



Example of directory structure for Packages

Example of directory structure for Packages



Contents

1. Example of directory structure for Packages	3
2. Boot chains overview	18
3. OpenEmbedded	33
4. OpenSTLinux distribution	48
5. Package repository for OpenSTLinux distribution	63
6. STM32CubeMP1 Package	78
7. STM32MP15 Flash mapping	93
8. Standard SDK directory structure	108
9. U-Boot overview	123



A quality version of this page, approved on 24 June 2020, was based off this revision.

Contents

1 Article purpose	4
2 Creating the structure	5
3 Focus on the Starter Package directory	6
4 Focus on the Developer Package directory	8
5 Focus on the Distribution Package directory	11
6 Appendix A: directory structure after build (Developer Package)	13
7 Appendix B: directory structure after build (Distribution Package)	16



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0  STM32MPU Embedded Software release
├── Developer-Package       Developer Package installation directory
├── Distribution-Package   Distribution Package installation directory
└── Starter-Package        Starter Package installation directory
```

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0  STM32MPU Embedded Software release
├── Developer-Package       Developer Package installation
│   └── directory
│       ├── SDK             SDK for OpenSTLinux distribution
│       ├── STM32Cube_FW_MP1_V1.5.0  STM32CubeMP1 Package
│       └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  Linux kernel, U-Boot, TF-A and OP-TEE OS source code (OpenSTLinux distribution)
├── Distribution-Package   Distribution Package installation
│   └── directory
│       └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution (full source code and OpenEmbedded-based build framework)
└── Starter-Package        Starter Package installation
    └── directory
        └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  Software image (binaries)
```



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard
│               │       size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│           ├── scripts
│           │   └── create_sdcard_from_flashlayout.sh
│           ├── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│           │   └── tfs partition
│           ├── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│           │   └── rfs partition
│           ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│           ├── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│           │   └── dorfs partition
│           ├── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│           │   └── tfs partition
│           ├── st-image-weston-openstlinux-weston-stm32mp1.license
│           ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│           ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│           ├── [...]
│           ├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│           │   └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│           │       ├── [...]
│           │       └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│           ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│           │   └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│           ├── [...]
│           ├── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│           │   └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│           ├── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│           │   └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

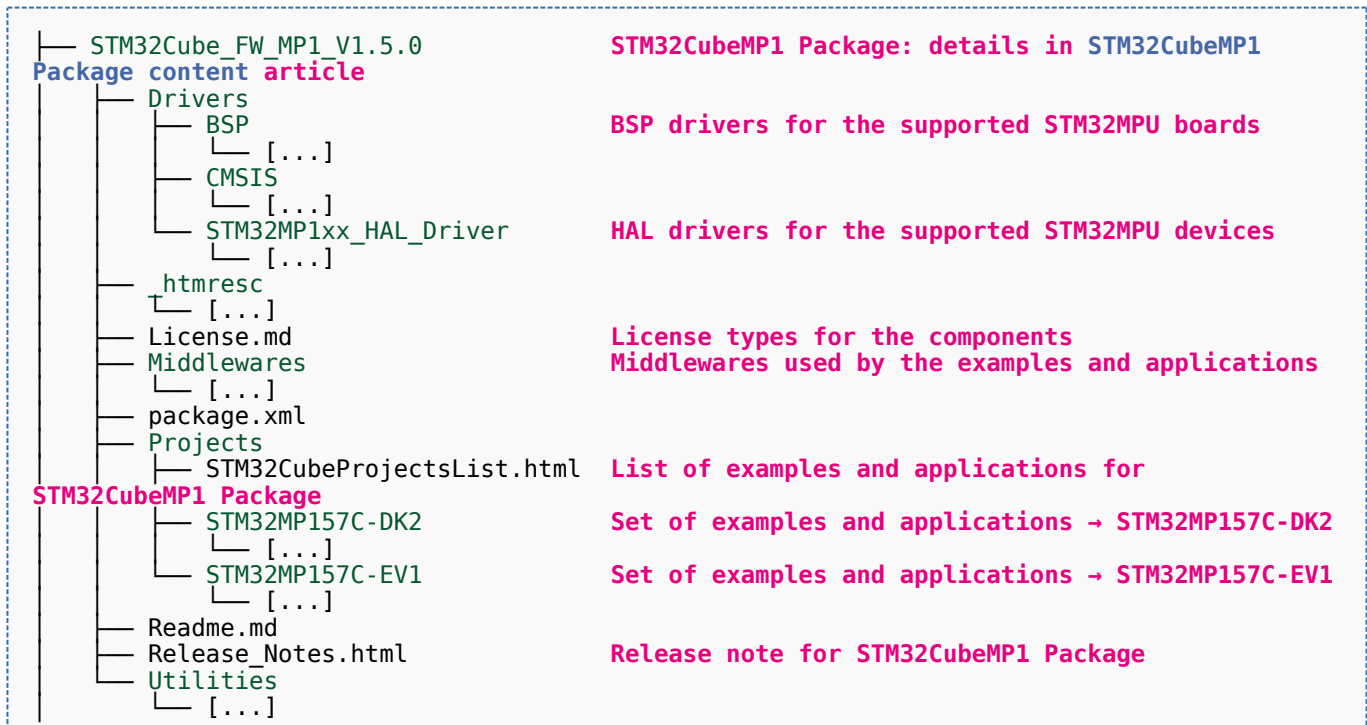
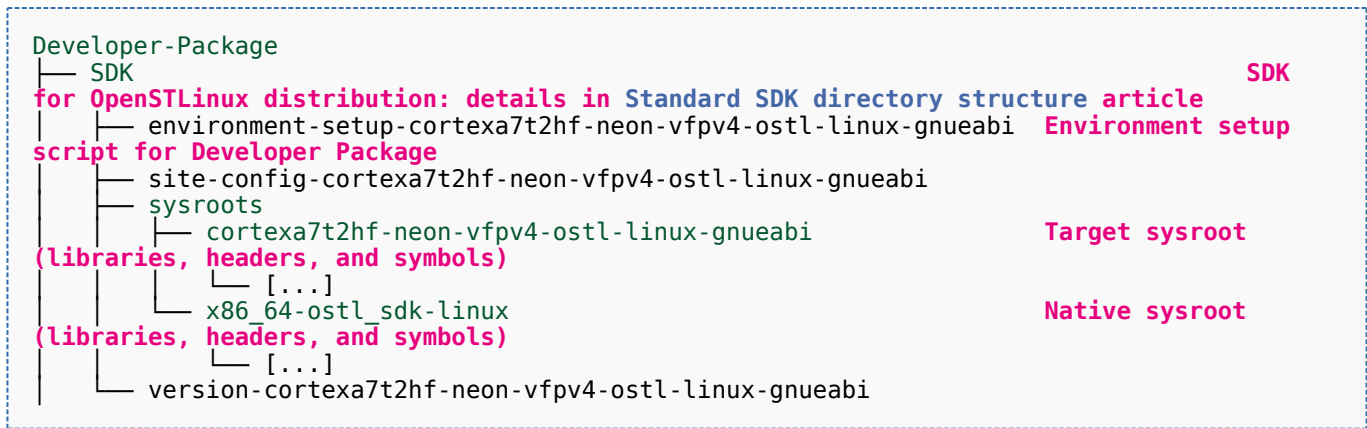
```
├─ u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└─ [...]
```



4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm® Cortex®-A processor):
 - Linux® kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm® Cortex®-M processor)





```

└─ stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
distribution
├─ images
│   └─ stm32mp1
directory
├─ tf-a-bl2-optee.elf           Debug symbol files installation
TEE OS → trusted boot firmware stage
├─ tf-a-bl2-trusted.elf       Debug symbol file for TF-A, with OP-
boot firmware stage
├─ tf-a-bl32-trusted.elf     Debug symbol file for TF-A → trusted
software stage
├─ u-boot-stm32mp157a-dk1-optee.elf Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157A-DK1
├─ u-boot-stm32mp157a-dk1-trusted.elf Debug symbol file for U-Boot →
STM32MP157A-DK1
├─ u-boot-stm32mp157c-dk2-optee.elf Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-trusted.elf Debug symbol file for U-Boot →
STM32MP157C-DK2
├─ u-boot-stm32mp157c-ev1-optee.elf Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-EV1
├─ u-boot-stm32mp157c-ev1-trusted.elf Debug symbol file for U-Boot →
STM32MP157C-EV1
├─ vmlinux
└─ [...]

```

```

└─ sources
├─ arm-openstlinux_weston-linux-gnueabi
│   └─ linux-5.10.61
│       ├── [*].patch           ST patches for Linux kernel
│       ├── fragment-[*].config ST configuration fragments for Linux kernel
│       ├── linux-5.10.61      Linux kernel source code directory
│       ├── linux-5.10.61.tar.xz
│       ├── README.HOW_TO.txt  Helper file for Linux kernel management: referenc
│       └─ series
code directory
└─ e for Linux kernel build

```

```

└─ optee-os-stm32mp-3.12.0-stm32mp-r2 OP-TEE OS installation directory
├─ [*].patch                       ST patches for OP-TEE OS
├─ optee-os-stm32mp-3.12.0-stm32mp-r2
├─ Makefile.sdk                     Makefile for the OP-TEE OS compilation
├─ optee-os-stm32mp-3.12.0-stm32mp-r2-r0.tar.gz OP-TEE OS source
code directory
├─ README.HOW_TO.txt               Helper file for OP-TEE OS management: reference
└─ series
for OP-TEE OS build

```

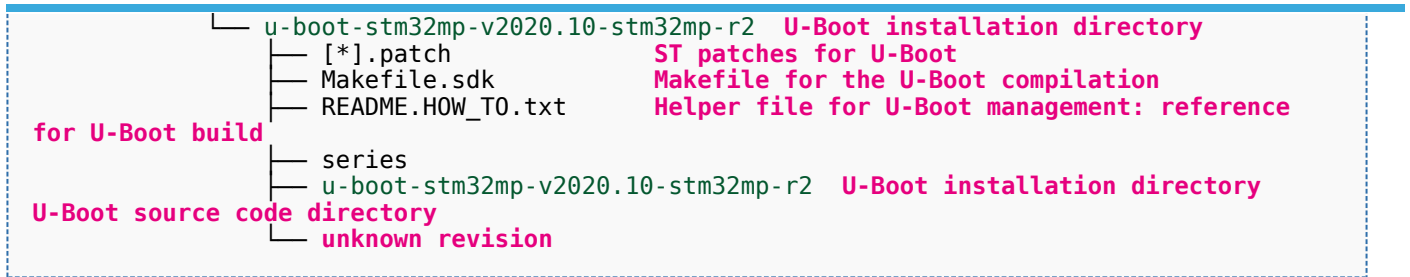
```

└─ tf-a-stm32mp-v2.4-stm32mp-r2 TF-A installation directory
├─ [*].patch                       ST patches for TF-A
├─ tf-a-stm32mp-v2.4-stm32mp-r2 TF-A source code directory
├─ Makefile.sdk                     Makefile for the TF-A compilation
├─ README.HOW_TO.txt               Helper file for TF-A management: reference
code directory
├─ series
└─ tf-a-stm32mp-v2.4-stm32mp-r2-r0.tar.gz
for TF-A build

```



Example of directory structure for Packages



Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.

```

Distribution-Package
├── openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution
│   └── layers
│       ├── meta-openembedded  Collection of layers for the OpenEmbedded-Core universe (Op
│       │                       enEmbedded standard)
│       │   └── [...]
│       └── meta-qt5           QT5 layer for OpenEmbedded (standard)
│           └── [...]

```

```

├── meta-st
│   ├── meta-st-openstlinux  STMicroelectronics layer that
│   │                       contains the settings of the frameworks and images for the OpenSTLinux distribution
│   │
│   ├── recipes-st
│   │   ├── images
│   │   │   └── st-image-core.bb  Core image for OpenSTLinux
│   │   └── st-image-weston.bb   Weston image with basic Wayland
│   │                               support for OpenSTLinux distribution: recommended setup
│   └── [...]
│       ├── packagegroups
│       │   └── [...]

```

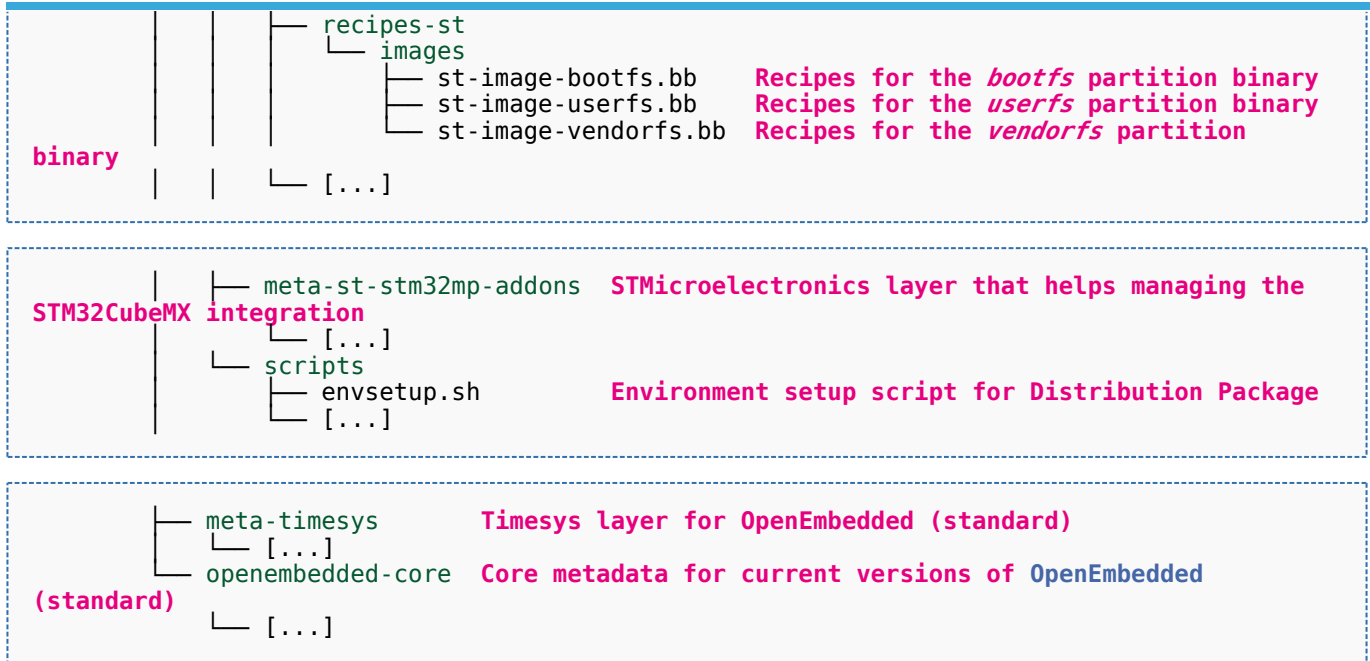
```

├── meta-st-stm32mp  STMicroelectronics layer that contains
│   │               the description of the BSP for the STM32 MPU devices
│   │
│   ├── recipes-bsp
│   │   ├── alsa
│   │   │   └── [...]
│   │   └── drivers  Recipes for Vivante GCNANO GPU kernel
│   │
│   ├── [...]
│   ├── trusted-firmware-a  Recipes for TF-A
│   │   └── [...]
│   ├── u-boot              Recipes for U-Boot
│   │   └── [...]
│   ├── recipes-extended
│   │   └── m4projects       Recipes for STM32Cube MPU Package
│   │       └── [...]
│   │       └── [...]
│   ├── recipes-graphics
│   │   └── gcnano-userland  Recipes for Vivante libraries OpenGL
│   │       └── [...]
│   │       └── [...]
│   ├── recipes-kernel
│   │   ├── linux           Recipes for Linux kernel
│   │   │   └── [...]
│   │   └── linux-firmware  Recipes for Linux firmwares (example,
│   │                       Bluetooth firmware)
│   │       └── [...]

```



Example of directory structure for Packages

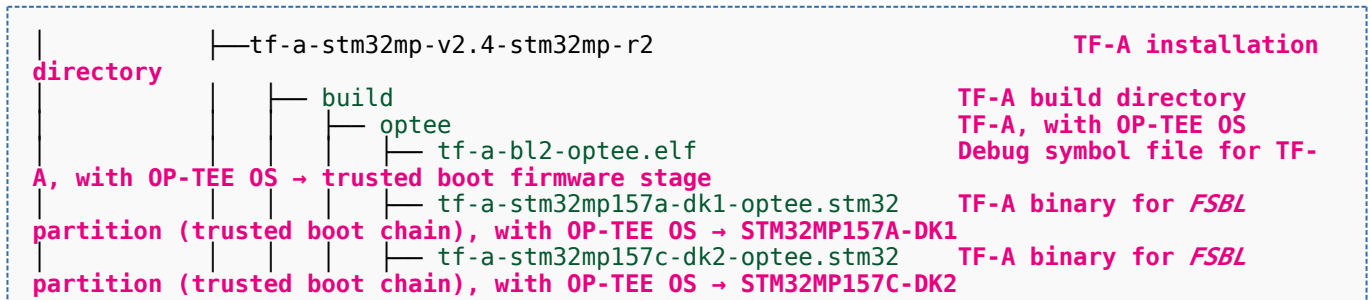
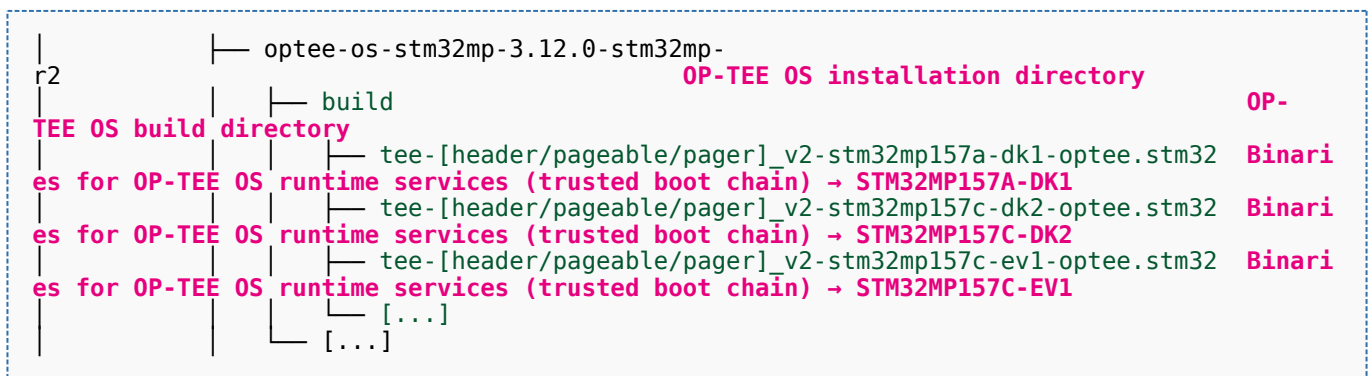
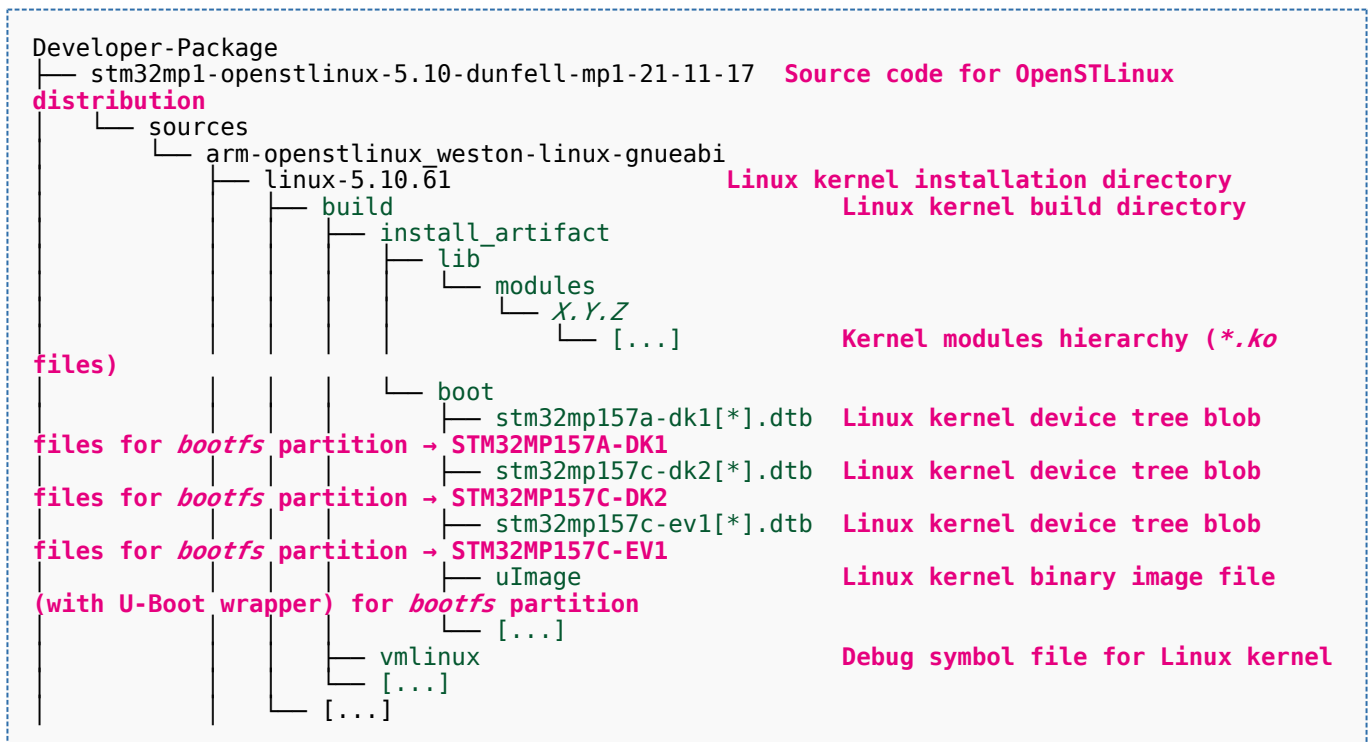


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv         Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv       Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv      Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv     Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv          Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv          Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv         Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv          Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv          Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv         Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv           Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv           Flash layout file
```




Example of directory structure for Packages

```

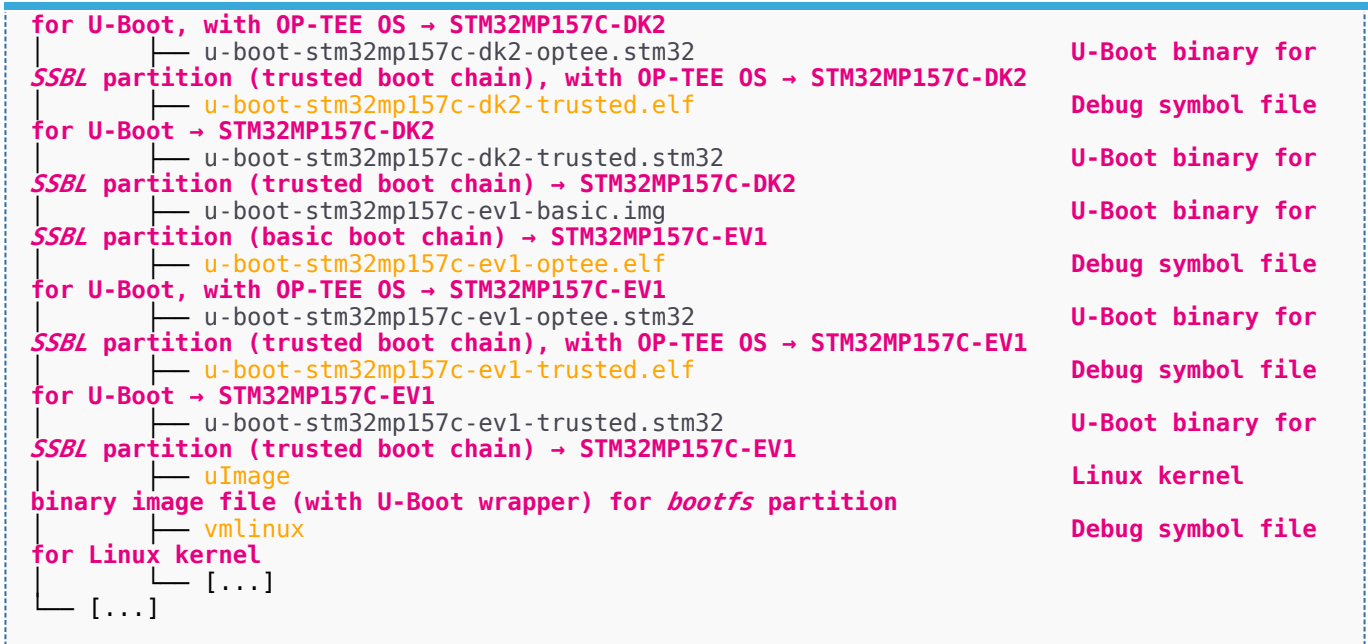
for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv      Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4      Binary for bootfs
partition └── st-image-userfs-openstlinux-weston-stm32mp1.ext4    Binary for userfs
s partition └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for vendorfs
partition └── st-image-weston-openstlinux-weston-stm32mp1.ext4   Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1
├── stm32mp157a-dk1[*].dtb                                         Linux kernel
device tree blob files for bootfs partition → STM32MP157C-DK2
├── stm32mp157c-dk2[*].dtb                                         Linux kernel
device tree blob files for bootfs partition → STM32MP157C-EV1
├── stm32mp157c-e[*].dtb                                           Linux kernel
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32   Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32   Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32   Binaries for OP-
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-optee.elf                                             Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                           Debug symbol file
for TF-A → runtime software stage
├── tf-a-bl32-trusted.elf                                           Debug symbol file
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-optee.stm32                               TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32                            TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                               TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                            TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                               TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                            TF-A binary for FS
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                         U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                         U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                         U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-basic.img                                U-Boot binary for
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                               Debug symbol file
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                          U-Boot binary for
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                          U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157c-dk2-basic.img                                U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                               Debug symbol file

```



Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 25.09.2020 - 08:36 / Revision: 25.09.2020 - 08:35

Contents

1 Article purpose	19
2 Creating the structure	20
3 Focus on the Starter Package directory	21
4 Focus on the Developer Package directory	23
5 Focus on the Distribution Package directory	26
6 Appendix A: directory structure after build (Developer Package)	28
7 Appendix B: directory structure after build (Distribution Package)	31



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
├── Distribution-Package
└── Starter-Package
```

