
st-image-resize-initrd
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image/tmp$ cd ..
alientek@ubuntu:~/linux/ATK-DLMP257B/alientek_linux/linux/build_image$ sudo umount tmp
```

Figure 3.2-7 bootfs contents

3.3 Compile the factory QT GUI comprehensive interface

Please refer to the development board network disk information \10_user_manual \ [ALIENTEK] ATK-DLMP257B factory QtUI compilation manual V1.0