STM32MPU Embedded Software release
Developer Package installation directory
Distribution Package installation directory
Starter Package installation directory

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
│   ├── SDK
│   ├── STM32Cube_FW_MP1_V1.5.0
│   └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│       ├── TEE OS source code (OpenSTLinux distribution)
│       └── Distribution-Package
│           ├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│           └── Starter-Package
│               └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
```

STM32MPU Embedded Software release
Developer Package installation
SDK for OpenSTLinux distribution
STM32CubeMP1 Package
Linux kernel, U-Boot, TF-A and OP-
TEE OS source code (OpenSTLinux distribution)
Distribution Package installation
OpenSTLinux distribution (full
source code and OpenEmbedded-based build framework)
Starter Package installation
Software image (binaries)



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard
│               │       size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│                   ├── scripts
│                   │   └── create_sdcard_from_flashlayout.sh
│                   └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│                       └── tfs partition
│                           ├── st-image-bootfs-openstlinux-weston-stm32mp1.manifest
│                           └── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│                               └── rfs partition
│                                   ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│                                   └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│                                       └── dorfs partition
│                                           └── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│                                               └── tfs partition
│                                                   ├── st-image-weston-openstlinux-weston-stm32mp1.license
│                                                   ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│                                                   ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│                                                   ├── [...]
│                                                   └── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│                                                       └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│                                                           ├── [...]
│                                                           └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│                                                               └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│                                                                   └── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│                                                                       └── for FSBL partition (trusted boot chain) → STM32MP157C-DK2
│                                                                           ├── [...]
│                                                                           └── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│                                                                               └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│                                                                                   └── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│                                                                                       └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

```
├─ u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└─ [...]
```



4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)

```

Developer-Package
├── SDK
│   └── environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│       └── site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           ├── sysroots
│           │   ├── cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           │   │   ├── [...]
│           │   │   └── x86_64-ostl_sdk-linux
│           │   │       ├── [...]
│           │   └── version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           └── [...]
└── [...]
  
```

for OpenSTLinux distribution: details in Standard SDK directory structure article

script for Developer Package

Environment setup

Target sysroot

Native sysroot

(libraries, headers, and symbols)

(libraries, headers, and symbols)

```

├── STM32Cube_FW_MP1_V1.5.0
│   ├── Drivers
│   │   ├── BSP
│   │   │   ├── [...]
│   │   │   └── CMSIS
│   │   │       ├── [...]
│   │   └── STM32MP1xx_HAL_Driver
│   │       ├── [...]
│   ├── htmresc
│   │   ├── [...]
│   ├── License.md
│   ├── Middlewares
│   │   ├── [...]
│   ├── package.xml
│   ├── Projects
│   │   ├── STM32CubeProjectsList.html
│   │   ├── STM32MP157C-DK2
│   │   │   ├── [...]
│   │   └── STM32MP157C-EV1
│   │       ├── [...]
│   ├── Readme.md
│   ├── Release_Notes.html
│   ├── Utilities
│   │   ├── [...]
  
```

STM32CubeMP1 Package: details in STM32CubeMP1 Package content article

BSP drivers for the supported STM32MPU boards

HAL drivers for the supported STM32MPU devices

License types for the components

Middlewares used by the examples and applications

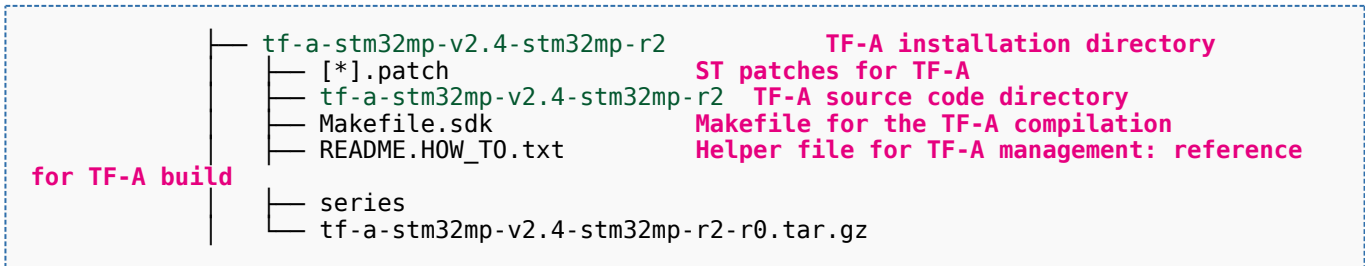
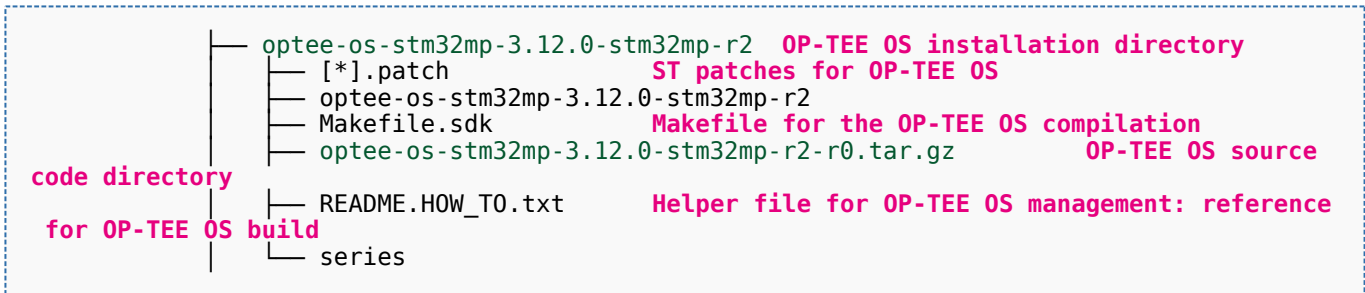
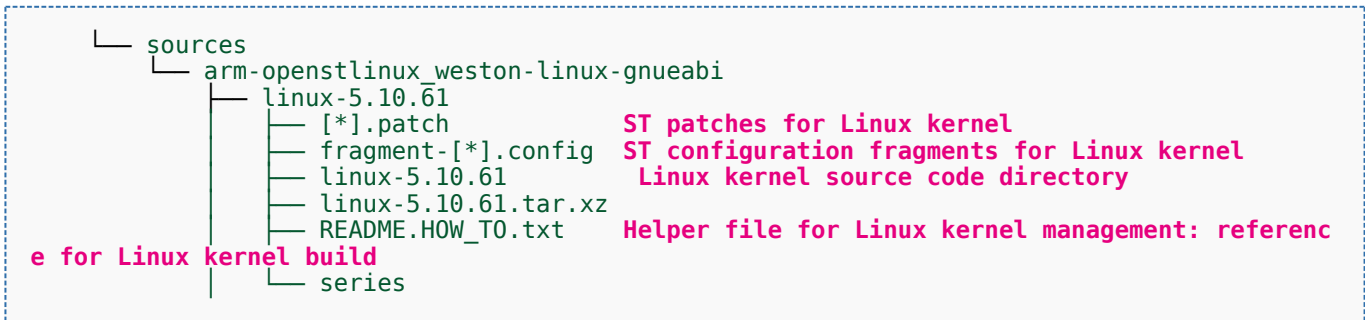
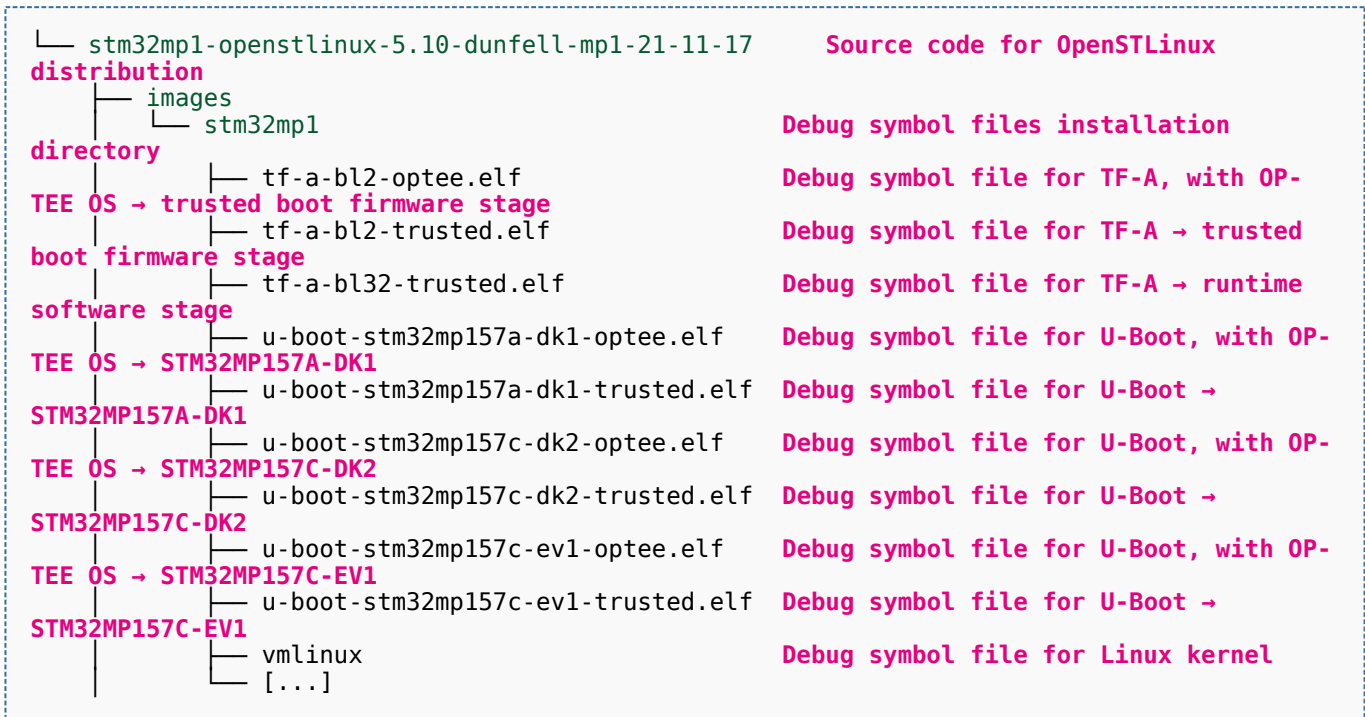
STM32CubeMP1 Package

List of examples and applications for

Set of examples and applications → STM32MP157C-DK2

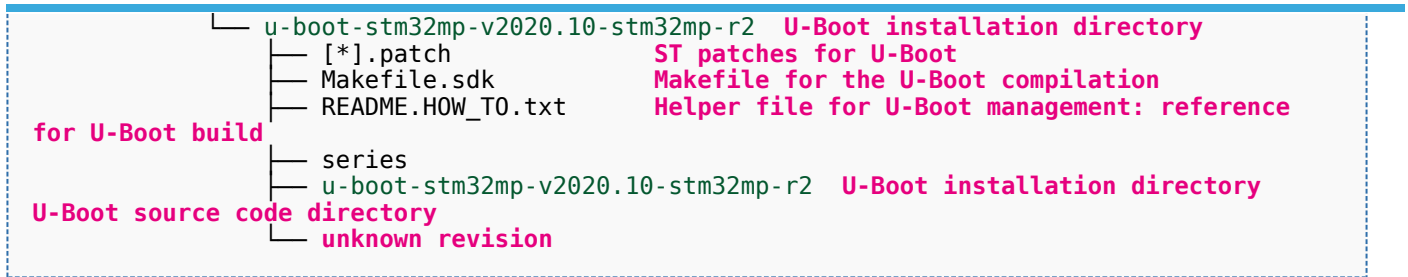
Set of examples and applications → STM32MP157C-EV1

Release note for STM32CubeMP1 Package





Example of directory structure for Packages

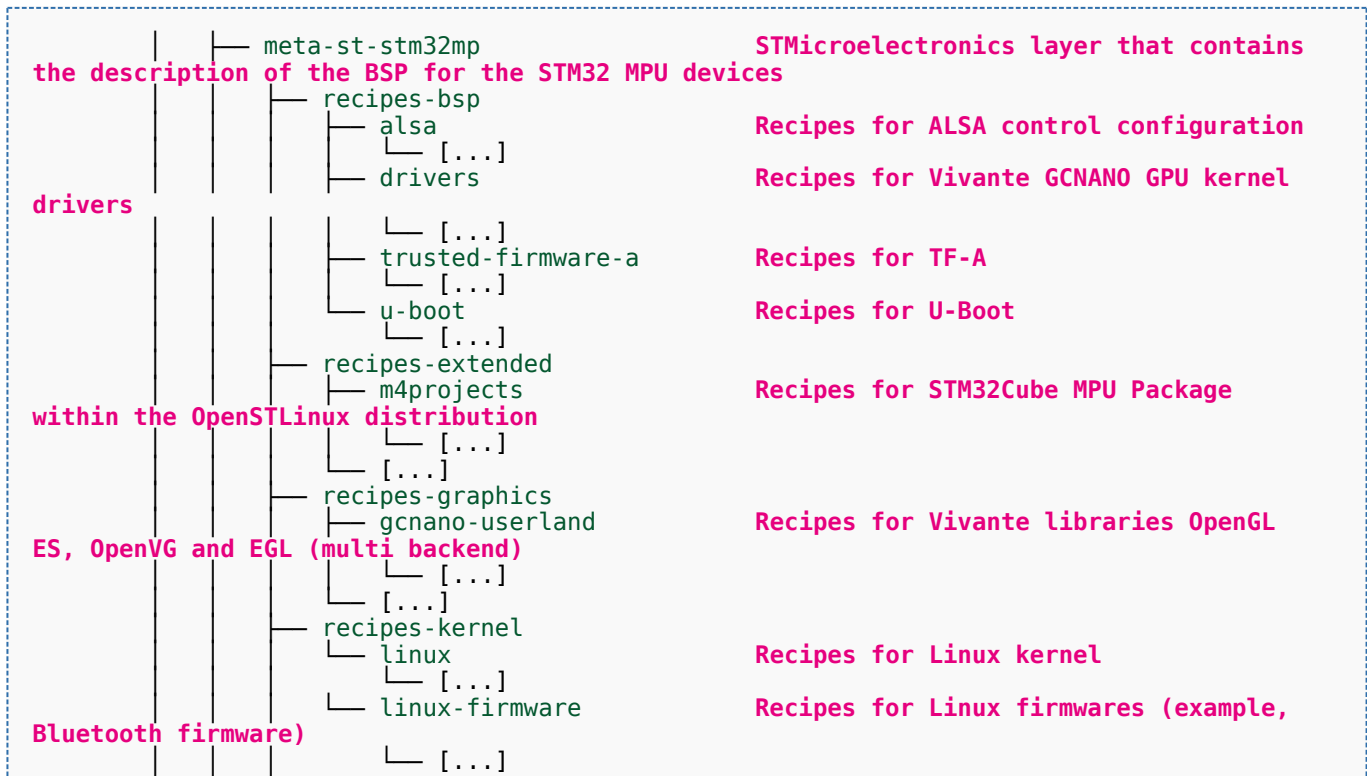
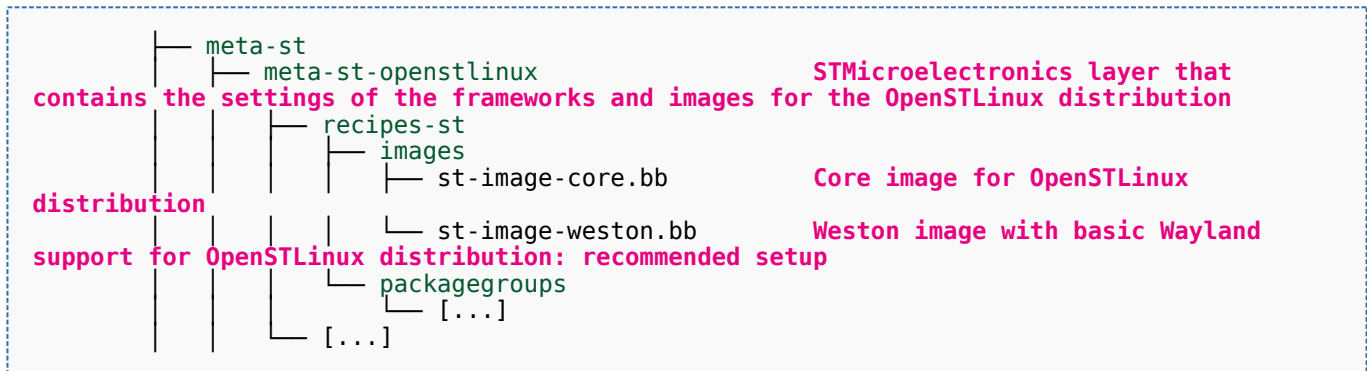
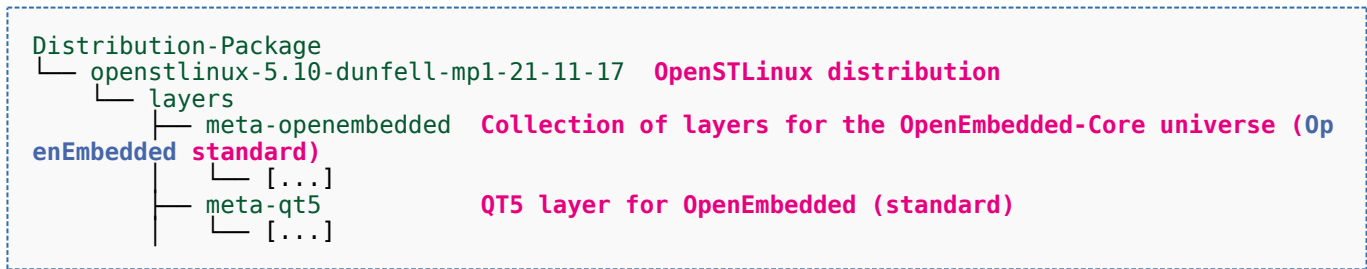


Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



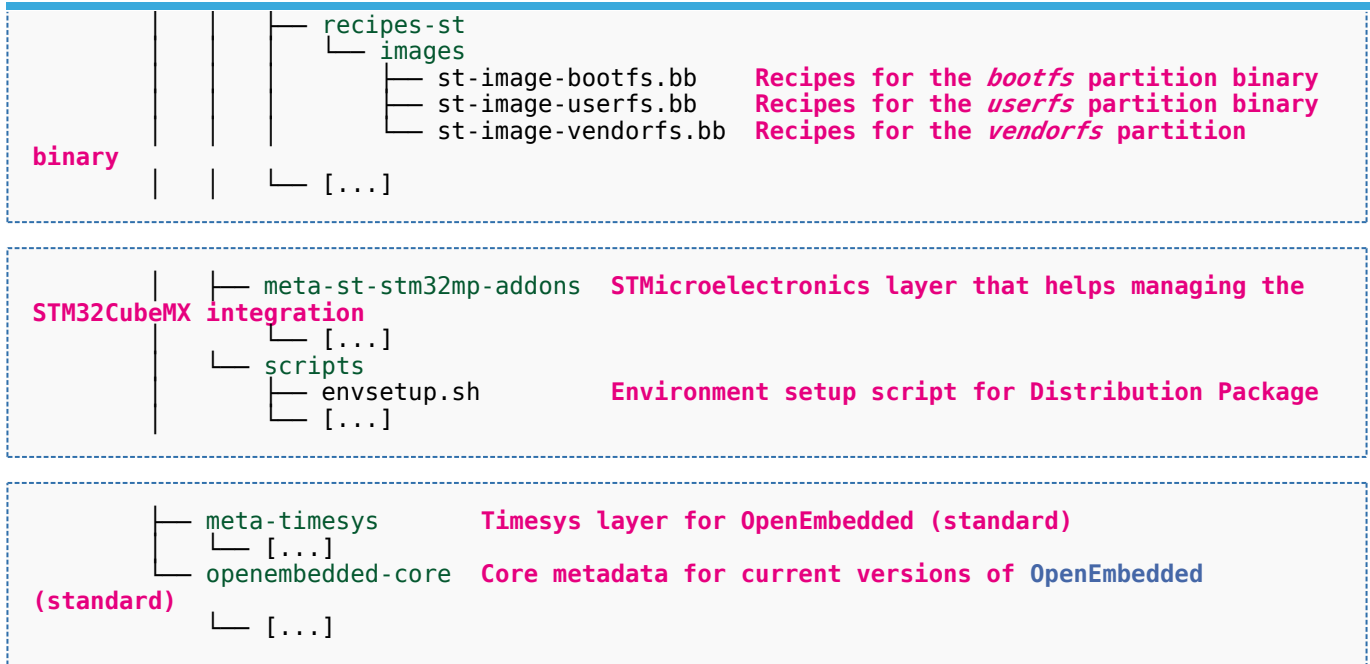
5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.





Example of directory structure for Packages

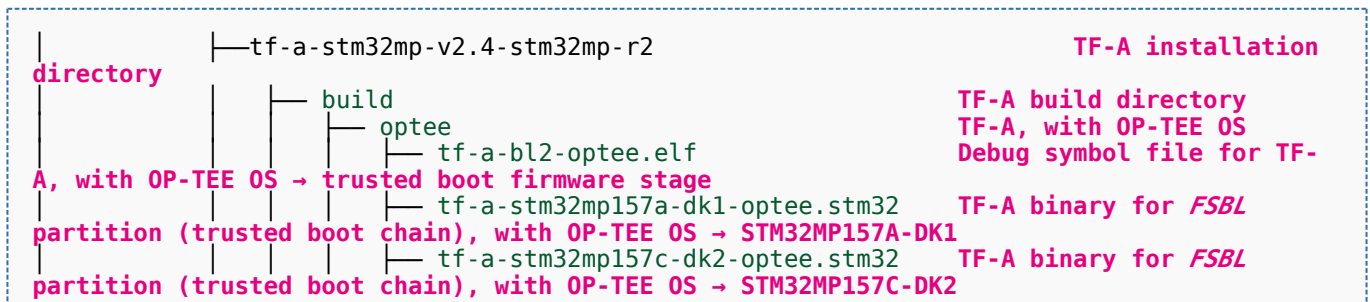
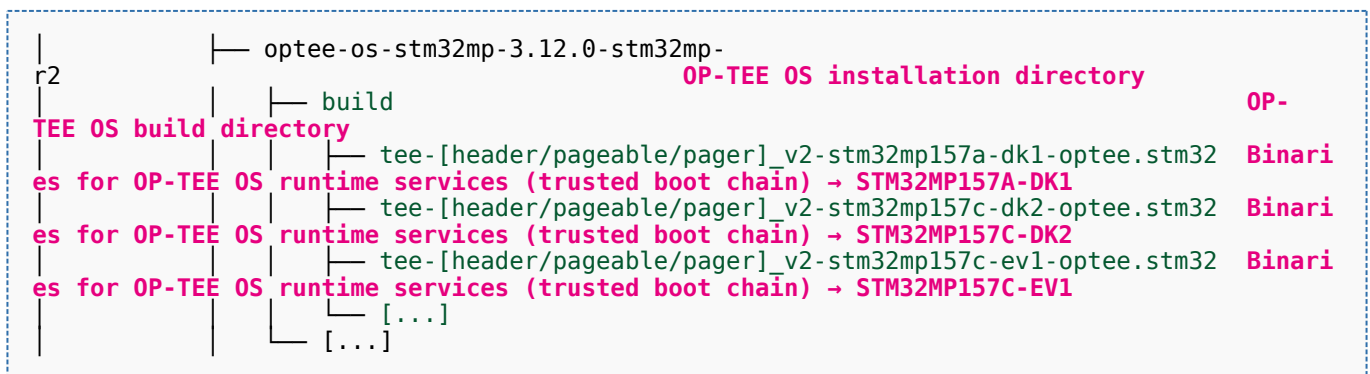
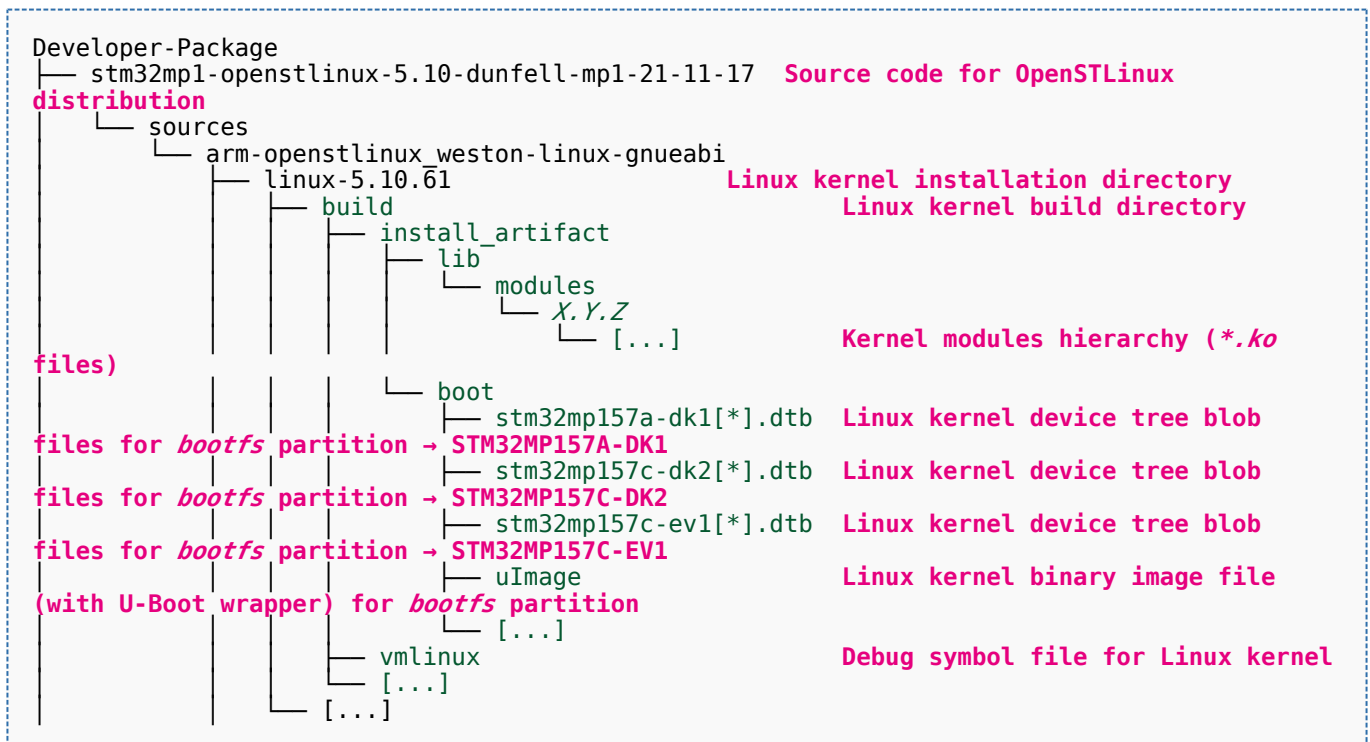


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages

partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-optee.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	trusted	TF-A, without OP-TEE OS
A → trusted boot firmware stage	tf-a-bl2-trusted.elf	Debug symbol file for TF-
A → trusted boot firmware stage	tf-a-bl32-trusted.elf	Debug symbol file for TF-
partition (trusted boot chain) → STM32MP157A-DK1	tf-a-stm32mp157a-dk1-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	tf-a-stm32mp157c-dk2-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-trusted.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	[...]	

installation directory	u-boot-stm32mp-v2020.10-stm32mp-r2	U-Boot
for basic boot chain	build-basic	U-Boot build directory
partition (basic boot chain) → STM32MP157A-DK1	u-boot-spl.stm32-stm32mp157a-dk1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-spl.stm32-stm32mp157c-dk2-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-spl.stm32-stm32mp157c-ev1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for <i>SSBL</i>
for trusted boot chain, with OP-TEE OS	build-optee	U-Boot build directory
Boot, with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
for trusted boot chain	build-trusted	U-Boot build directory
Boot → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
	[...]	



Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv         Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv       Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv         Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv         Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv         Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv         Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv         Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv         Flash layout file
```



Example of directory structure for Packages

```

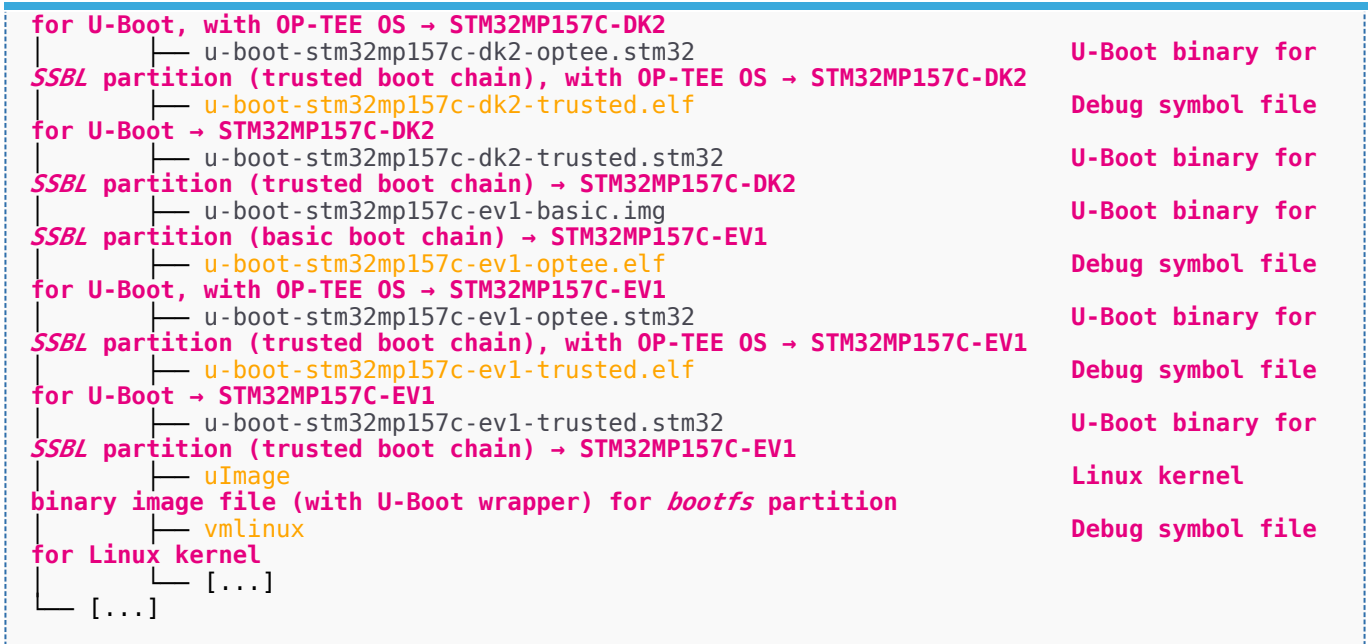
for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv          Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition ─── st-image-bootfs-openstlinux-weston-stm32mp1.ext4          Binary for bootfs
partition ─── st-image-userfs-openstlinux-weston-stm32mp1.ext4        Binary for userfs
partition ─── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4      Binary for vendorfs
partition ─── st-image-weston-openstlinux-weston-stm32mp1.ext4        Binary for rootfs
partition ─── stm32mp157a-dk1[*].dtb                                    Linux kernel
device tree blob files for bootfs partition → STM32MP157A-DK1        Linux kernel
├── stm32mp157c-dk2[*].dtb
device tree blob files for bootfs partition → STM32MP157C-DK2        Linux kernel
├── stm32mp157c-e[*].dtb
device tree blob files for bootfs partition → STM32MP157C-EV1
tee- [header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32          Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
tee- [header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32          Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
tee- [header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32          Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tf-a-bl2-optee.elf                                                Debug symbol file
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                              Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl32-trusted.elf                                             Debug symbol file
for TF-A → runtime software stage
├── tf-a-stm32mp157a-dk1-optee.stm32                                  TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-trusted.stm32                                TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                                  TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                                TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                                  TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                                TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                            U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                            U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                            U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-stm32mp157a-dk1-basic.img                                    U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                                  Debug symbol file
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                              U-Boot binary for
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.elf                                Debug symbol file
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                              U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157c-dk2-basic.img                                    U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                                  Debug symbol file

```

Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 09.12.2020 - 17:46 / Revision: 09.12.2020 - 13:33

Contents

1 Article purpose	34
2 Creating the structure	35
3 Focus on the Starter Package directory	36
4 Focus on the Developer Package directory	38
5 Focus on the Distribution Package directory	41
6 Appendix A: directory structure after build (Developer Package)	43
7 Appendix B: directory structure after build (Distribution Package)	46



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
├── Distribution-Package
└── Starter-Package
```

STM32MPU Embedded Software release
Developer Package installation directory
Distribution Package installation directory
Starter Package installation directory

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
│   ├── SDK
│   ├── STM32Cube_FW_MP1_V1.5.0
│   └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│       ├── TEE OS source code (OpenSTLinux distribution)
│       └── Distribution-Package
│           ├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│           └── Starter-Package
│               └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
```

STM32MPU Embedded Software release
Developer Package installation
SDK for OpenSTLinux distribution
STM32CubeMP1 Package
Linux kernel, U-Boot, TF-A and OP-
TEE OS source code (OpenSTLinux distribution)
Distribution Package installation
OpenSTLinux distribution (full
source code and OpenEmbedded-based build framework)
Starter Package installation
Software image (binaries)



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard
│               │       size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│           ├── scripts
│           │   ├── create_sdcard_from_flashlayout.sh
│           │   └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│           └── tfs partition
│               ├── st-image-bootfs-openstlinux-weston-stm32mp1.manifest
│               ├── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│               └── rfs partition
│                   ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│                   ├── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│                   └── dorfs partition
│                       ├── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│                       └── tfs partition
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.license
│                           ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│                           ├── [...]
│                           └── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│                               └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│                                   ├── [...]
│                                   └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│                                       └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│                                           ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│                                           └── for FSBL partition (trusted boot chain) → STM32MP157C-DK2
│                                               ├── [...]
│                                               └── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│                                                   └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│                                                       ├── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│                                                       └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

```
├─ u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└─ [...]
```



4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)

```

Developer-Package
├── SDK
│   └── environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│       └── site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           ├── sysroots
│           │   ├── cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           │   │   └── [...]
│           │   └── x86_64-ostl_sdk-linux
│           │       └── [...]
│           └── version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
└── [...]
  
```

for OpenSTLinux distribution: details in Standard SDK directory structure article

script for Developer Package

Environment setup

Target sysroot

Native sysroot

(libraries, headers, and symbols)

(libraries, headers, and symbols)

```

├── STM32Cube_FW_MP1_V1.5.0
│   ├── Drivers
│   │   ├── BSP
│   │   │   └── [...]
│   │   ├── CMSIS
│   │   │   └── [...]
│   │   └── STM32MP1xx_HAL_Driver
│   │       └── [...]
│   ├── htmresc
│   │   └── [...]
│   ├── License.md
│   ├── Middlewares
│   │   └── [...]
│   ├── package.xml
│   ├── Projects
│   │   ├── STM32CubeProjectsList.html
│   │   ├── STM32MP157C-DK2
│   │   │   └── [...]
│   │   └── STM32MP157C-EV1
│   │       └── [...]
│   ├── Readme.md
│   ├── Release_Notes.html
│   └── Utilities
│       └── [...]
  
```

STM32CubeMP1 Package: details in STM32CubeMP1 Package content article

BSP drivers for the supported STM32MPU boards

HAL drivers for the supported STM32MPU devices

License types for the components

Middlewares used by the examples and applications

STM32CubeMP1 Package

List of examples and applications for

Set of examples and applications → STM32MP157C-DK2

Set of examples and applications → STM32MP157C-EV1

Release note for STM32CubeMP1 Package



```

└─ stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
  distribution
  └─ images
    └─ stm32mp1
      directory
      └─ tf-a-bl2-optee.elf      Source code for OpenSTLinux
      TEE OS → trusted boot firmware stage
      └─ tf-a-bl2-trusted.elf   Debug symbol files installation
      boot firmware stage
      └─ tf-a-bl32-trusted.elf  Debug symbol file for TF-A, with OP-
      software stage
      └─ u-boot-stm32mp157a-dk1-optee.elf  Debug symbol file for TF-A → trusted
      TEE OS → STM32MP157A-DK1
      └─ u-boot-stm32mp157a-dk1-trusted.elf  Debug symbol file for TF-A → runtime
      STM32MP157A-DK1
      └─ u-boot-stm32mp157c-dk2-optee.elf  Debug symbol file for U-Boot, with OP-
      TEE OS → STM32MP157C-DK2
      └─ u-boot-stm32mp157c-dk2-trusted.elf  Debug symbol file for U-Boot →
      STM32MP157C-DK2
      └─ u-boot-stm32mp157c-ev1-optee.elf  Debug symbol file for U-Boot, with OP-
      TEE OS → STM32MP157C-EV1
      └─ u-boot-stm32mp157c-ev1-trusted.elf  Debug symbol file for U-Boot →
      STM32MP157C-EV1
      └─ vmlinux                Debug symbol file for Linux kernel
      └─ [...]

```

```

└─ sources
  └─ arm-openstlinux_weston-linux-gnueabi
    └─ linux-5.10.61
      └─ [*].patch              ST patches for Linux kernel
      └─ fragment-[*].config   ST configuration fragments for Linux kernel
      └─ linux-5.10.61         Linux kernel source code directory
      └─ linux-5.10.61.tar.xz
      └─ README.HOW_TO.txt    Helper file for Linux kernel management: referenc
      └─ series

```

e for Linux kernel build

```

└─ code directory
  └─ optee-os-stm32mp-3.12.0-stm32mp-r2  OP-TEE OS installation directory
    └─ [*].patch                        ST patches for OP-TEE OS
    └─ optee-os-stm32mp-3.12.0-stm32mp-r2
    └─ Makefile.sdk                     Makefile for the OP-TEE OS compilation
    └─ optee-os-stm32mp-3.12.0-stm32mp-r2-r0.tar.gz  OP-TEE OS source
  └─ README.HOW_TO.txt                 Helper file for OP-TEE OS management: reference
  └─ series

```

for OP-TEE OS build

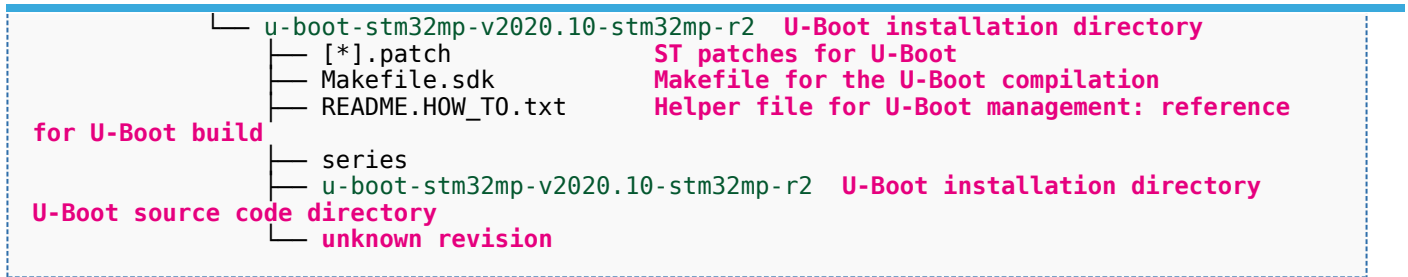
```

└─ for TF-A build
  └─ tf-a-stm32mp-v2.4-stm32mp-r2      TF-A installation directory
    └─ [*].patch                       ST patches for TF-A
    └─ tf-a-stm32mp-v2.4-stm32mp-r2   TF-A source code directory
    └─ Makefile.sdk                   Makefile for the TF-A compilation
    └─ README.HOW_TO.txt              Helper file for TF-A management: reference
  └─ series
  └─ tf-a-stm32mp-v2.4-stm32mp-r2-r0.tar.gz

```



Example of directory structure for Packages



Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.

```

Distribution-Package
├── openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution
│   └── layers
│       ├── meta-openembedded  Collection of layers for the OpenEmbedded-Core universe (Op
│       │   enEmbedded standard)
│       │   ├── [...]
│       │   └── meta-qt5      QT5 layer for OpenEmbedded (standard)
│       │       └── [...]

```

```

├── meta-st
│   ├── meta-st-openstlinux  STMicroelectronics layer that
│   │   contains the settings of the frameworks and images for the OpenSTLinux distribution
│   │   └── recipes-st
│   │       ├── images
│   │       │   └── st-image-core.bb  Core image for OpenSTLinux
│   │       └── st-image-weston.bb  Weston image with basic Wayland
│   │           support for OpenSTLinux distribution: recommended setup
│   │               ├── packagegroups
│   │               │   └── [...]
│   │               └── [...]

```

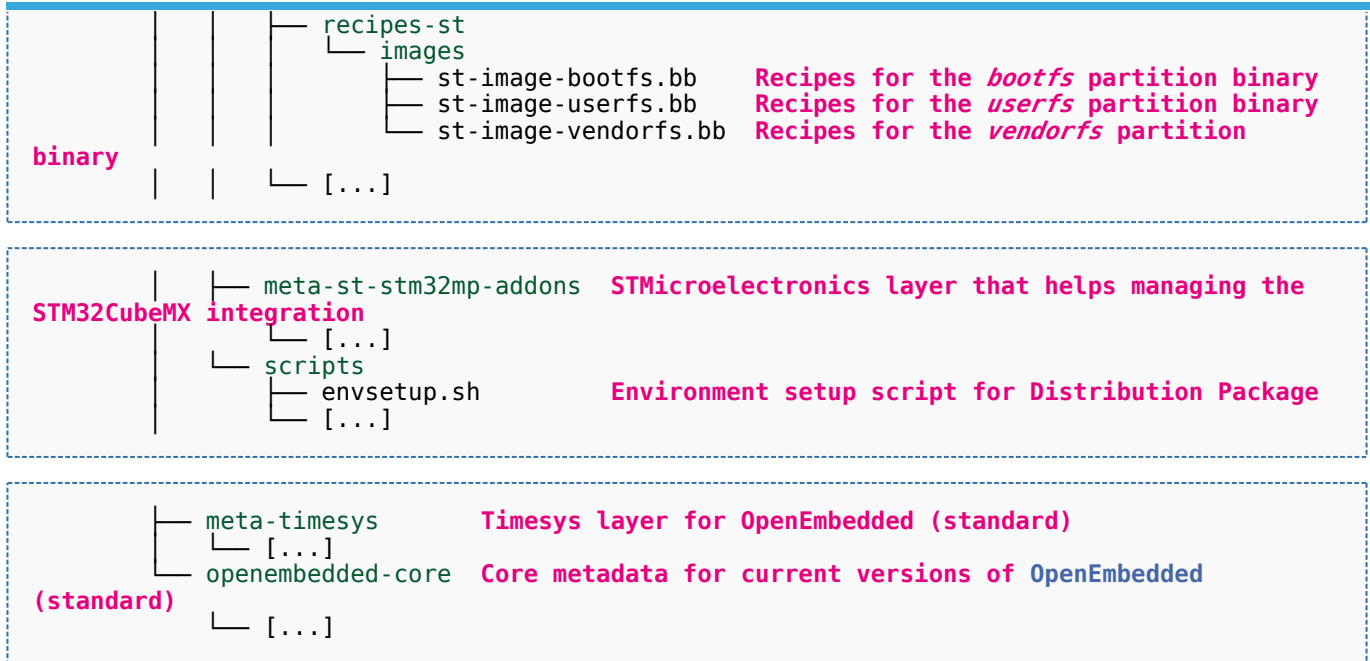
```

├── meta-st-stm32mp  STMicroelectronics layer that contains
│   the description of the BSP for the STM32 MPU devices
│   └── recipes-bsp
│       ├── alsa  Recipes for ALSA control configuration
│       │   └── [...]
│       └── drivers  Recipes for Vivante GCNANO GPU kernel
│           ├── [...]
│           ├── trusted-firmware-a  Recipes for TF-A
│           │   └── [...]
│           └── u-boot  Recipes for U-Boot
│               └── [...]
├── recipes-extended  Recipes for STM32Cube MPU Package
│   ├── m4projects
│   │   ├── [...]
│   │   └── [...]
│   ├── recipes-graphics
│   │   ├── gcnano-userland  Recipes for Vivante libraries OpenGL
│   │   │   ├── [...]
│   │   │   └── [...]
│   │   └── [...]
│   └── recipes-kernel
│       ├── linux  Recipes for Linux kernel
│       │   ├── [...]
│       │   └── linux-firmware  Recipes for Linux firmwares (example,
│       │       Bluetooth firmware)
│       └── [...]

```



Example of directory structure for Packages

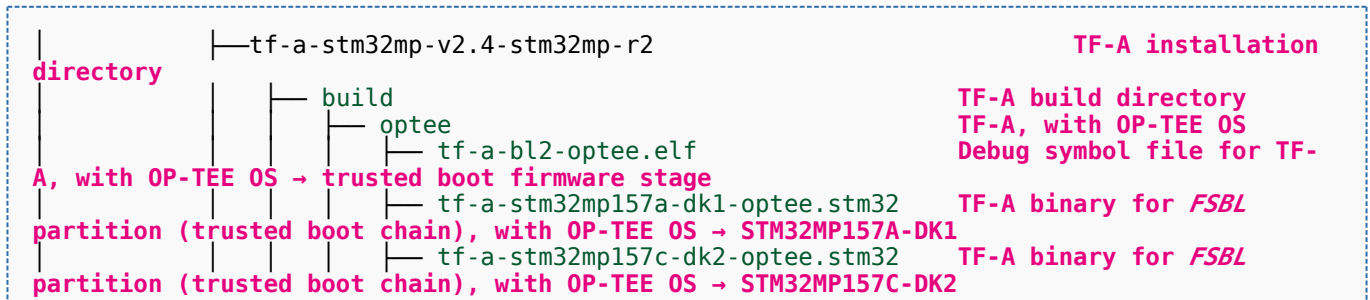
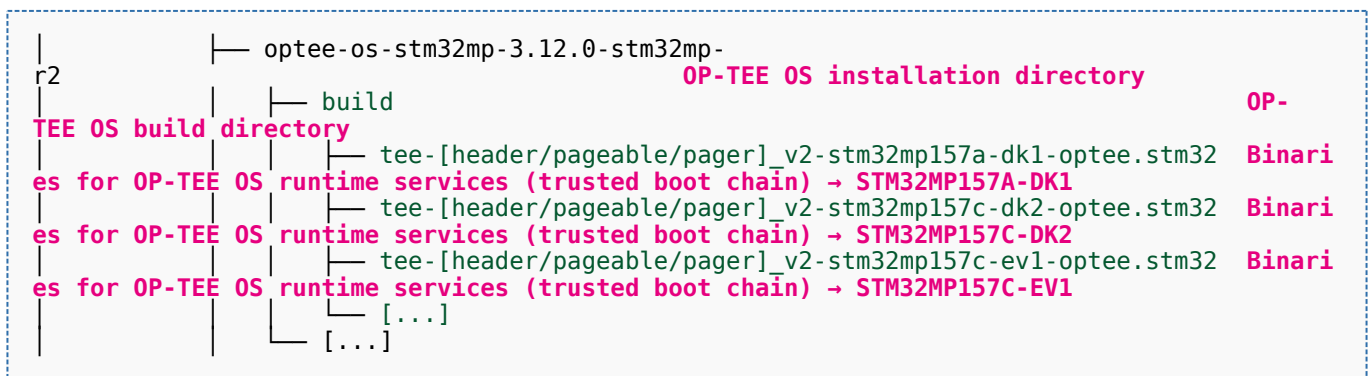
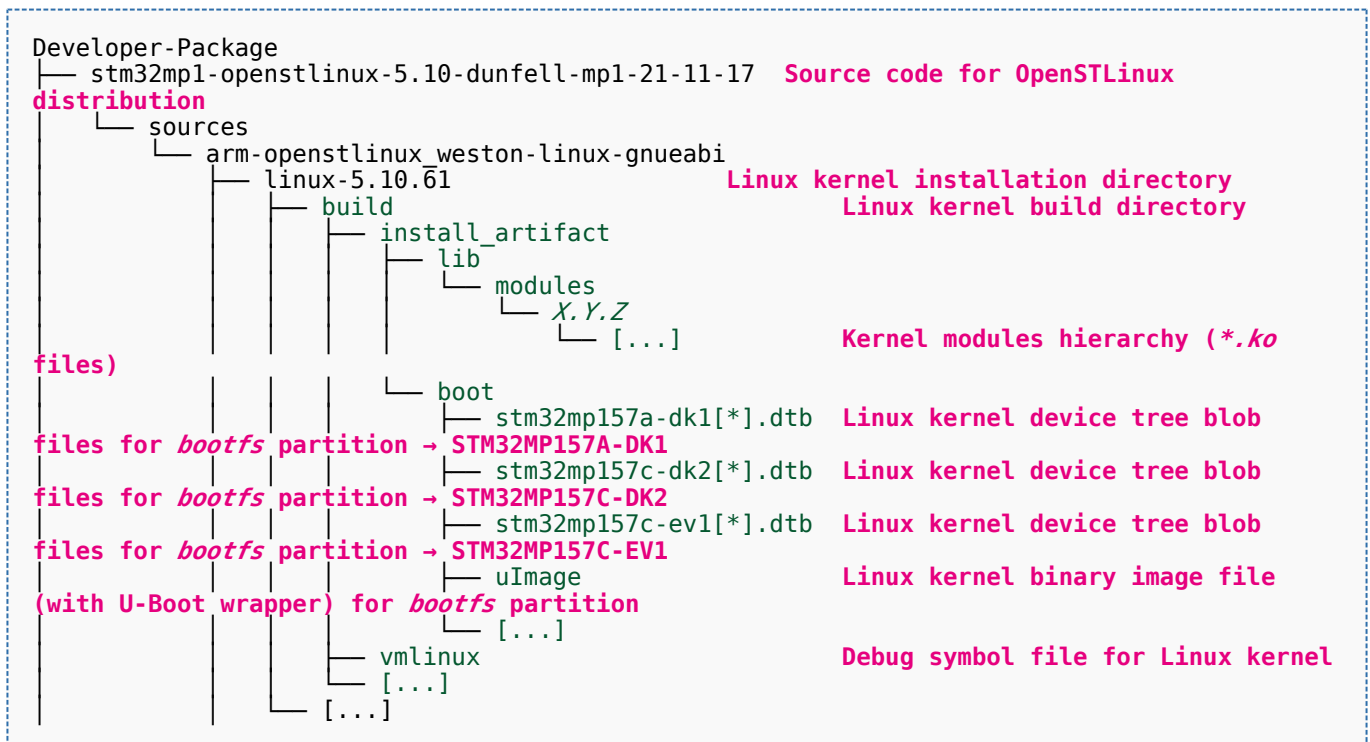


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages

partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-optee.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	trusted	TF-A, without OP-TEE OS
A → trusted boot firmware stage	tf-a-bl2-trusted.elf	Debug symbol file for TF-
A → trusted boot firmware stage	tf-a-bl32-trusted.elf	Debug symbol file for TF-
partition (trusted boot chain) → STM32MP157A-DK1	tf-a-stm32mp157a-dk1-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	tf-a-stm32mp157c-dk2-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-trusted.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	[...]	

installation directory	u-boot-stm32mp-v2020.10-stm32mp-r2	U-Boot
for basic boot chain	build-basic	U-Boot build directory
partition (basic boot chain) → STM32MP157A-DK1	u-boot-spl.stm32-stm32mp157a-dk1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-spl.stm32-stm32mp157c-dk2-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-spl.stm32-stm32mp157c-ev1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for <i>SSBL</i>
for trusted boot chain, with OP-TEE OS	build-optee	U-Boot build directory
Boot, with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
for trusted boot chain	build-trusted	U-Boot build directory
Boot → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
	[...]	



Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv       Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv        Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv     Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv       Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv     Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv       Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv       Flash layout file
```



Example of directory structure for Packages

```

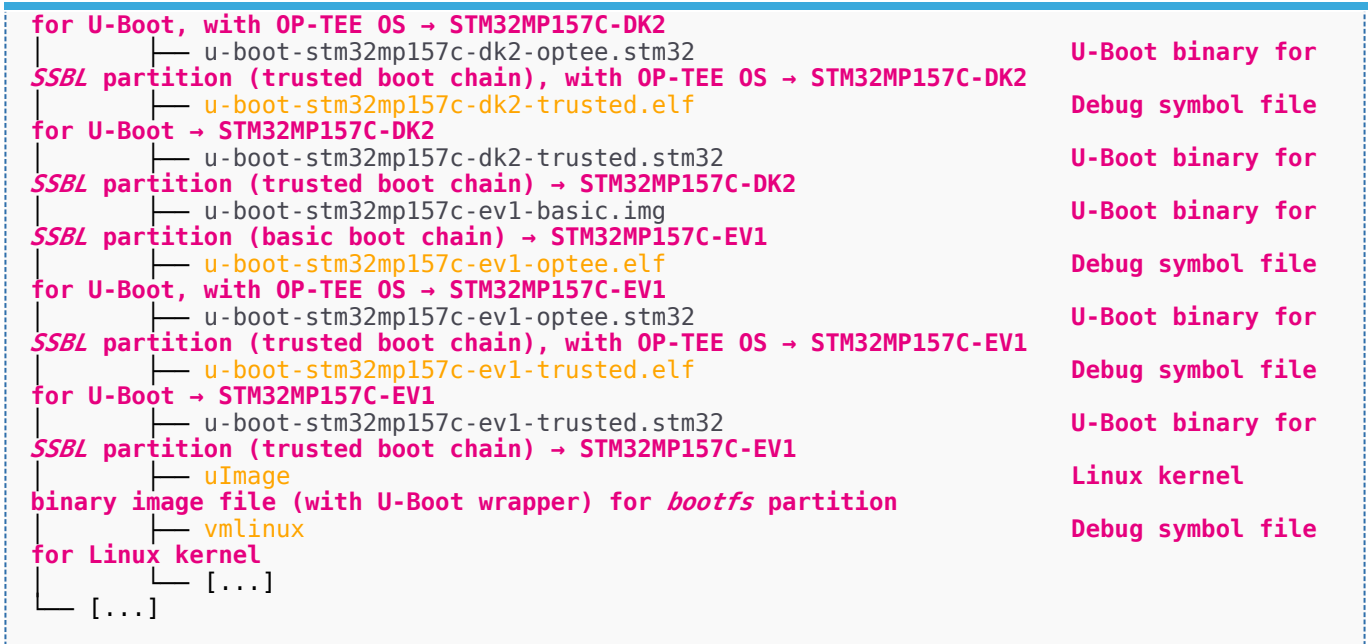
for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv      Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4      Binary for bootfs
partition └── st-image-userfs-openstlinux-weston-stm32mp1.ext4    Binary for userfs
s partition └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for vendorfs
partition └── st-image-weston-openstlinux-weston-stm32mp1.ext4    Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1
├── stm32mp157a-dk1[*].dtb                                          Linux kernel
device tree blob files for bootfs partition → STM32MP157C-DK2
├── stm32mp157c-dk2[*].dtb                                          Linux kernel
device tree blob files for bootfs partition → STM32MP157C-EV1
├── stm32mp157c-e[*].dtb                                           Linux kernel
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32    Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32    Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32    Binaries for OP-
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-optee.elf                                              Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                           Debug symbol file
for TF-A → runtime software stage
├── tf-a-bl32-trusted.elf                                           Debug symbol file
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-optee.stm32                                TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32                             TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                                TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                             TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                                TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                             TF-A binary for FS
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                          U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                          U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                          U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-basic.img                                 U-Boot binary for
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                                Debug symbol file
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157c-dk2-basic.img                                 U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                                Debug symbol file

```

Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 17.11.2021 - 07:46 / Revision: 19.10.2021 - 17:08

Contents

1 Article purpose	49
2 Creating the structure	50
3 Focus on the Starter Package directory	51
4 Focus on the Developer Package directory	53
5 Focus on the Distribution Package directory	56
6 Appendix A: directory structure after build (Developer Package)	58
7 Appendix B: directory structure after build (Distribution Package)	61



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0  STM32MPU Embedded Software release
├── Developer-Package       Developer Package installation directory
├── Distribution-Package    Distribution Package installation directory
└── Starter-Package        Starter Package installation directory
```

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0  STM32MPU Embedded Software release
├── Developer-Package       Developer Package installation
│   └── directory
│       ├── SDK             SDK for OpenSTLinux distribution
│       ├── STM32Cube_FW_MP1_V1.5.0  STM32CubeMP1 Package
│       └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  Linux kernel, U-Boot, TF-A and OP-TEE OS source code (OpenSTLinux distribution)
├── Distribution-Package    Distribution Package installation
│   └── directory
│       └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution (full source code and OpenEmbedded-based build framework)
└── Starter-Package        Starter Package installation
    └── directory
        └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  Software image (binaries)
```



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard
│               │       size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│           ├── scripts
│           │   ├── create_sdcard_from_flashlayout.sh
│           │   └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│           └── tfs partition
│               ├── st-image-bootfs-openstlinux-weston-stm32mp1.manifest
│               ├── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│               └── rfs partition
│                   ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│                   ├── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│                   └── dorfs partition
│                       ├── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│                       └── tfs partition
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.license
│                           ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│                           ├── [...]
│                           └── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│                               └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│                                   ├── [...]
│                                   └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│                                       └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│                                           ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│                                           └── for FSBL partition (trusted boot chain) → STM32MP157C-DK2
│                                               ├── [...]
│                                               └── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│                                                   └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│                                                       ├── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│                                                       └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

```
├─ u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└─ [...]
```



4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)

```

Developer-Package
├── SDK
│   └── environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│       └── site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           ├── sysroots
│           │   ├── cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           │   │   ├── [...]
│           │   │   └── x86_64-ostl_sdk-linux
│           │   │       ├── [...]
│           │   └── version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           └── [...]
└── [...]
  
```

for OpenSTLinux distribution: details in Standard SDK directory structure article

script for Developer Package

Environment setup

Target sysroot

Native sysroot

(libraries, headers, and symbols)

(libraries, headers, and symbols)

```

├── STM32Cube_FW_MP1_V1.5.0
│   ├── Drivers
│   │   ├── BSP
│   │   │   ├── [...]
│   │   │   └── CMSIS
│   │   │       ├── [...]
│   │   └── STM32MP1xx_HAL_Driver
│   │       ├── [...]
│   ├── htmresc
│   │   ├── [...]
│   ├── License.md
│   ├── Middlewares
│   │   ├── [...]
│   ├── package.xml
│   ├── Projects
│   │   ├── STM32CubeProjectsList.html
│   │   ├── STM32MP157C-DK2
│   │   │   ├── [...]
│   │   └── STM32MP157C-EV1
│   │       ├── [...]
│   ├── Readme.md
│   ├── Release_Notes.html
│   ├── Utilities
│   │   ├── [...]
  
```

STM32CubeMP1 Package: details in STM32CubeMP1 Package content article

BSP drivers for the supported STM32MPU boards

HAL drivers for the supported STM32MPU devices

License types for the components

Middlewares used by the examples and applications

STM32CubeMP1 Package

List of examples and applications for

Set of examples and applications → STM32MP157C-DK2

Set of examples and applications → STM32MP157C-EV1

Release note for STM32CubeMP1 Package



```

└─ stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
distribution
├─ images
│ └─ stm32mp1
directory
├─ tf-a-bl2-optee.elf      Debug symbol files installation
TEE OS → trusted boot firmware stage
├─ tf-a-bl2-trusted.elf  Debug symbol file for TF-A, with OP-
boot firmware stage
├─ tf-a-bl32-trusted.elf Debug symbol file for TF-A → trusted
software stage
├─ u-boot-stm32mp157a-dk1-optee.elf Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157A-DK1
├─ u-boot-stm32mp157a-dk1-trusted.elf Debug symbol file for U-Boot →
STM32MP157A-DK1
├─ u-boot-stm32mp157c-dk2-optee.elf Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-trusted.elf Debug symbol file for U-Boot →
STM32MP157C-DK2
├─ u-boot-stm32mp157c-ev1-optee.elf Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-EV1
├─ u-boot-stm32mp157c-ev1-trusted.elf Debug symbol file for U-Boot →
STM32MP157C-EV1
├─ vmlinux                Debug symbol file for Linux kernel
└─ [...]

```

```

└─ sources
├─ arm-openstlinux_weston-linux-gnueabi
│ └─ linux-5.10.61
│   ├── [*].patch          ST patches for Linux kernel
│   ├── fragment-[*].config ST configuration fragments for Linux kernel
│   ├── linux-5.10.61      Linux kernel source code directory
│   ├── linux-5.10.61.tar.xz
│   ├── README.HOW_TO.txt  Helper file for Linux kernel management: referenc
│   └─ series
code directory
└─ e for Linux kernel build

```

```

└─ optee-os-stm32mp-3.12.0-stm32mp-r2 OP-TEE OS installation directory
├─ [*].patch          ST patches for OP-TEE OS
├─ optee-os-stm32mp-3.12.0-stm32mp-r2
├─ Makefile.sdk       Makefile for the OP-TEE OS compilation
├─ optee-os-stm32mp-3.12.0-stm32mp-r2-r0.tar.gz OP-TEE OS source
code directory
├─ README.HOW_TO.txt  Helper file for OP-TEE OS management: reference
└─ series
for OP-TEE OS build

```

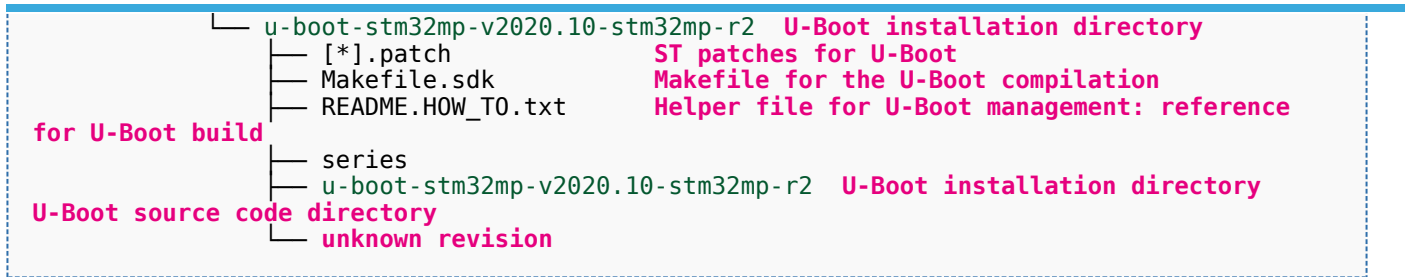
```

└─ tf-a-stm32mp-v2.4-stm32mp-r2 TF-A installation directory
├─ [*].patch          ST patches for TF-A
├─ tf-a-stm32mp-v2.4-stm32mp-r2 TF-A source code directory
├─ Makefile.sdk       Makefile for the TF-A compilation
├─ README.HOW_TO.txt  Helper file for TF-A management: reference
code directory
├─ series
└─ tf-a-stm32mp-v2.4-stm32mp-r2-r0.tar.gz
for TF-A build

```



Example of directory structure for Packages



Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.

```

Distribution-Package
├── openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution
│   └── layers
│       ├── meta-openembedded  Collection of layers for the OpenEmbedded-Core universe (Op
│       │                       enEmbedded standard)
│       │   └── [...]
│       └── meta-qt5           QT5 layer for OpenEmbedded (standard)
│           └── [...]

```

```

├── meta-st
│   ├── meta-st-openstlinux  STMicroelectronics layer that
│   │                       contains the settings of the frameworks and images for the OpenSTLinux distribution
│   │
│   ├── recipes-st
│   │   ├── images
│   │   │   └── st-image-core.bb  Core image for OpenSTLinux
│   │   └── st-image-weston.bb   Weston image with basic Wayland
│   │                               support for OpenSTLinux distribution: recommended setup
│   └── [...]
│       ├── packagegroups
│       │   └── [...]

```

```

├── meta-st-stm32mp  STMicroelectronics layer that contains
│   │               the description of the BSP for the STM32 MPU devices
│   │
│   ├── recipes-bsp
│   │   ├── alsa
│   │   │   └── [...]
│   │   └── drivers
│   │       └── [...]
│   │
│   ├── recipes-extended
│   │   ├── m4projects
│   │   │   └── [...]
│   │   └── [...]
│   │
│   ├── recipes-graphics
│   │   ├── gcnano-userland
│   │   │   └── [...]
│   │   └── [...]
│   │
│   ├── recipes-kernel
│   │   ├── linux
│   │   │   └── [...]
│   │   └── linux-firmware
│   │       └── [...]
│   │
│   └── [...]

```

drivers

within the OpenSTLinux distribution

ES, OpenVG and EGL (multi backend)

Bluetooth firmware)

Recipes for ALSA control configuration

Recipes for Vivante GCNANO GPU kernel

Recipes for TF-A

Recipes for U-Boot

Recipes for STM32Cube MPU Package

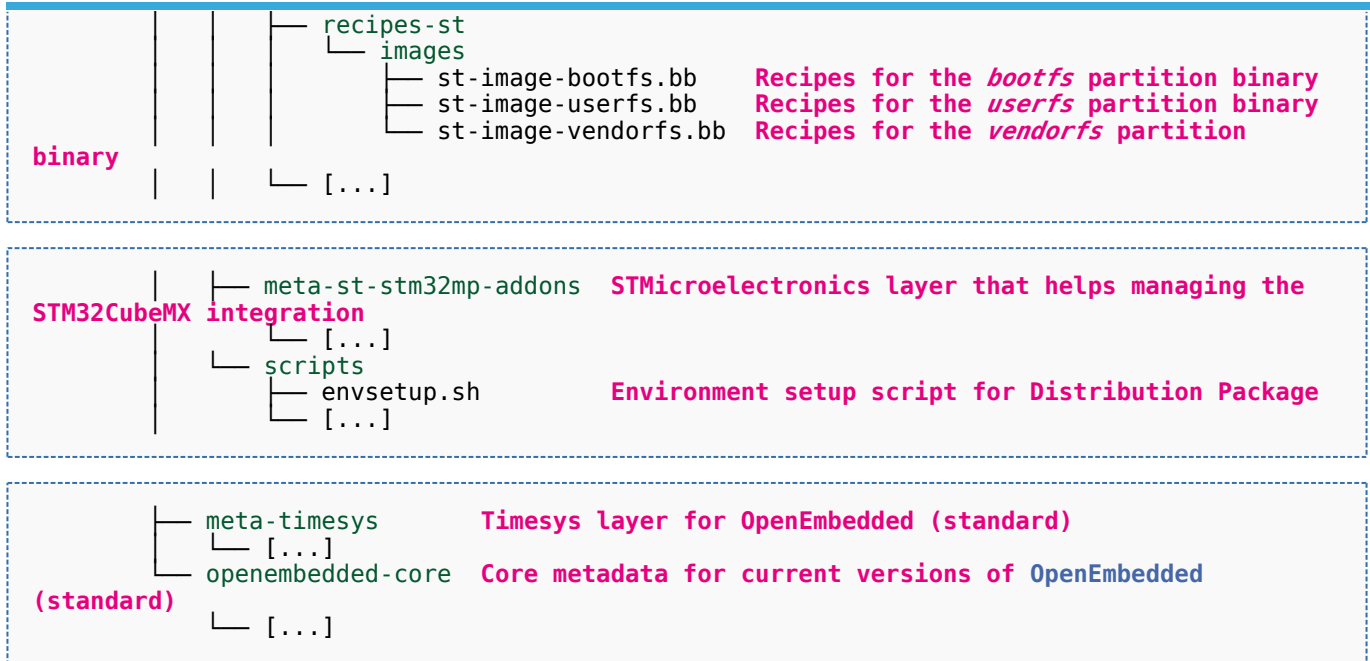
Recipes for Vivante libraries OpenGL

Recipes for Linux kernel

Recipes for Linux firmwares (example,



Example of directory structure for Packages

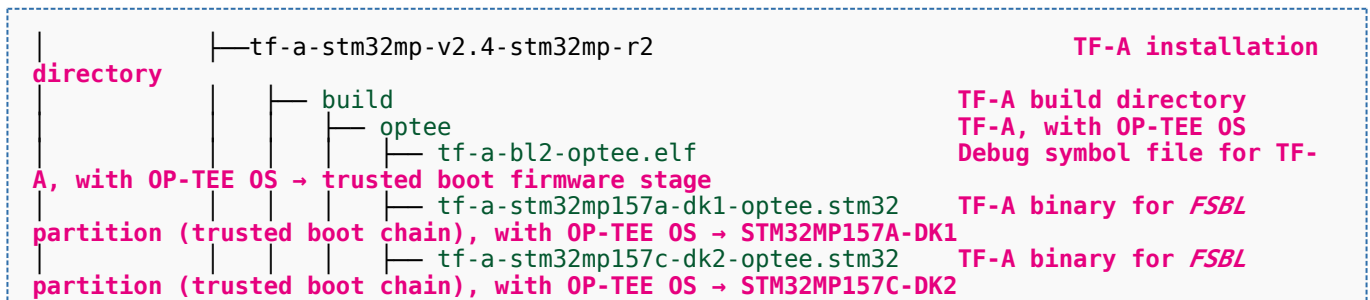
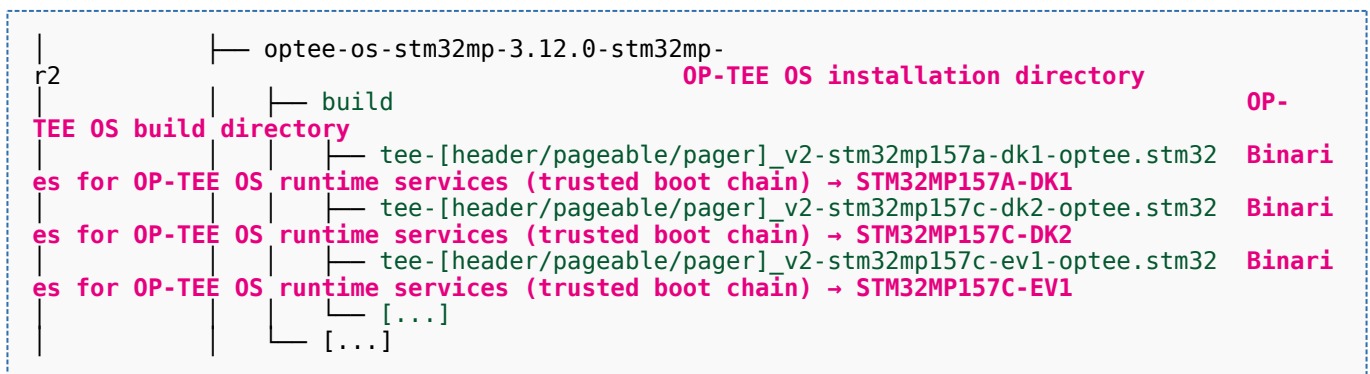
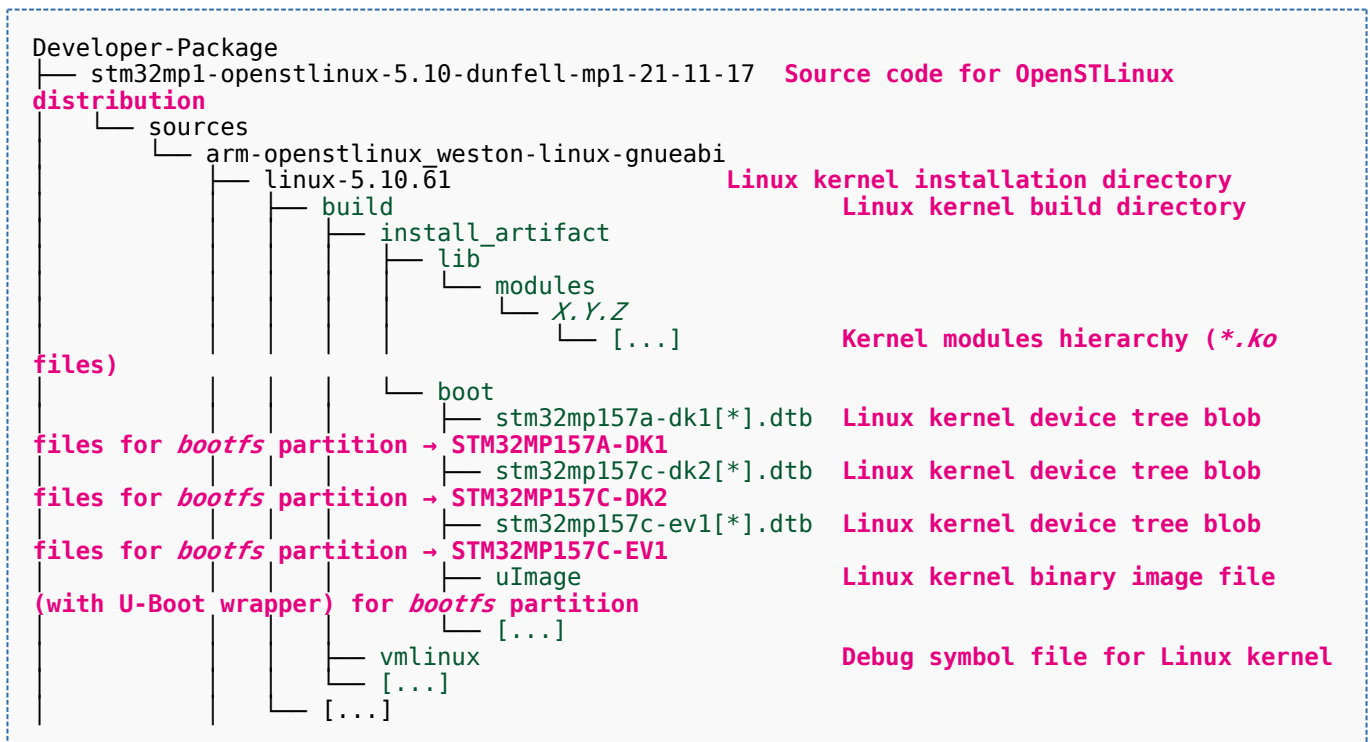


Appendix B shows the structure of the build directory.



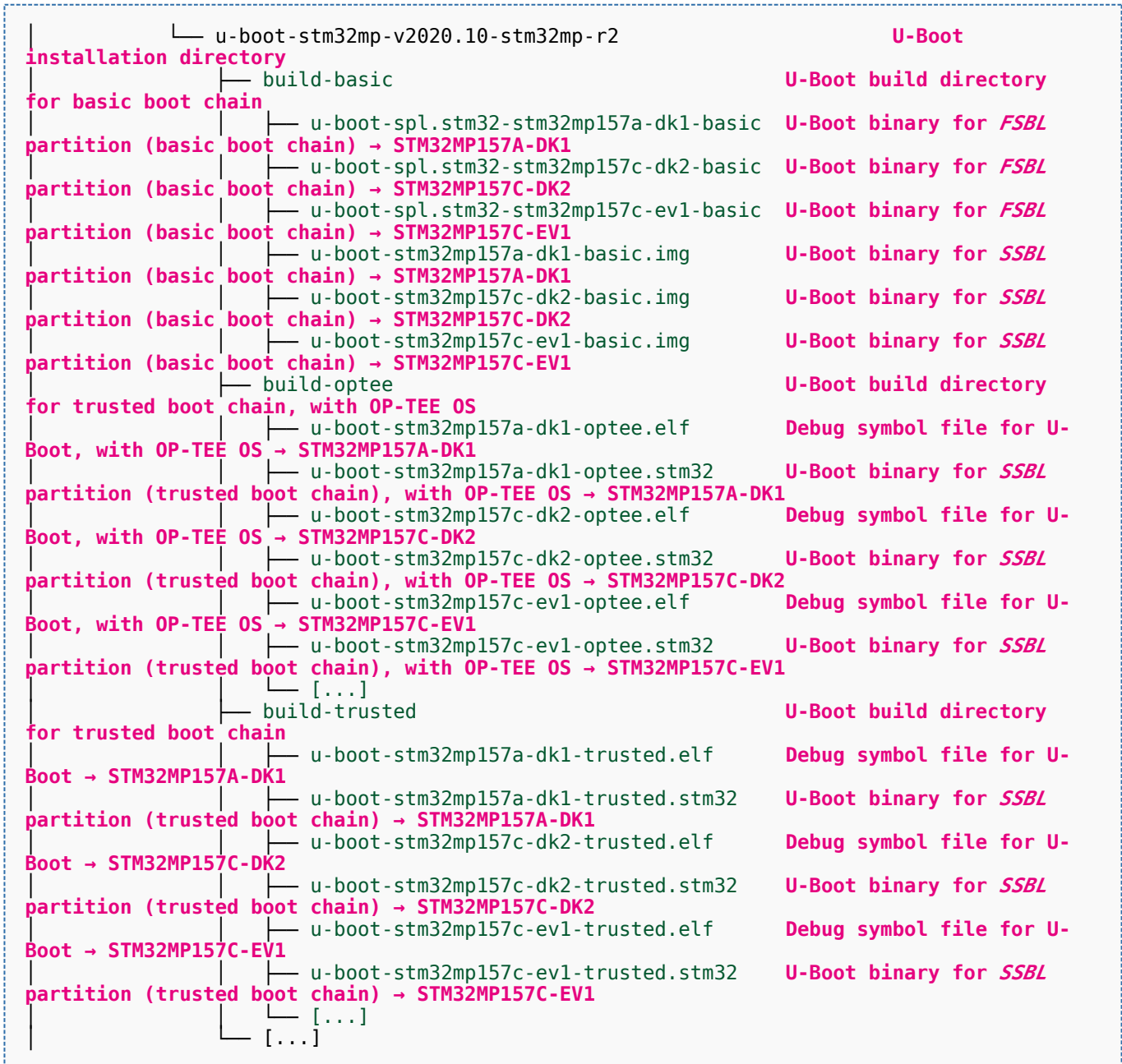
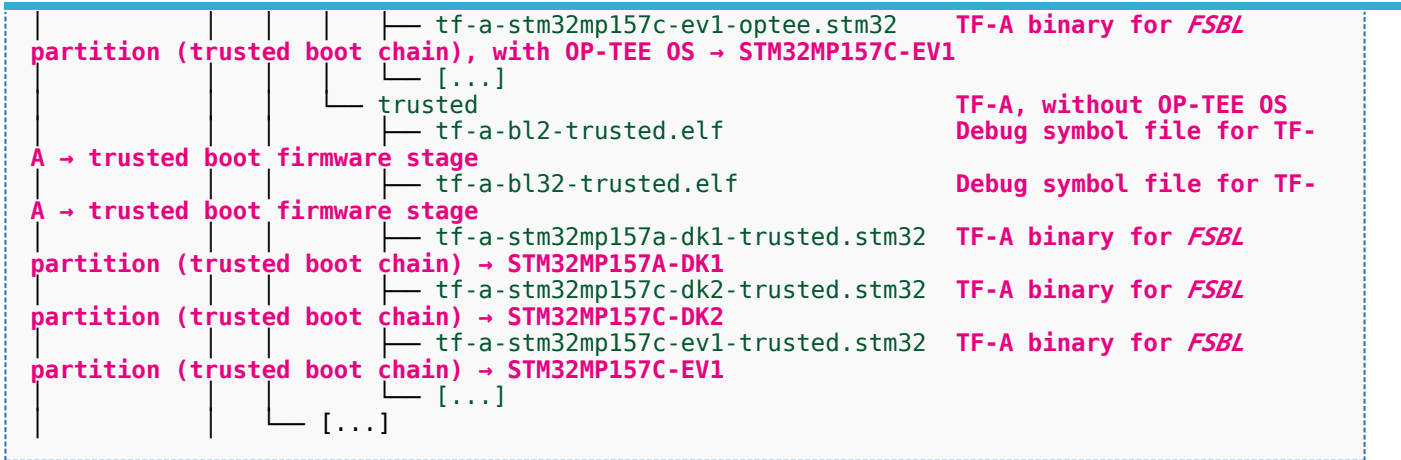
6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages





Example of directory structure for Packages





7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv       Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv    Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv  Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv        Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv        Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv      Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv        Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv        Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv      Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv        Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv        Flash layout file
```



Example of directory structure for Packages

```

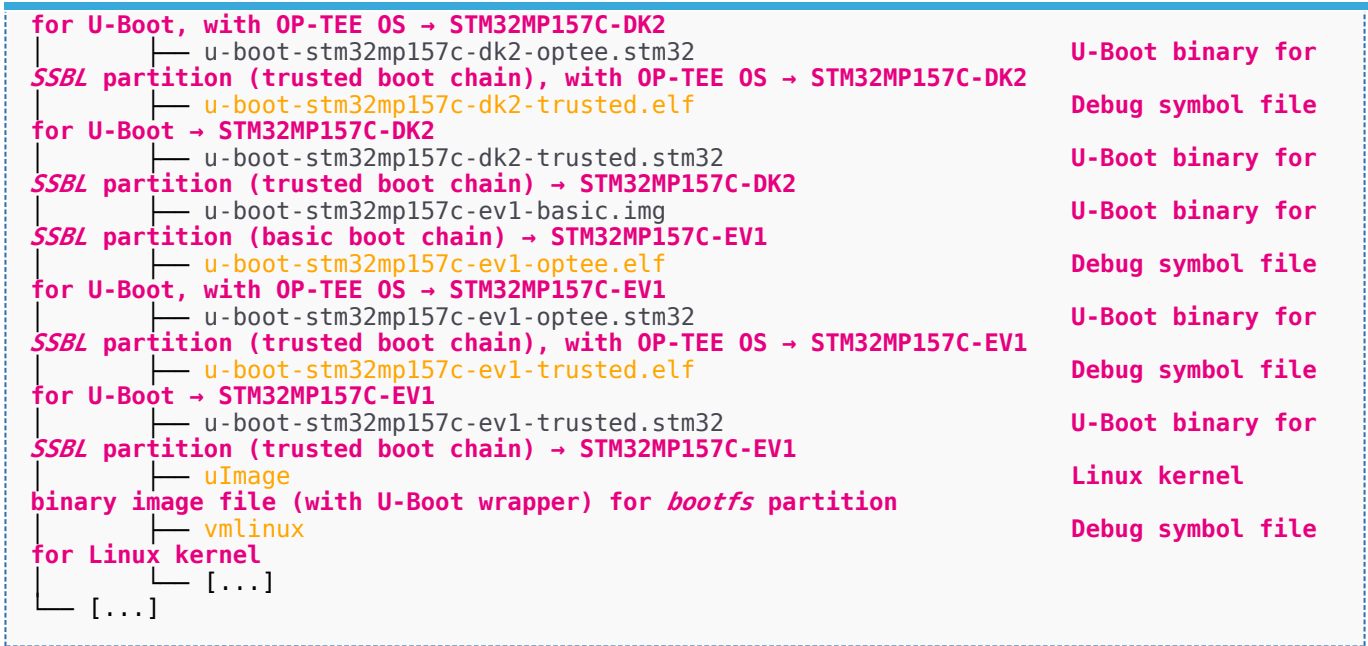
for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv      Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4      Binary for bootfs
partition └── st-image-userfs-openstlinux-weston-stm32mp1.ext4    Binary for userfs
s partition └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for vendorfs
partition └── st-image-weston-openstlinux-weston-stm32mp1.ext4    Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1
├── stm32mp157a-dk1[*].dtb                                          Linux kernel
device tree blob files for bootfs partition → STM32MP157C-DK2
├── stm32mp157c-dk2[*].dtb                                          Linux kernel
device tree blob files for bootfs partition → STM32MP157C-EV1
├── stm32mp157c-e[*].dtb                                           Linux kernel
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32    Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32    Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32    Binaries for OP-
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-optee.elf                                              Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                           Debug symbol file
for TF-A → runtime software stage
├── tf-a-bl32-trusted.elf                                           Debug symbol file
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-optee.stm32                                TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32                             TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                                TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                             TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                                TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                             TF-A binary for FS
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                          U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                          U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                          U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-basic.img                                 U-Boot binary for
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                                Debug symbol file
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157c-dk2-basic.img                                 U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                                Debug symbol file

```



Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 17.11.2021 - 16:17 / Revision: 10.11.2021 - 15:06

Contents

1 Article purpose	64
2 Creating the structure	65
3 Focus on the Starter Package directory	66
4 Focus on the Developer Package directory	68
5 Focus on the Distribution Package directory	71
6 Appendix A: directory structure after build (Developer Package)	73
7 Appendix B: directory structure after build (Distribution Package)	76



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0  STM32MPU Embedded Software release
├── Developer-Package      Developer Package installation directory
├── Distribution-Package   Distribution Package installation directory
└── Starter-Package       Starter Package installation directory
```

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0  STM32MPU Embedded Software release
├── Developer-Package      Developer Package installation
│   └── directory
│       ├── SDK           SDK for OpenSTLinux distribution
│       ├── STM32Cube_FW_MP1_V1.5.0  STM32CubeMP1 Package
│       └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  Linux kernel, U-Boot, TF-A and OP-TEE OS source code (OpenSTLinux distribution)
├── Distribution-Package   Distribution Package installation
│   └── directory
│       └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution (full source code and OpenEmbedded-based build framework)
└── Starter-Package       Starter Package installation
    └── directory
        └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17  Software image (binaries)
```



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│           ├── scripts
│           │   ├── create_sdcard_from_flashlayout.sh
│           │   └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│           └── tfs partition
│               ├── st-image-bootfs-openstlinux-weston-stm32mp1.manifest
│               ├── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│               └── rfs partition
│                   ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│                   ├── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│                   └── dorfs partition
│                       ├── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│                       └── tfs partition
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.license
│                           ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│                           ├── [...]
│                           └── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│                               └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│                                   ├── [...]
│                                   └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│                                       └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│                                           ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│                                           └── for FSBL partition (trusted boot chain) → STM32MP157C-DK2
│                                               ├── [...]
│                                               └── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│                                                   └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│                                                       ├── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│                                                       └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

```
├─ u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└─ [...]
```



4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)

```

Developer-Package
├── SDK
├── environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
├── site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
├── sysroots
│   ├── cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│   ├── x86_64-ostl_sdk-linux
│   └── [...]
├── version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
└── [...]
  
```

for OpenSTLinux distribution: details in Standard SDK directory structure article

script for Developer Package

SDK

Environment setup

Target sysroot

Native sysroot

(libraries, headers, and symbols)

(libraries, headers, and symbols)

```

├── STM32Cube_FW_MP1_V1.5.0
├── Drivers
│   ├── BSP
│   ├── CMSIS
│   └── STM32MP1xx_HAL_Driver
├── htmresc
├── License.md
├── Middlewares
├── package.xml
├── Projects
│   ├── STM32CubeProjectsList.html
│   ├── STM32MP157C-DK2
│   └── STM32MP157C-EV1
├── Readme.md
├── Release_Notes.html
├── Utilities
└── [...]
  
```

STM32CubeMP1 Package: details in STM32CubeMP1 Package content article

BSP drivers for the supported STM32MPU boards

HAL drivers for the supported STM32MPU devices

License types for the components

Middlewares used by the examples and applications

STM32CubeMP1 Package

List of examples and applications for

Set of examples and applications → STM32MP157C-DK2

Set of examples and applications → STM32MP157C-EV1

Release note for STM32CubeMP1 Package



```

└─ stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
distribution
├─ images
│ └─ stm32mp1
directory
├─ tf-a-bl2-optee.elf      Debug symbol files installation
TEE OS → trusted boot firmware stage
├─ tf-a-bl2-trusted.elf   Debug symbol file for TF-A, with OP-
boot firmware stage
├─ tf-a-bl32-trusted.elf  Debug symbol file for TF-A → trusted
software stage
├─ u-boot-stm32mp157a-dk1-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157A-DK1
├─ u-boot-stm32mp157a-dk1-trusted.elf  Debug symbol file for U-Boot →
STM32MP157A-DK1
├─ u-boot-stm32mp157c-dk2-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-trusted.elf  Debug symbol file for U-Boot →
STM32MP157C-DK2
├─ u-boot-stm32mp157c-ev1-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-EV1
├─ u-boot-stm32mp157c-ev1-trusted.elf  Debug symbol file for U-Boot →
STM32MP157C-EV1
├─ vmlinux
└─ [...]

```

```

└─ sources
├─ arm-openstlinux_weston-linux-gnueabi
│ └─ linux-5.10.61
│   ├── [*].patch      ST patches for Linux kernel
│   ├── fragment-[*].config  ST configuration fragments for Linux kernel
│   ├── linux-5.10.61  Linux kernel source code directory
│   ├── linux-5.10.61.tar.xz
│   ├── README.HOW_TO.txt  Helper file for Linux kernel management: referenc
│   └─ series
code directory
└─ for Linux kernel build

```

```

└─ optee-os-stm32mp-3.12.0-stm32mp-r2  OP-TEE OS installation directory
│ ├── [*].patch      ST patches for OP-TEE OS
│ ├── optee-os-stm32mp-3.12.0-stm32mp-r2
│ ├── Makefile.sdk   Makefile for the OP-TEE OS compilation
│ └─ optee-os-stm32mp-3.12.0-stm32mp-r2-r0.tar.gz  OP-TEE OS source
code directory
└─ for OP-TEE OS build
├─ README.HOW_TO.txt  Helper file for OP-TEE OS management: reference
└─ series

```

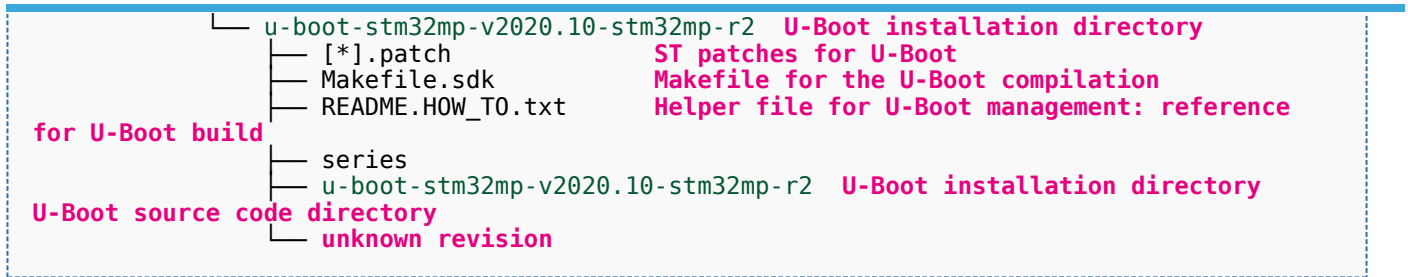
```

└─ tf-a-stm32mp-v2.4-stm32mp-r2  TF-A installation directory
│ ├── [*].patch      ST patches for TF-A
│ ├── tf-a-stm32mp-v2.4-stm32mp-r2  TF-A source code directory
│ ├── Makefile.sdk   Makefile for the TF-A compilation
│ └─ README.HOW_TO.txt  Helper file for TF-A management: reference
code directory
└─ for TF-A build
├─ series
└─ tf-a-stm32mp-v2.4-stm32mp-r2-r0.tar.gz

```



Example of directory structure for Packages



Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.

```

Distribution-Package
├── openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution
│   └── layers
│       ├── meta-openembedded  Collection of layers for the OpenEmbedded-Core universe (Op
OpenEmbedded standard)
│       │   └── [...]
│       └── meta-qt5          QT5 layer for OpenEmbedded (standard)
│           └── [...]

```

```

├── meta-st
│   ├── meta-st-openstlinux  STMicroelectronics layer that
contains the settings of the frameworks and images for the OpenSTLinux distribution
│   ├── recipes-st
│   │   ├── images
│   │   │   └── st-image-core.bb  Core image for OpenSTLinux
│   │   └── st-image-weston.bb  Weston image with basic Wayland
│   │                               support for OpenSTLinux distribution: recommended setup
│   └── [...]
│       ├── packagegroups
│       │   └── [...]

```

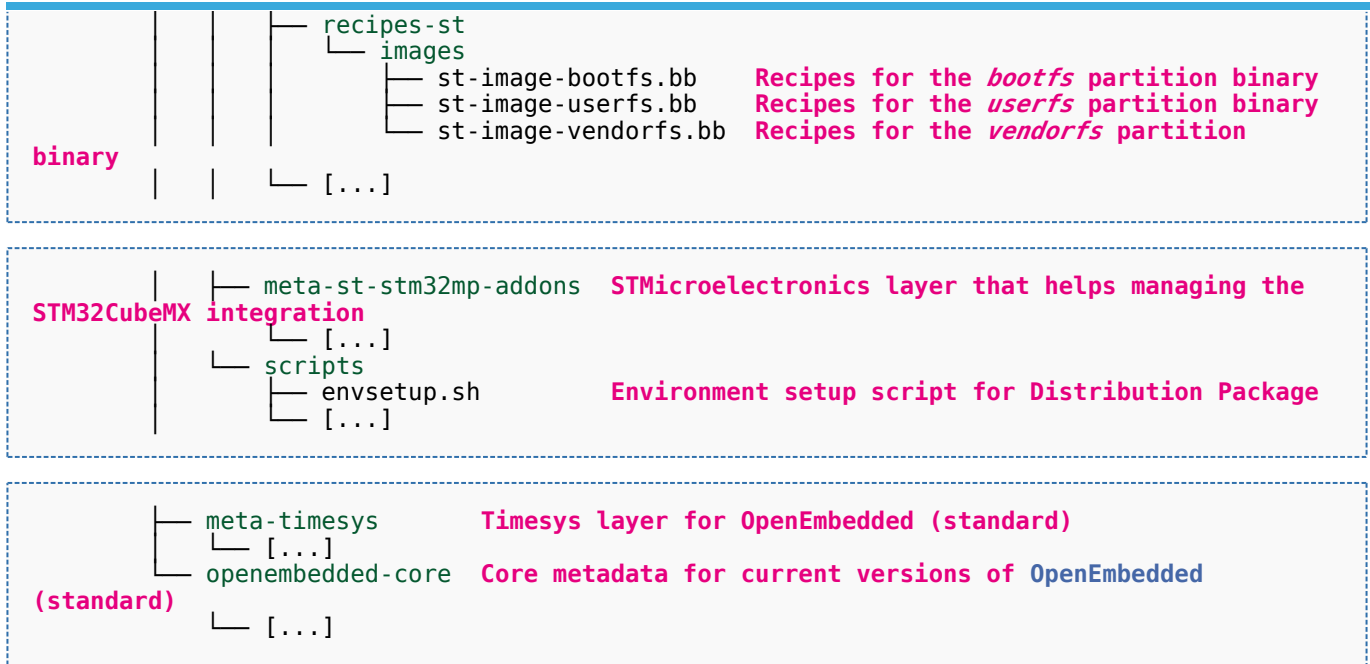
```

├── meta-st-stm32mp  STMicroelectronics layer that contains
the description of the BSP for the STM32 MPU devices
├── recipes-bsp
│   ├── alsa  Recipes for ALSA control configuration
│   └── drivers  Recipes for Vivante GCNANO GPU kernel
├── [...]
├── trusted-firmware-a  Recipes for TF-A
├── u-boot  Recipes for U-Boot
├── recipes-extended  Recipes for STM32Cube MPU Package
├── m4projects
│   ├── [...]
│   └── [...]
├── recipes-graphics  Recipes for Vivante libraries OpenGL
├── gcnano-userland
│   ├── [...]
│   └── [...]
├── recipes-kernel  Recipes for Linux kernel
├── linux
│   ├── [...]
├── linux-firmware  Recipes for Linux firmwares (example,
Bluetooth firmware)
└── [...]

```



Example of directory structure for Packages

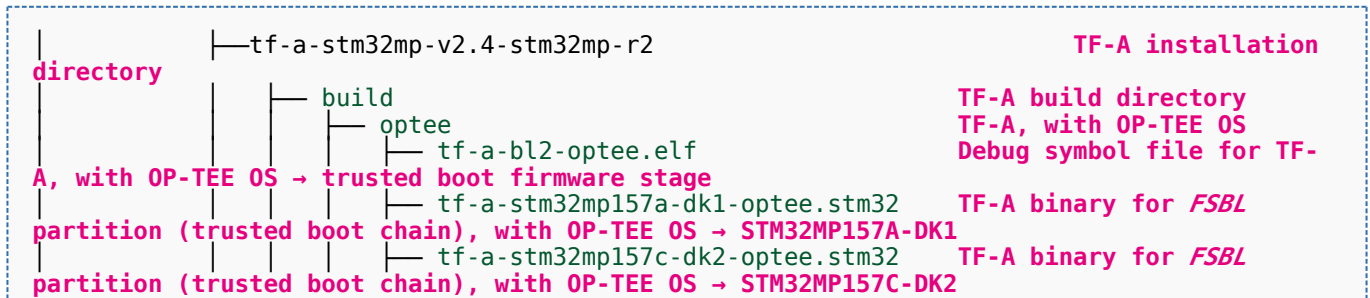
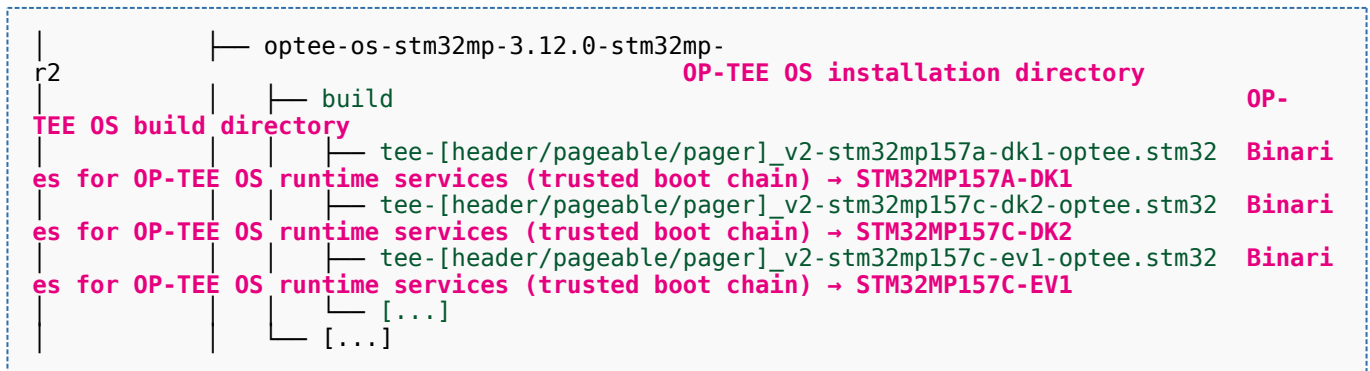
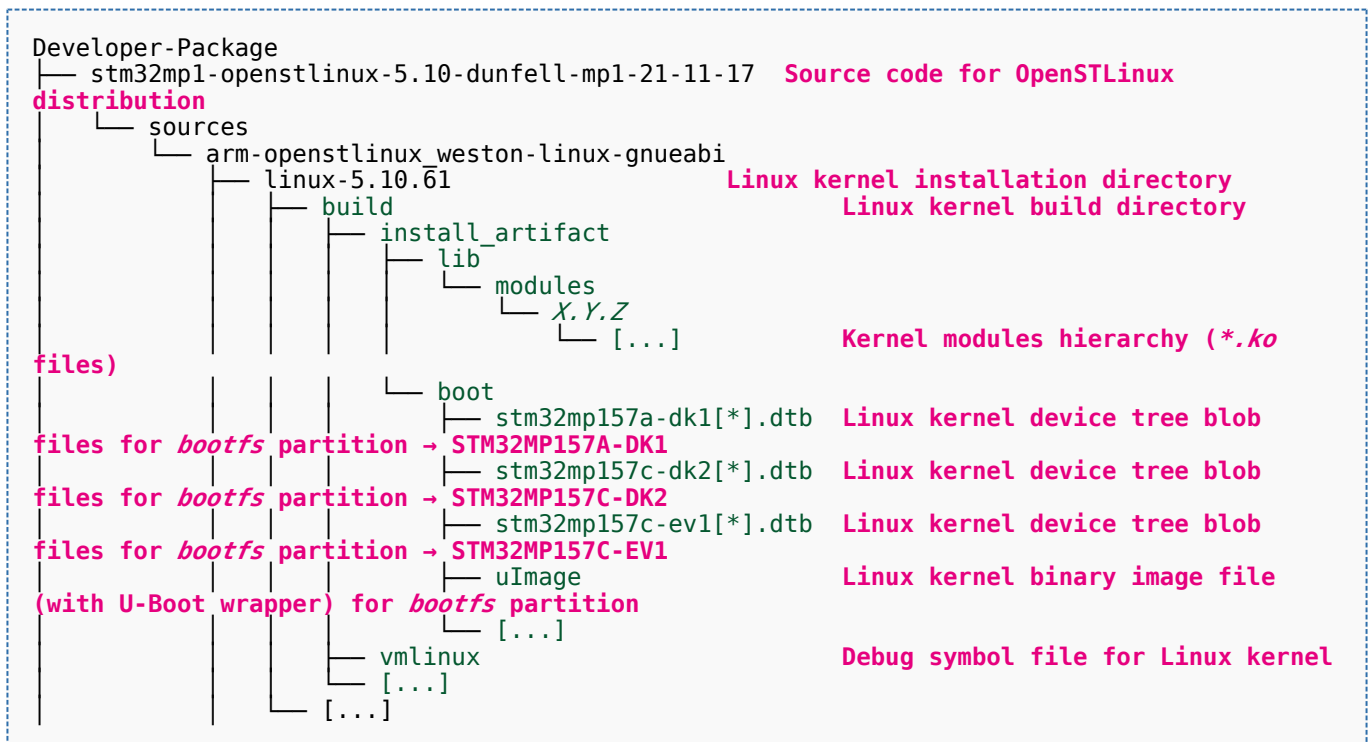


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages

partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-optee.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	trusted	TF-A, without OP-TEE OS
A → trusted boot firmware stage	tf-a-bl2-trusted.elf	Debug symbol file for TF-
A → trusted boot firmware stage	tf-a-bl32-trusted.elf	Debug symbol file for TF-
partition (trusted boot chain) → STM32MP157A-DK1	tf-a-stm32mp157a-dk1-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	tf-a-stm32mp157c-dk2-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-trusted.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	[...]	

installation directory	u-boot-stm32mp-v2020.10-stm32mp-r2	U-Boot
for basic boot chain	build-basic	U-Boot build directory
partition (basic boot chain) → STM32MP157A-DK1	u-boot-spl.stm32-stm32mp157a-dk1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-spl.stm32-stm32mp157c-dk2-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-spl.stm32-stm32mp157c-ev1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for <i>SSBL</i>
for trusted boot chain, with OP-TEE OS	build-optee	U-Boot build directory
Boot, with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
for trusted boot chain	build-trusted	U-Boot build directory
Boot → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
	[...]	



Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv       Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv    Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv  Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv        Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv        Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv      Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv        Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv        Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv      Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv        Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv        Flash layout file
```



Example of directory structure for Packages

```

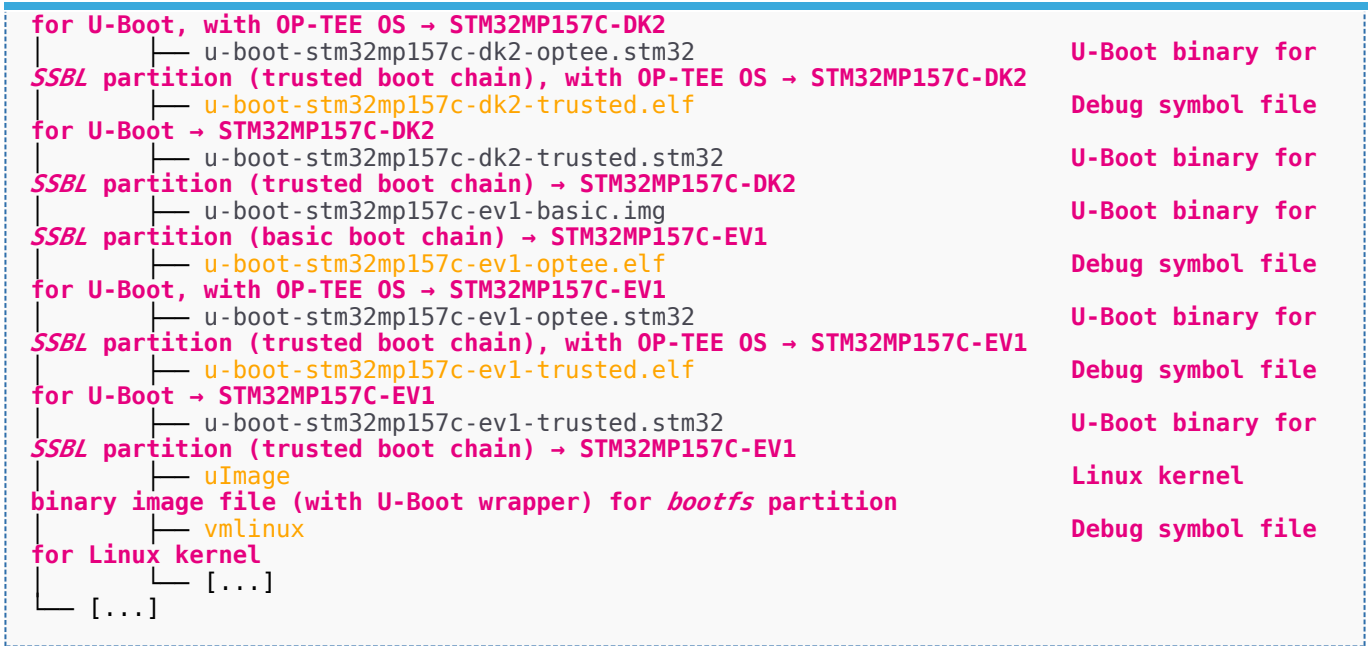
for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv      Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4      Binary for bootfs
partition └── st-image-userfs-openstlinux-weston-stm32mp1.ext4    Binary for userfs
s partition └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for vendorfs
partition └── st-image-weston-openstlinux-weston-stm32mp1.ext4    Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1
├── stm32mp157a-dk1[*].dtb                                          Linux kernel
device tree blob files for bootfs partition → STM32MP157C-DK2
├── stm32mp157c-dk2[*].dtb                                          Linux kernel
device tree blob files for bootfs partition → STM32MP157C-EV1
├── stm32mp157c-e[*].dtb                                           Linux kernel
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32    Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32    Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32    Binaries for OP-
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-optee.elf                                              Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                           Debug symbol file
for TF-A → runtime software stage
├── tf-a-bl32-trusted.elf                                           Debug symbol file
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-optee.stm32                                TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32                             TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                                TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                             TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                                TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                             TF-A binary for FS
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                          U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                          U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                          U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-basic.img                                 U-Boot binary for
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                                Debug symbol file
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157c-dk2-basic.img                                 U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                                Debug symbol file

```



Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 17.11.2021 - 16:45 / Revision: 17.11.2021 - 12:44

Contents

1 Article purpose	79
2 Creating the structure	80
3 Focus on the Starter Package directory	81
4 Focus on the Developer Package directory	83
5 Focus on the Distribution Package directory	86
6 Appendix A: directory structure after build (Developer Package)	88
7 Appendix B: directory structure after build (Distribution Package)	91



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
├── Distribution-Package
└── Starter-Package
```

STM32MPU Embedded Software release
Developer Package installation directory
Distribution Package installation directory
Starter Package installation directory

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
│   ├── SDK
│   ├── STM32Cube_FW_MP1_V1.5.0
│   └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│       ├── TEE OS source code (OpenSTLinux distribution)
│       └── Distribution-Package
│           ├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│           └── Starter-Package
│               ├── source code and OpenEmbedded-based build framework
│               └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│                   ├── OpenSTLinux distribution (full
│                   │   source code and OpenEmbedded-based build framework)
│                   └── Starter Package installation
│                       ├── Software image (binaries)
```

STM32MPU Embedded Software release
Developer Package installation
SDK for OpenSTLinux distribution
STM32CubeMP1 Package
Linux kernel, U-Boot, TF-A and OP-
TEE OS source code (OpenSTLinux distribution)
Distribution Package installation
OpenSTLinux distribution (full
source code and OpenEmbedded-based build framework)
Starter Package installation
Software image (binaries)



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard
│               │       size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│           ├── scripts
│           │   └── create_sdcard_from_flashlayout.sh
│           ├── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│           │   └── tfs partition
│           ├── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│           │   └── rfs partition
│           ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│           ├── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│           │   └── dorfs partition
│           ├── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│           │   └── tfs partition
│           ├── st-image-weston-openstlinux-weston-stm32mp1.license
│           ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│           ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│           ├── [...]
│           ├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│           │   └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│           │       ├── [...]
│           │       └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│           ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│           │   └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│           ├── [...]
│           ├── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│           │   └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│           ├── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│           │   └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

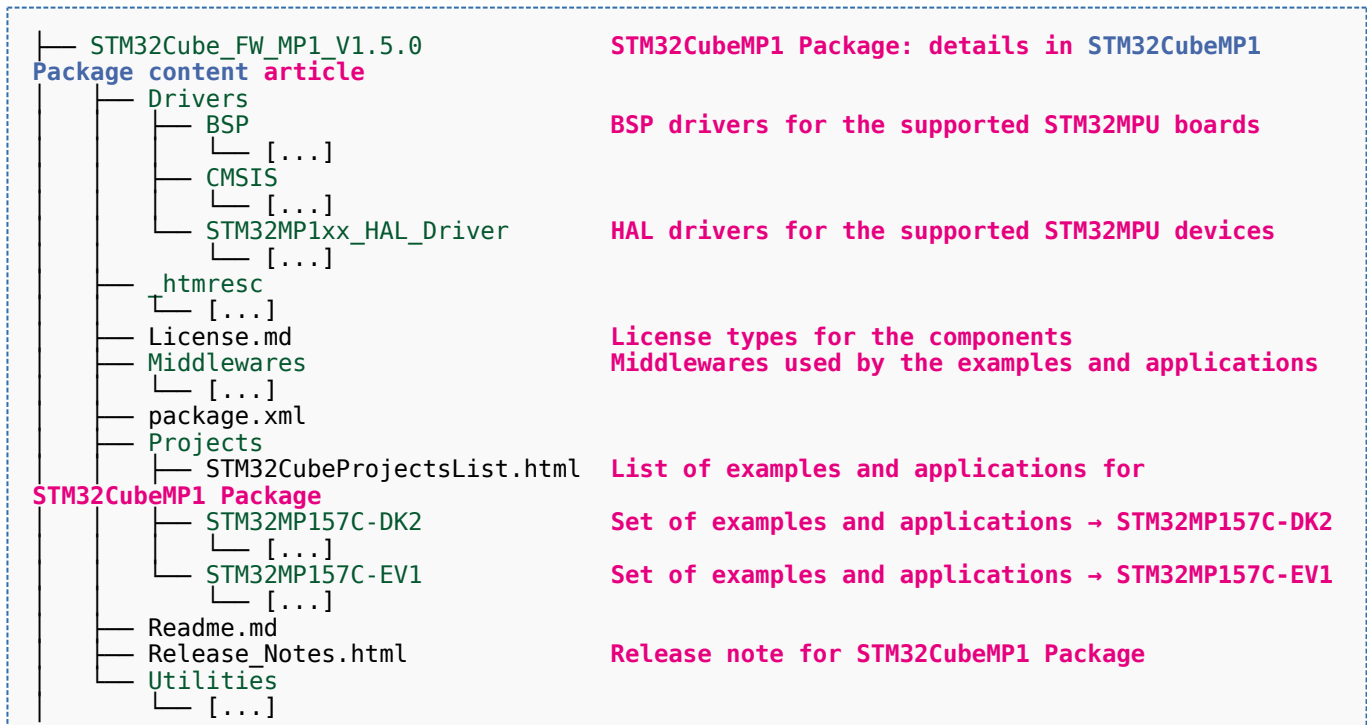
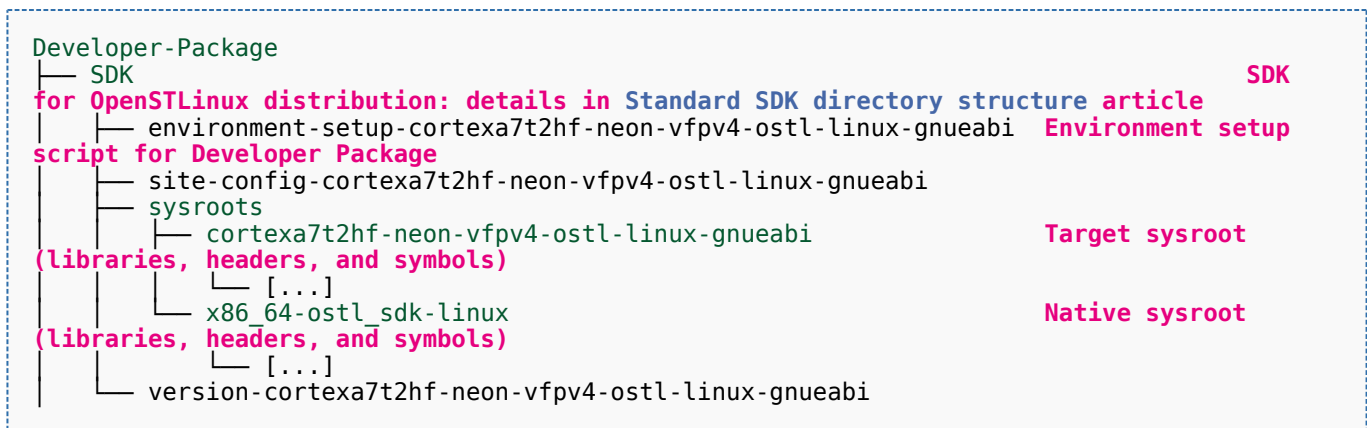
```
├── u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└── [...]
```

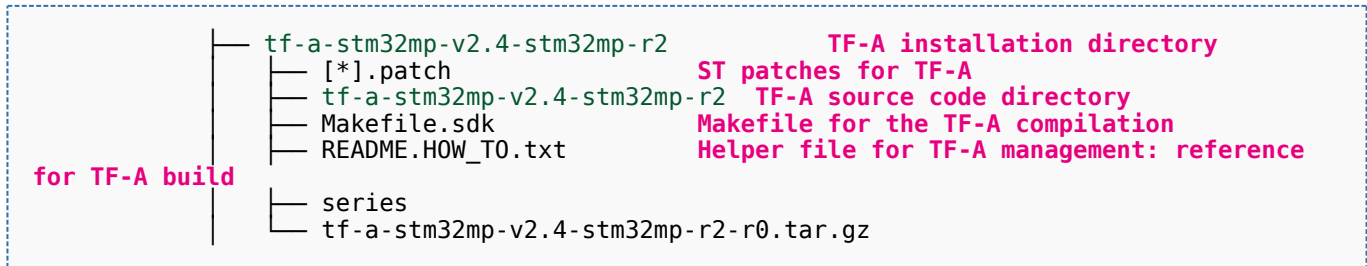
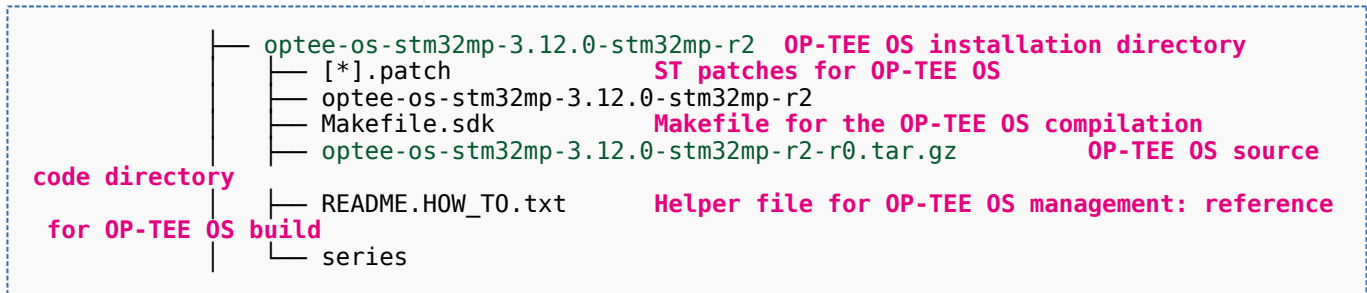
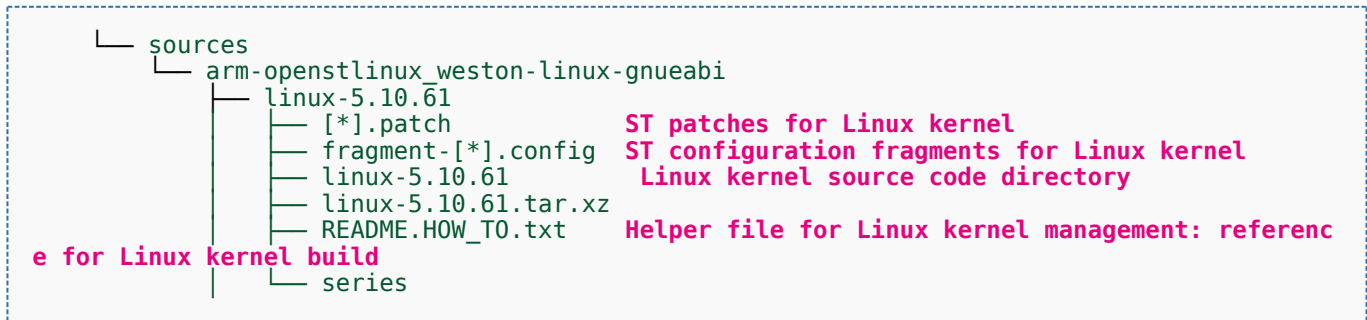
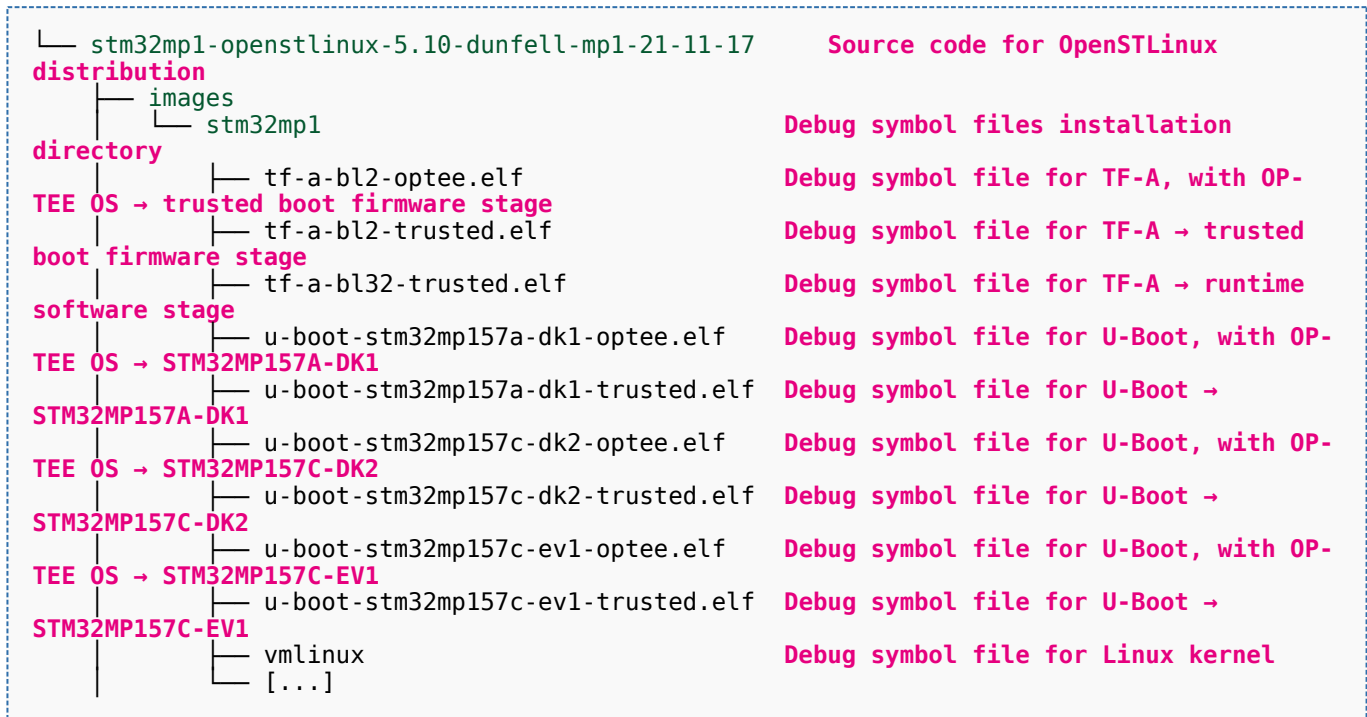


4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

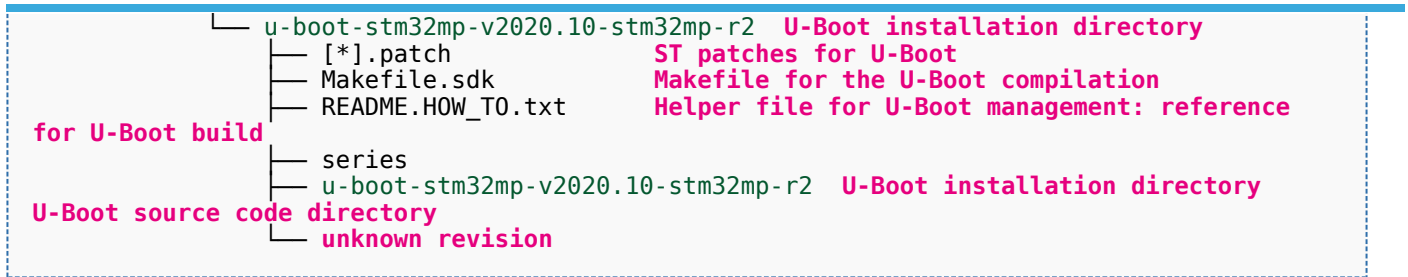
- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)







Example of directory structure for Packages

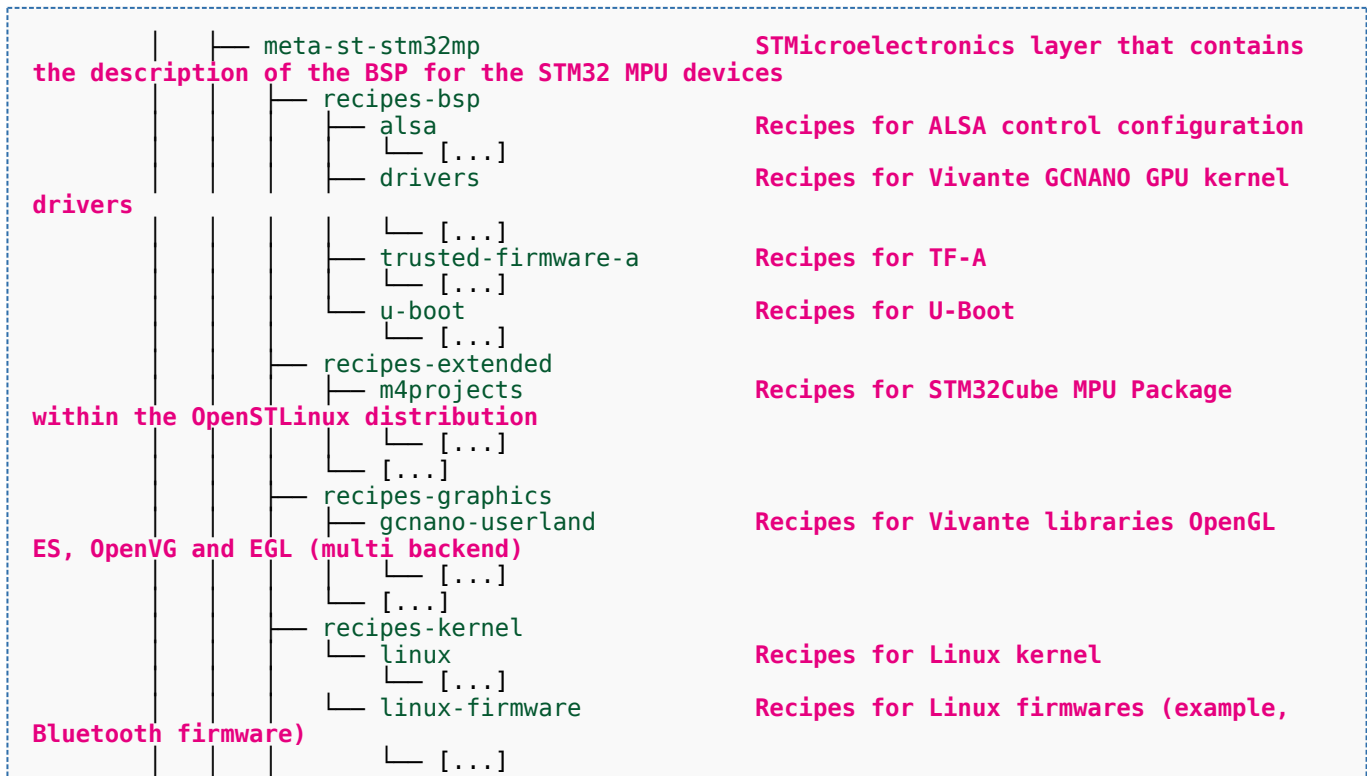
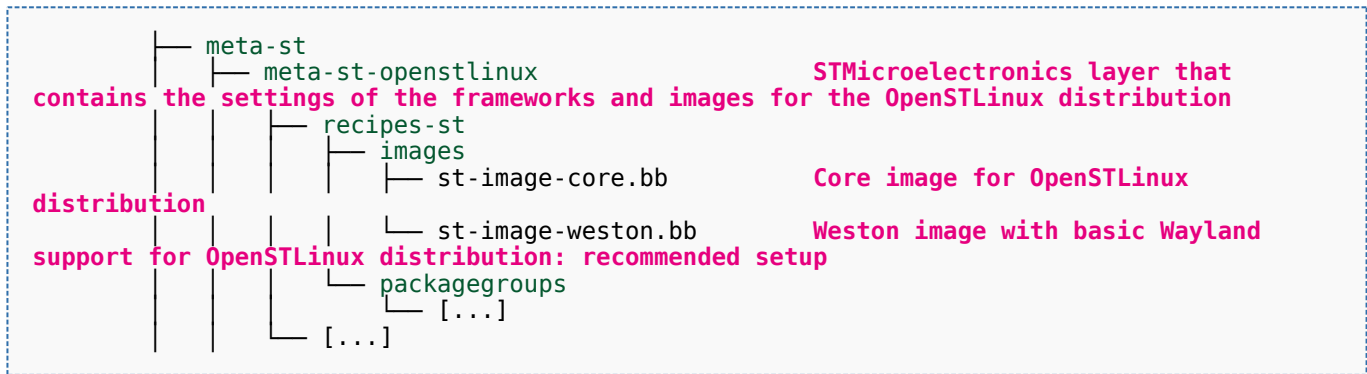
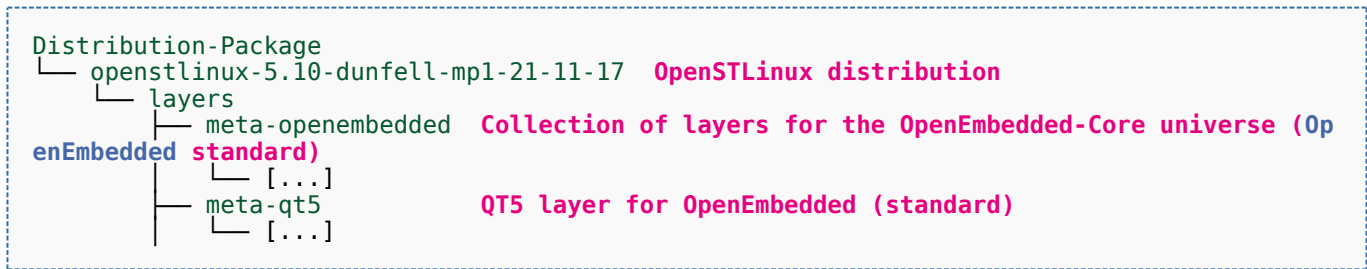


Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



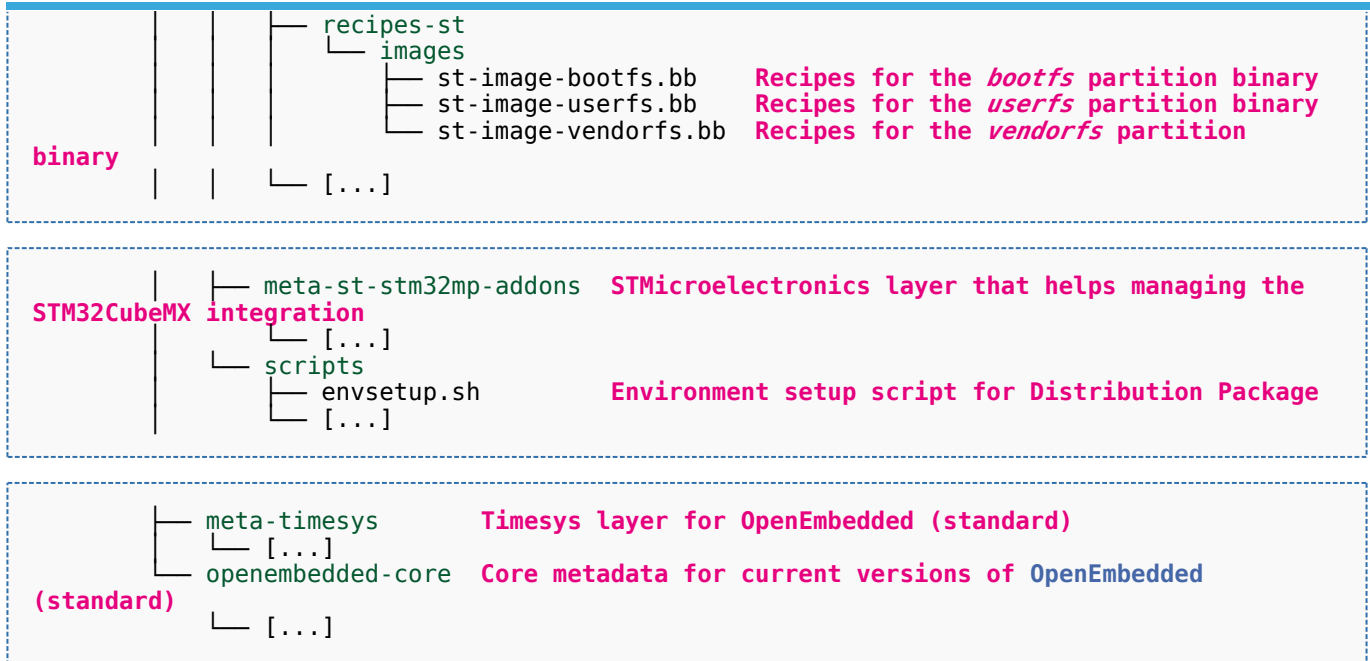
5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.





Example of directory structure for Packages

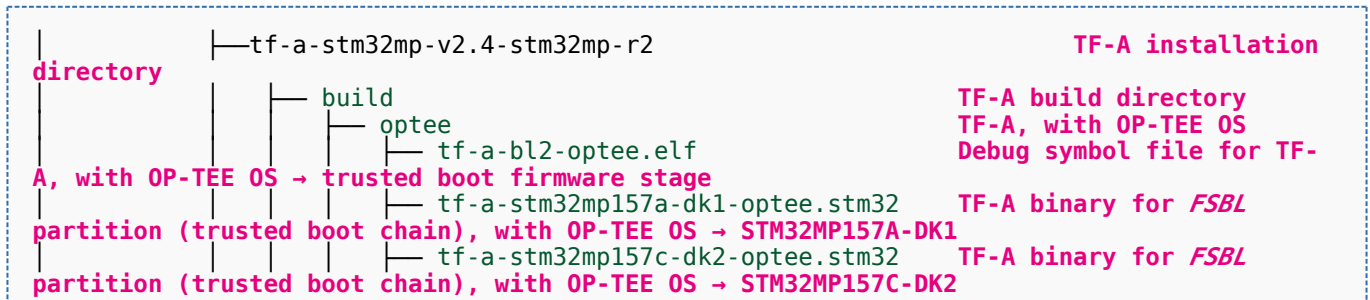
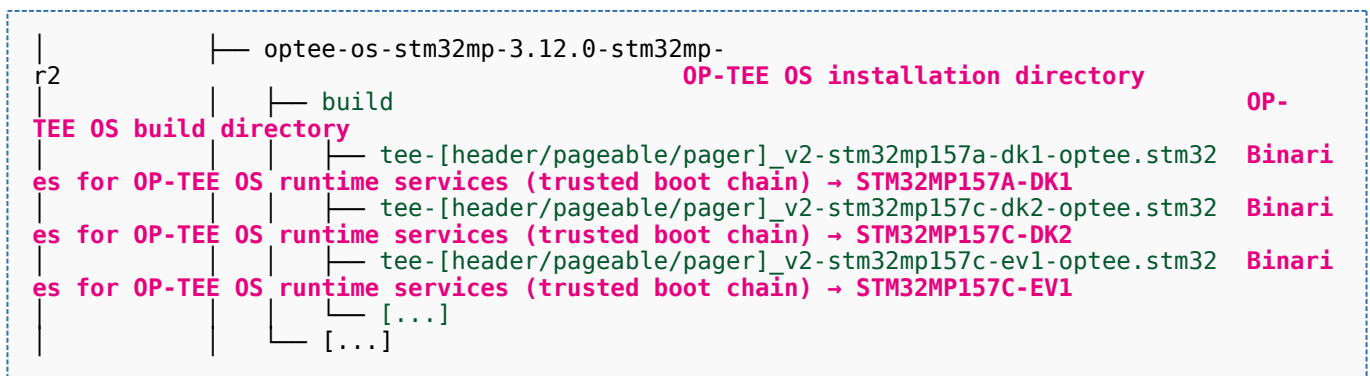
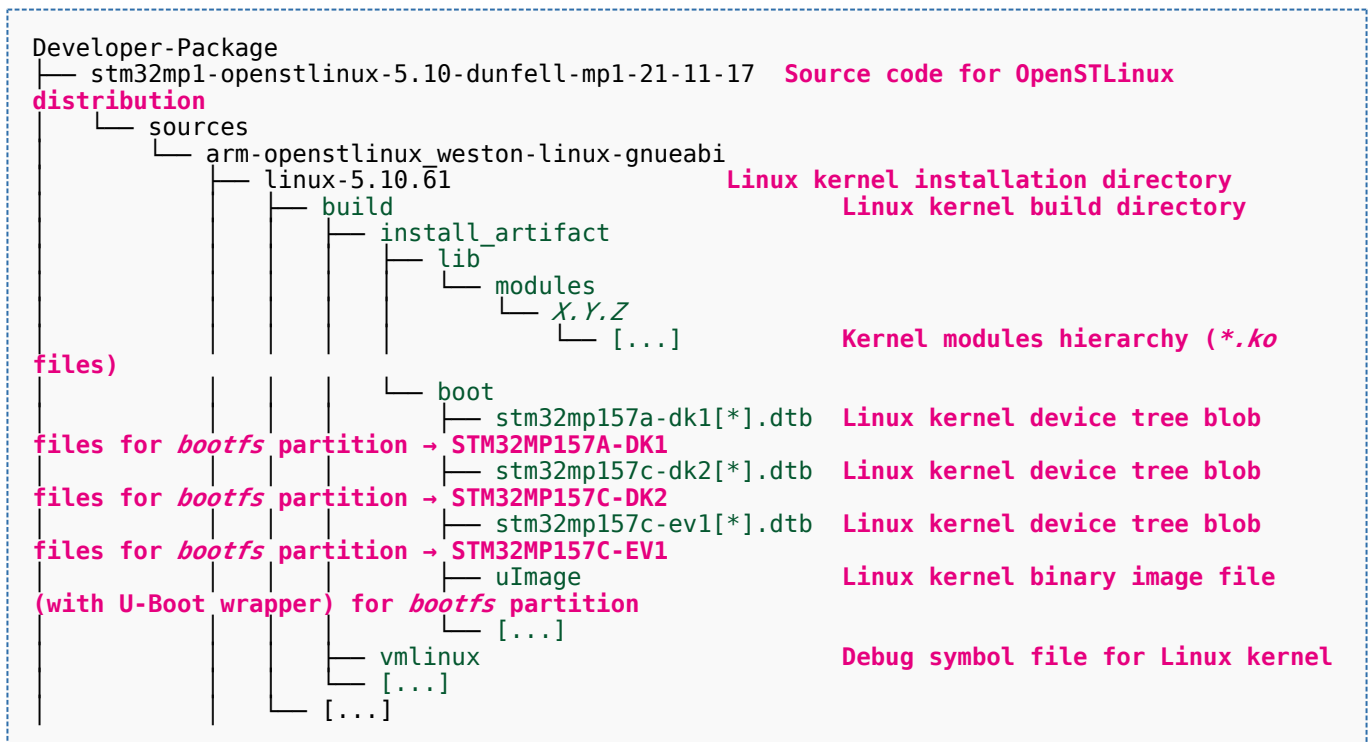


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages

partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-optee.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	trusted	TF-A, without OP-TEE OS
A → trusted boot firmware stage	tf-a-bl2-trusted.elf	Debug symbol file for TF-
A → trusted boot firmware stage	tf-a-bl32-trusted.elf	Debug symbol file for TF-
partition (trusted boot chain) → STM32MP157A-DK1	tf-a-stm32mp157a-dk1-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	tf-a-stm32mp157c-dk2-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-trusted.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	[...]	

	u-boot-stm32mp-v2020.10-stm32mp-r2	U-Boot
installation directory	build-basic	U-Boot build directory
for basic boot chain	u-boot-spl.stm32-stm32mp157a-dk1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-spl.stm32-stm32mp157c-dk2-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-spl.stm32-stm32mp157c-ev1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-stm32mp157a-dk1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-stm32mp157c-dk2-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	build-optee	U-Boot build directory
for trusted boot chain, with OP-TEE OS	u-boot-stm32mp157a-dk1-optee.elf	Debug symbol file for U-
Boot, with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.stm32	U-Boot binary for <i>SSBL</i>
partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157c-dk2-optee.elf	Debug symbol file for U-
Boot, with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for <i>SSBL</i>
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file for U-
Boot, with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for <i>SSBL</i>
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	[...]	
	build-trusted	U-Boot build directory
for trusted boot chain	u-boot-stm32mp157a-dk1-trusted.elf	Debug symbol file for U-
Boot → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.stm32	U-Boot binary for <i>SSBL</i>
partition (trusted boot chain) → STM32MP157A-DK1	u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file for U-
Boot → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for <i>SSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file for U-
Boot → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for <i>SSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	[...]	
	[...]	



Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv       Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv       Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv      Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv    Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv      Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv     Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv    Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv      Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv      Flash layout file
```



Example of directory structure for Packages

```

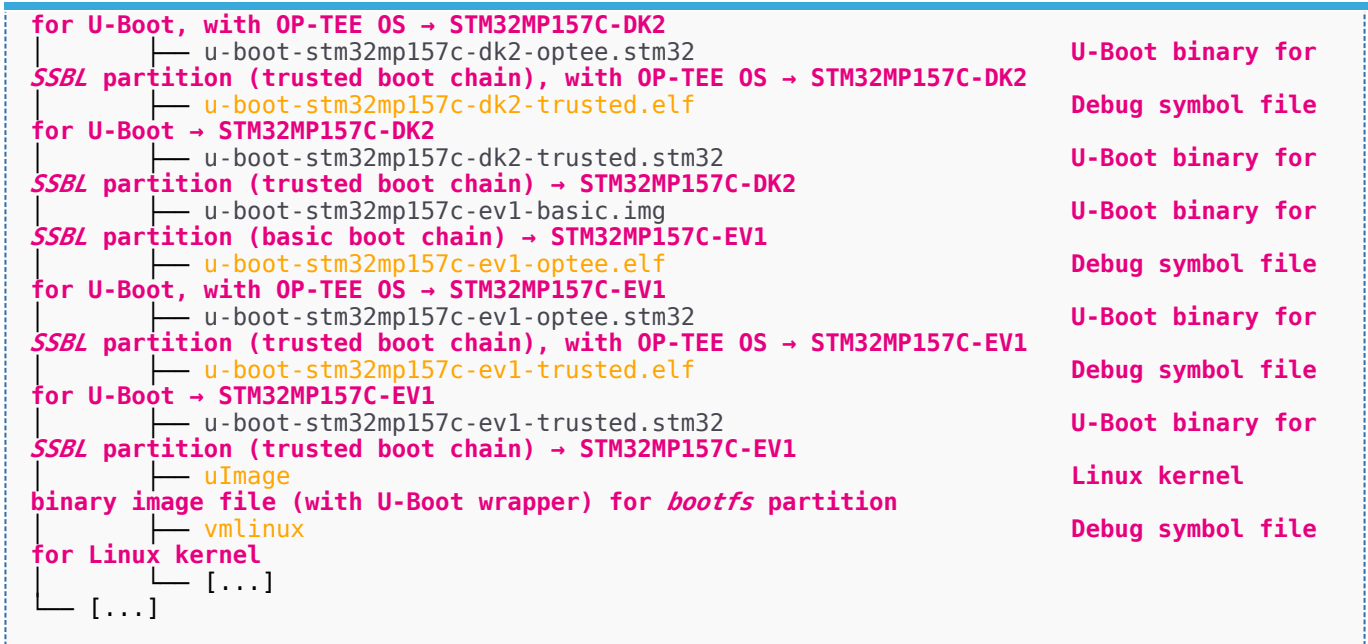
for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv      Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4      Binary for bootfs
partition └── st-image-userfs-openstlinux-weston-stm32mp1.ext4    Binary for userfs
s partition └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for vendorfs
partition └── st-image-weston-openstlinux-weston-stm32mp1.ext4   Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1
├── stm32mp157a-dk1[*].dtb                                         Linux kernel
device tree blob files for bootfs partition → STM32MP157C-DK2
├── stm32mp157c-dk2[*].dtb                                         Linux kernel
device tree blob files for bootfs partition → STM32MP157C-EV1
├── stm32mp157c-e[*].dtb                                           Linux kernel
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32   Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32   Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32   Binaries for OP-
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-optee.elf                                             Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                           Debug symbol file
for TF-A → runtime software stage
├── tf-a-bl32-trusted.elf                                           Debug symbol file
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-optee.stm32                               TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32                            TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                               TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                            TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                               TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                            TF-A binary for FS
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                         U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                         U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                         U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-basic.img                                U-Boot binary for
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                               Debug symbol file
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                           U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157c-dk2-basic.img                                U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                               Debug symbol file

```



Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 26.03.2021 - 11:22 / Revision: 26.03.2021 - 11:20

Contents

1 Article purpose	94
2 Creating the structure	95
3 Focus on the Starter Package directory	96
4 Focus on the Developer Package directory	98
5 Focus on the Distribution Package directory	101
6 Appendix A: directory structure after build (Developer Package)	103
7 Appendix B: directory structure after build (Distribution Package)	106



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
├── Distribution-Package
└── Starter-Package
```

STM32MPU Embedded Software release
Developer Package installation directory
Distribution Package installation directory
Starter Package installation directory

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
│   ├── SDK
│   ├── STM32Cube_FW_MP1_V1.5.0
│   └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│       ├── TEE OS source code (OpenSTLinux distribution)
│       └── Distribution-Package
│           ├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│           └── Starter-Package
│               ├── source code and OpenEmbedded-based build framework
│               └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│                   ├── OpenSTLinux distribution (full
│                   │   source code and OpenEmbedded-based build framework)
│                   └── Starter Package installation
│                       ├── Software image (binaries)
```

STM32MPU Embedded Software release
Developer Package installation
SDK for OpenSTLinux distribution
STM32CubeMP1 Package
Linux kernel, U-Boot, TF-A and OP-
TEE OS source code (OpenSTLinux distribution)
Distribution Package installation
OpenSTLinux distribution (full
source code and OpenEmbedded-based build framework)
Starter Package installation
Software image (binaries)



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard
│               │       size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│           ├── scripts
│           │   ├── create_sdcard_from_flashlayout.sh
│           │   └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│           └── tfs partition
│               ├── st-image-bootfs-openstlinux-weston-stm32mp1.manifest
│               ├── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│               └── rfs partition
│                   ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│                   ├── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│                   └── dorfs partition
│                       ├── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│                       └── tfs partition
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.license
│                           ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│                           ├── [...]
│                           └── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│                               └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│                                   ├── [...]
│                                   └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│                                       └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│                                           ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│                                           └── for FSBL partition (trusted boot chain) → STM32MP157C-DK2
│                                               ├── [...]
│                                               └── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│                                                   └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│                                                       ├── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│                                                       └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```




Example of directory structure for Packages

```
├─ u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└─ [...]
```



4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)

```

Developer-Package
├── SDK
│   └── environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│       └── site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           ├── sysroots
│           │   ├── cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           │   │   └── [...]
│           │   └── x86_64-ostl_sdk-linux
│           │       └── [...]
│           └── version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
└── [...]
  
```

for OpenSTLinux distribution: details in Standard SDK directory structure article

script for Developer Package

Environment setup

Target sysroot

Native sysroot

(libraries, headers, and symbols)

(libraries, headers, and symbols)

SDK

```

├── STM32Cube_FW_MP1_V1.5.0
│   ├── Drivers
│   │   ├── BSP
│   │   │   └── [...]
│   │   ├── CMSIS
│   │   │   └── [...]
│   │   └── STM32MP1xx_HAL_Driver
│   │       └── [...]
│   ├── htmresc
│   │   └── [...]
│   ├── License.md
│   ├── Middlewares
│   │   └── [...]
│   ├── package.xml
│   ├── Projects
│   │   ├── STM32CubeProjectsList.html
│   │   ├── STM32MP157C-DK2
│   │   │   └── [...]
│   │   └── STM32MP157C-EV1
│   │       └── [...]
│   ├── Readme.md
│   ├── Release_Notes.html
│   └── Utilities
│       └── [...]
└── [...]
  
```

STM32CubeMP1 Package: details in STM32CubeMP1 Package content article

BSP drivers for the supported STM32MPU boards

HAL drivers for the supported STM32MPU devices

License types for the components

Middlewares used by the examples and applications

STM32CubeMP1 Package

List of examples and applications for

Set of examples and applications → STM32MP157C-DK2

Set of examples and applications → STM32MP157C-EV1

Release note for STM32CubeMP1 Package



```

└─ stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
distribution
├─ images
│   └─ stm32mp1
directory
├─ tf-a-bl2-optee.elf           Debug symbol files installation
TEE OS → trusted boot firmware stage
├─ tf-a-bl2-trusted.elf       Debug symbol file for TF-A, with OP-
boot firmware stage
├─ tf-a-bl32-trusted.elf      Debug symbol file for TF-A → trusted
software stage
├─ u-boot-stm32mp157a-dk1-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157A-DK1
├─ u-boot-stm32mp157a-dk1-trusted.elf  Debug symbol file for U-Boot →
STM32MP157A-DK1
├─ u-boot-stm32mp157c-dk2-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-trusted.elf  Debug symbol file for U-Boot →
STM32MP157C-DK2
├─ u-boot-stm32mp157c-ev1-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-EV1
├─ u-boot-stm32mp157c-ev1-trusted.elf  Debug symbol file for U-Boot →
STM32MP157C-EV1
├─ vmlinux
└─ [...]

```

```

└─ sources
├─ arm-openstlinux_weston-linux-gnueabi
│   └─ linux-5.10.61
│       ├── [*].patch           ST patches for Linux kernel
│       ├── fragment-[*].config  ST configuration fragments for Linux kernel
│       ├── linux-5.10.61        Linux kernel source code directory
│       ├── linux-5.10.61.tar.xz
│       ├── README.HOW_TO.txt    Helper file for Linux kernel management: referenc
│       └─ series
code directory
└─ e for Linux kernel build

```

```

└─ optee-os-stm32mp-3.12.0-stm32mp-r2  OP-TEE OS installation directory
│   ├── [*].patch                 ST patches for OP-TEE OS
│   ├── optee-os-stm32mp-3.12.0-stm32mp-r2
│   ├── Makefile.sdk              Makefile for the OP-TEE OS compilation
│   └─ optee-os-stm32mp-3.12.0-stm32mp-r2-r0.tar.gz  OP-TEE OS source
code directory
└─ for OP-TEE OS build
    ├── README.HOW_TO.txt         Helper file for OP-TEE OS management: reference
    └─ series

```

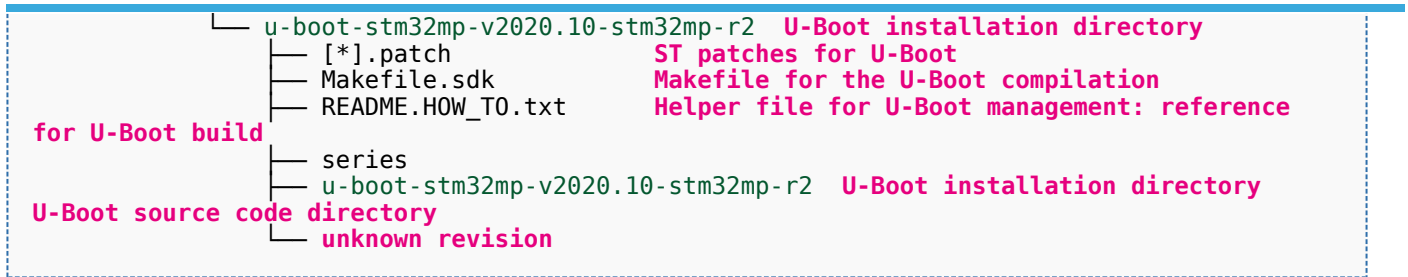
```

└─ tf-a-stm32mp-v2.4-stm32mp-r2  TF-A installation directory
│   ├── [*].patch                 ST patches for TF-A
│   ├── tf-a-stm32mp-v2.4-stm32mp-r2  TF-A source code directory
│   ├── Makefile.sdk              Makefile for the TF-A compilation
│   └─ README.HOW_TO.txt         Helper file for TF-A management: reference
code directory
└─ for TF-A build
    ├── series
    └─ tf-a-stm32mp-v2.4-stm32mp-r2-r0.tar.gz

```



Example of directory structure for Packages



Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.

```

Distribution-Package
├── openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution
│   └── layers
│       ├── meta-openembedded  Collection of layers for the OpenEmbedded-Core universe (Op
│       │                       enEmbedded standard)
│       │   └── [...]
│       └── meta-qt5           QT5 layer for OpenEmbedded (standard)
│           └── [...]

```

```

├── meta-st
│   ├── meta-st-openstlinux  STMicroelectronics layer that
│   │                       contains the settings of the frameworks and images for the OpenSTLinux distribution
│   │
│   ├── recipes-st
│   │   ├── images
│   │   │   └── st-image-core.bb  Core image for OpenSTLinux
│   │   └── st-image-weston.bb  Weston image with basic Wayland
│   │                           support for OpenSTLinux distribution: recommended setup
│   └── [...]
│       ├── packagegroups
│       │   └── [...]

```

```

├── meta-st-stm32mp  STMicroelectronics layer that contains
│   │               the description of the BSP for the STM32 MPU devices
│   │
│   ├── recipes-bsp
│   │   ├── alsa
│   │   │   └── [...]
│   │   └── drivers
│   │       └── [...]
│   │
│   ├── recipes-extended
│   │   ├── m4projects
│   │   │   └── [...]
│   │   └── [...]
│   │
│   ├── recipes-graphics
│   │   └── gcnano-userland
│   │       └── [...]
│   │
│   ├── recipes-kernel
│   │   ├── linux
│   │   │   └── [...]
│   │   └── linux-firmware
│   │       └── [...]
│   └── [...]

```

drivers

within the OpenSTLinux distribution

ES, OpenVG and EGL (multi backend)

Bluetooth firmware)

Recipes for ALSA control configuration

Recipes for Vivante GCNANO GPU kernel

Recipes for TF-A

Recipes for U-Boot

Recipes for STM32Cube MPU Package

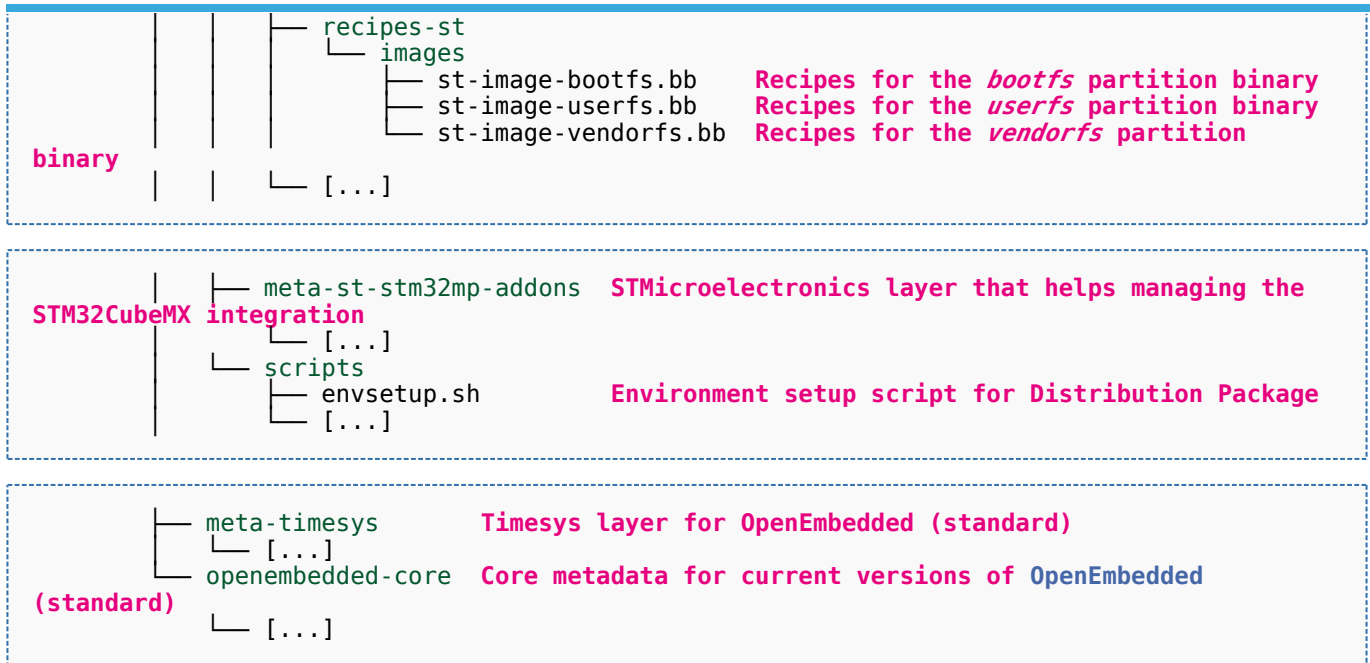
Recipes for Vivante libraries OpenGL

Recipes for Linux kernel

Recipes for Linux firmwares (example,



Example of directory structure for Packages

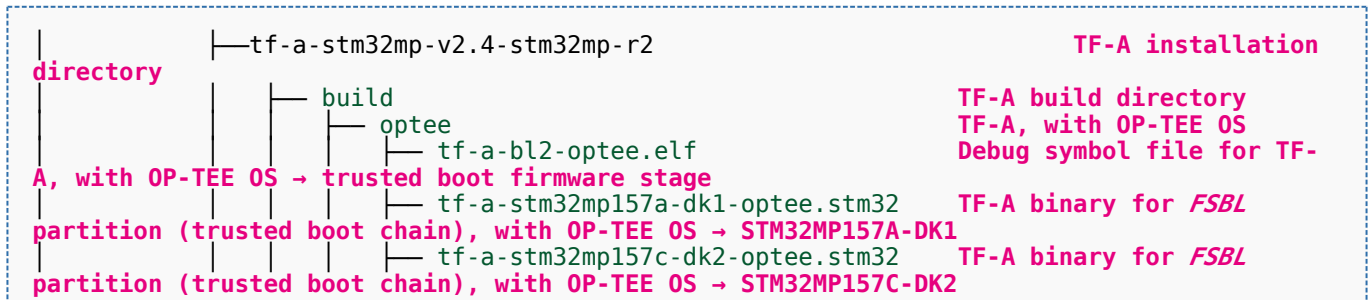
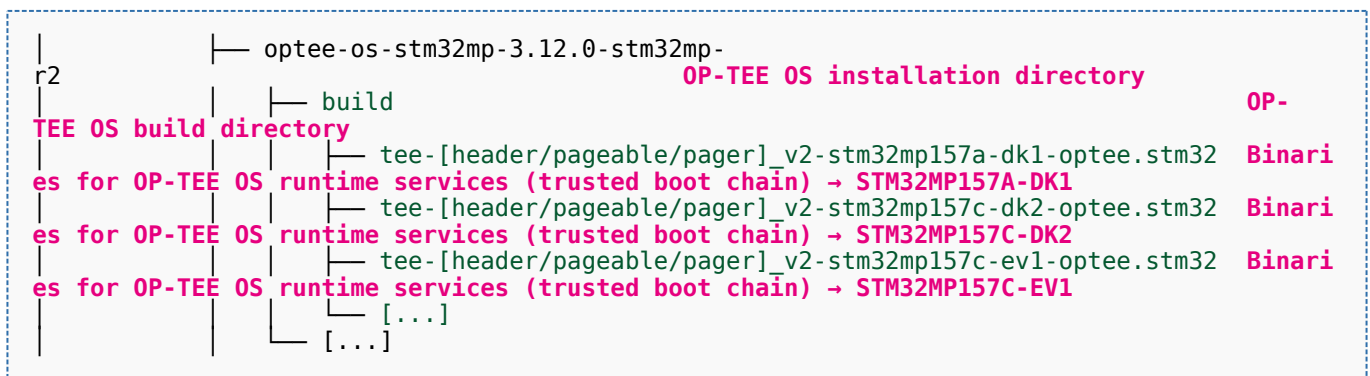
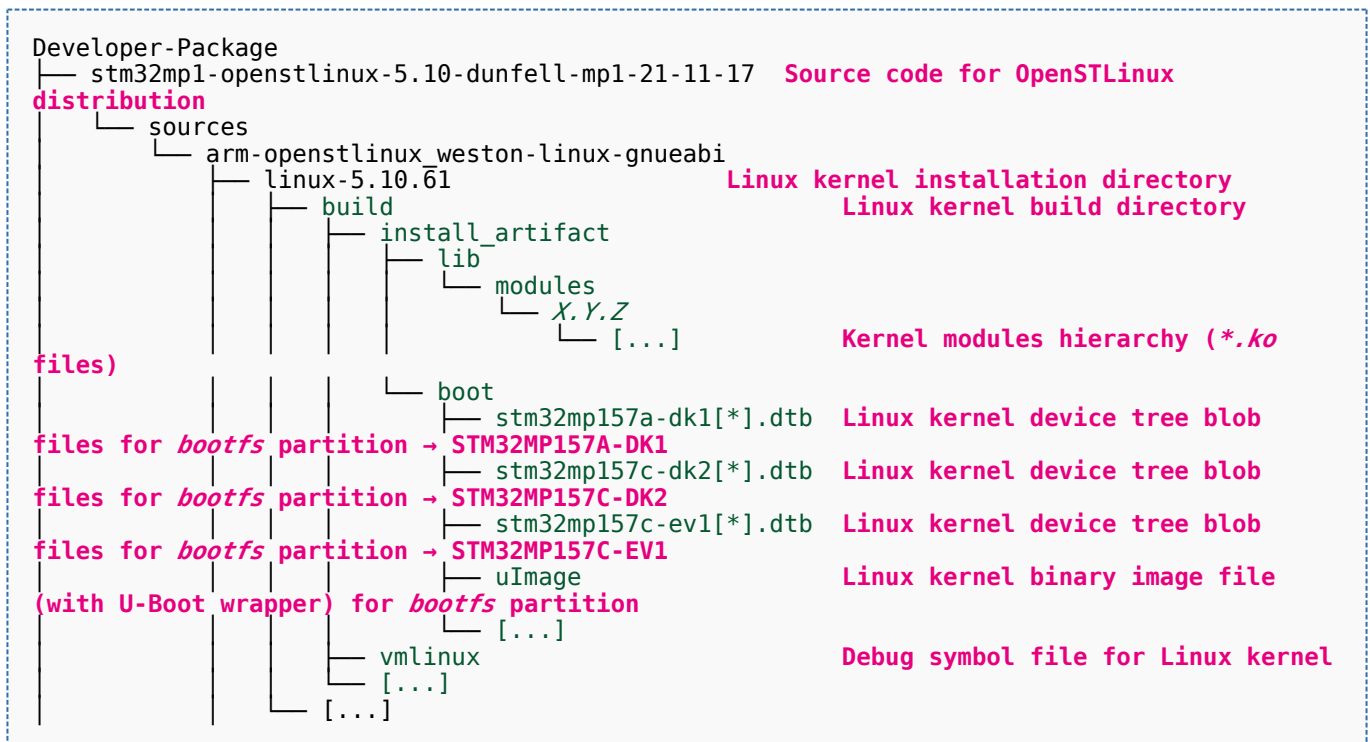


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages

partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-optee.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	trusted	TF-A, without OP-TEE OS
A → trusted boot firmware stage	tf-a-bl2-trusted.elf	Debug symbol file for TF-
A → trusted boot firmware stage	tf-a-bl32-trusted.elf	Debug symbol file for TF-
partition (trusted boot chain) → STM32MP157A-DK1	tf-a-stm32mp157a-dk1-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	tf-a-stm32mp157c-dk2-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-trusted.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	[...]	

installation directory	u-boot-stm32mp-v2020.10-stm32mp-r2	U-Boot
for basic boot chain	build-basic	U-Boot build directory
partition (basic boot chain) → STM32MP157A-DK1	u-boot-spl.stm32-stm32mp157a-dk1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-spl.stm32-stm32mp157c-dk2-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-spl.stm32-stm32mp157c-ev1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for <i>SSBL</i>
for trusted boot chain, with OP-TEE OS	build-optee	U-Boot build directory
Boot, with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
for trusted boot chain	build-trusted	U-Boot build directory
Boot → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
	[...]	



Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv       Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv   Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv       Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv     Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv       Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv     Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv       Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv       Flash layout file
```



Example of directory structure for Packages

```

for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv          Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4          Binary for bootfs
partition └── st-image-userfs-openstlinux-weston-stm32mp1.ext4        Binary for userfs
s partition └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4    Binary for vendorfs
partition └── st-image-weston-openstlinux-weston-stm32mp1.ext4        Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1
├── stm32mp157a-dk1[*].dtb                                             Linux kernel
device tree blob files for bootfs partition → STM32MP157C-DK2
├── stm32mp157c-dk2[*].dtb                                             Linux kernel
device tree blob files for bootfs partition → STM32MP157C-EV1
├── stm32mp157c-e[*].dtb                                              Linux kernel
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32       Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32       Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32       Binaries for OP-
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-optee.elf                                                 Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                               Debug symbol file
for TF-A → runtime software stage
├── tf-a-bl32-trusted.elf                                              Debug symbol file
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-optee.stm32                                   TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32                                TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                                   TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                                TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                                   TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                                TF-A binary for FS
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                             U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                             U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                             U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-basic.img                                    U-Boot binary for
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                                  Debug symbol file
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.elf                                Debug symbol file
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.stm32                                U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                                U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-basic.img                                    U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                                  Debug symbol file

```



for U-Boot, with OP-TEE OS → STM32MP157C-DK2	
— u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	
— u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file
for U-Boot → STM32MP157C-DK2	
— u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157C-DK2	
— u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file
for U-Boot, with OP-TEE OS → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file
for U-Boot → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157C-EV1	
— uImage	Linux kernel
binary image file (with U-Boot wrapper) for <i>bootfs</i> partition	
— vmlinux	Debug symbol file
for Linux kernel	
— [...]	
[...]	

Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 09.11.2021 - 13:16 / Revision: 09.11.2021 - 13:16

Contents

1 Article purpose	109
2 Creating the structure	110
3 Focus on the Starter Package directory	111
4 Focus on the Developer Package directory	113
5 Focus on the Distribution Package directory	116
6 Appendix A: directory structure after build (Developer Package)	118
7 Appendix B: directory structure after build (Distribution Package)	121



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
├── Distribution-Package
└── Starter-Package
```

STM32MPU Embedded Software release
Developer Package installation directory
Distribution Package installation directory
Starter Package installation directory

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
│   ├── SDK
│   ├── STM32Cube_FW_MP1_V1.5.0
│   └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│       ├── TEE OS source code (OpenSTLinux distribution)
│       └── Distribution-Package
│           ├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│           └── Starter-Package
│               ├── source code and OpenEmbedded-based build framework
│               └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│                   ├── OpenSTLinux distribution (full
│                   │   source code and OpenEmbedded-based build framework)
│                   └── Starter Package installation
│                       ├── Software image (binaries)
```

STM32MPU Embedded Software release
Developer Package installation
SDK for OpenSTLinux distribution
STM32CubeMP1 Package
Linux kernel, U-Boot, TF-A and OP-
Distribution Package installation
OpenSTLinux distribution (full
source code and OpenEmbedded-based build framework)
Starter Package installation
Software image (binaries)



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard
│               │       size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│                   ├── scripts
│                   │   └── create_sdcard_from_flashlayout.sh
│                   └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│                       └── tfs partition
│                           ├── st-image-bootfs-openstlinux-weston-stm32mp1.manifest
│                           └── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│                               └── rfs partition
│                                   ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│                                   └── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│                                       └── dorfs partition
│                                           └── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│                                               └── tfs partition
│                                                   ├── st-image-weston-openstlinux-weston-stm32mp1.license
│                                                   ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│                                                   ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│                                                   ├── [...]
│                                                   └── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│                                                       └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│                                                           ├── [...]
│                                                           └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│                                                               └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│                                                                   ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│                                                                   └── for FSBL partition (trusted boot chain) → STM32MP157C-DK2
│                                                                       ├── [...]
│                                                                       └── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│                                                                           └── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│                                                                               ├── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│                                                                               └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

```
├── u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└── [...]
```




4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)

```

Developer-Package
├── SDK
│   └── environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│       └── site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           ├── sysroots
│           │   ├── cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           │   │   ├── [...]
│           │   │   └── x86_64-ostl_sdk-linux
│           │   │       ├── [...]
│           │   └── version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
│           └── [...]
└── [...]
  
```

for OpenSTLinux distribution: details in Standard SDK directory structure article

script for Developer Package

Environment setup

Target sysroot

Native sysroot

(libraries, headers, and symbols)

(libraries, headers, and symbols)

```

├── STM32Cube_FW_MP1_V1.5.0
│   ├── Drivers
│   │   ├── BSP
│   │   │   ├── [...]
│   │   │   └── CMSIS
│   │   │       ├── [...]
│   │   └── STM32MP1xx_HAL_Driver
│   │       ├── [...]
│   ├── htmresc
│   │   ├── [...]
│   ├── License.md
│   ├── Middlewares
│   │   ├── [...]
│   ├── package.xml
│   ├── Projects
│   │   ├── STM32CubeProjectsList.html
│   │   ├── STM32MP157C-DK2
│   │   │   ├── [...]
│   │   └── STM32MP157C-EV1
│   │       ├── [...]
│   ├── Readme.md
│   ├── Release_Notes.html
│   ├── Utilities
│   │   ├── [...]
  
```

STM32CubeMP1 Package: details in STM32CubeMP1 Package content article

BSP drivers for the supported STM32MPU boards

HAL drivers for the supported STM32MPU devices

License types for the components

Middlewares used by the examples and applications

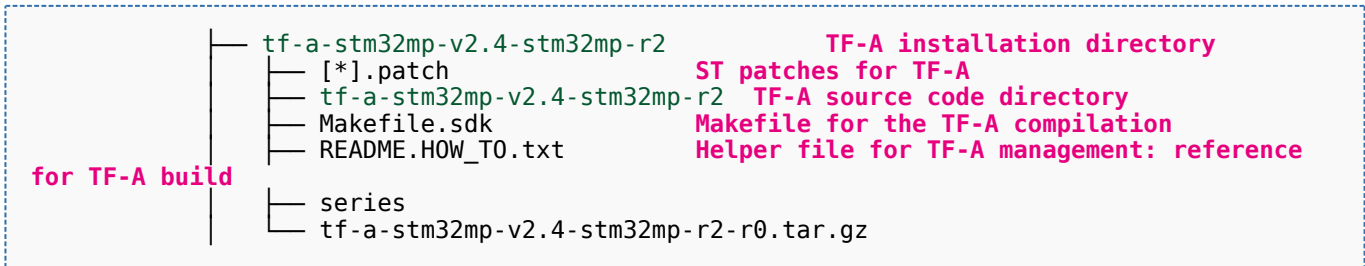
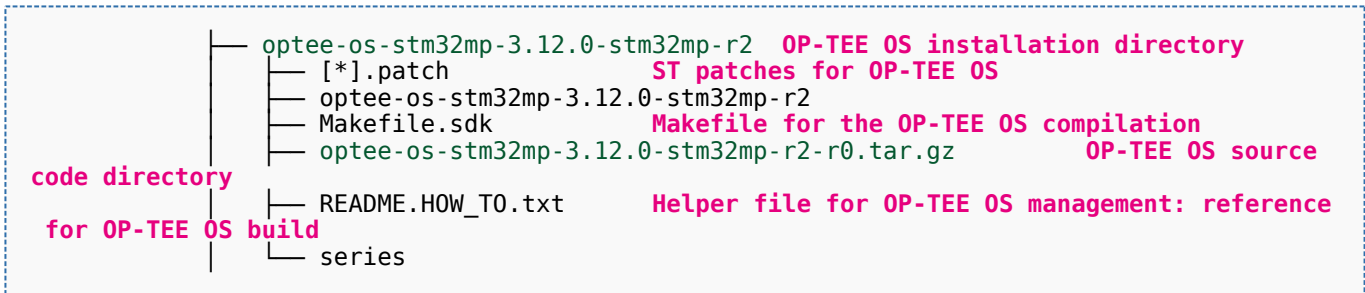
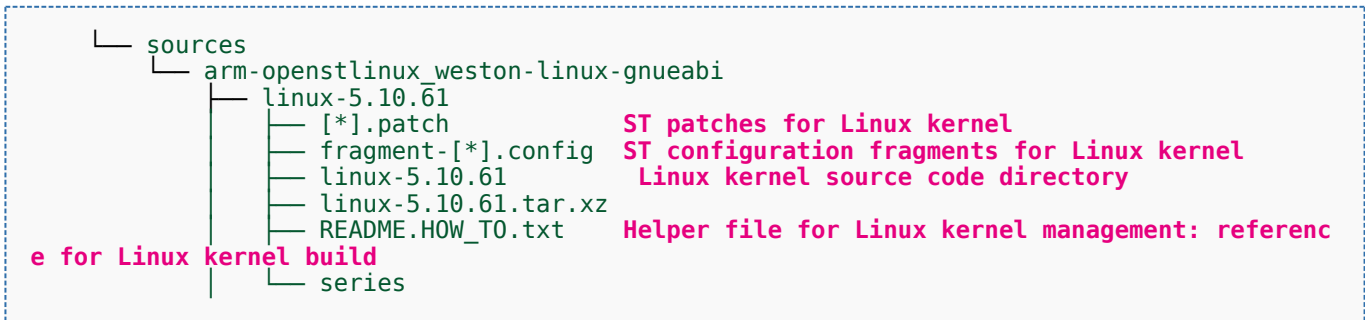
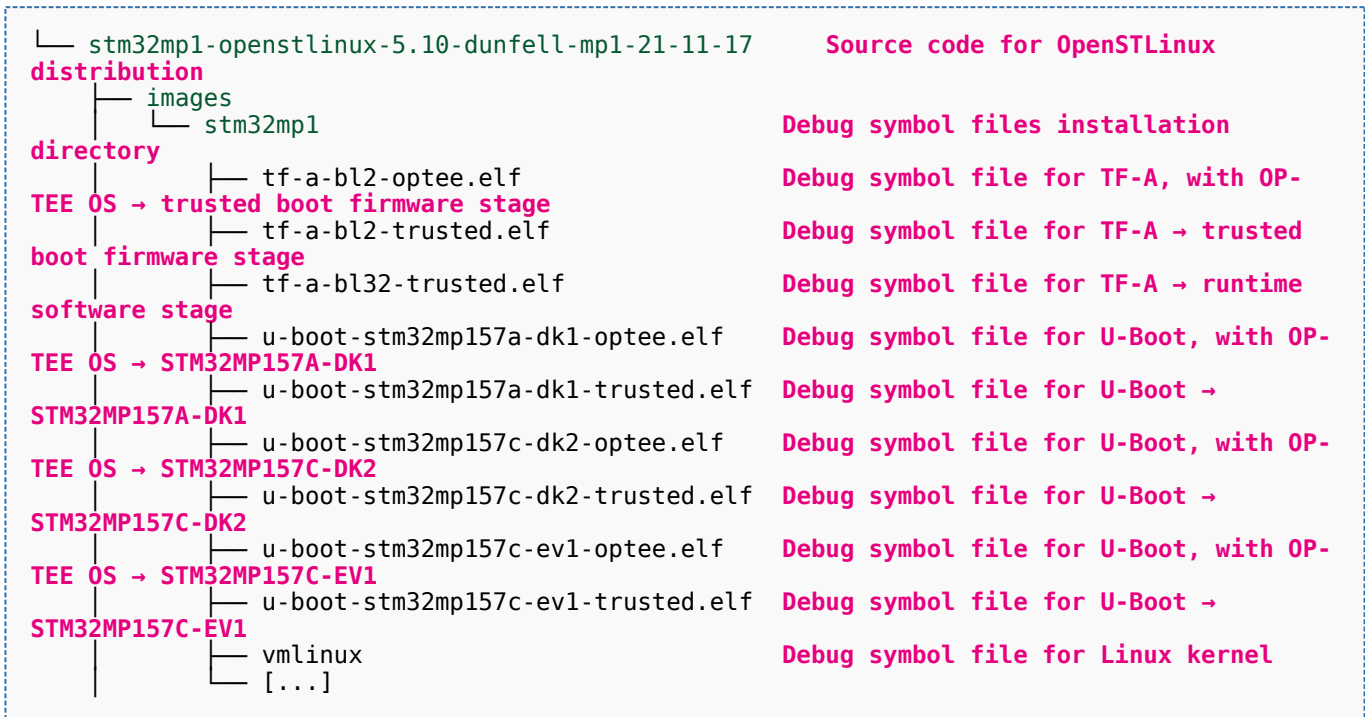
STM32CubeMP1 Package

List of examples and applications for

Set of examples and applications → STM32MP157C-DK2

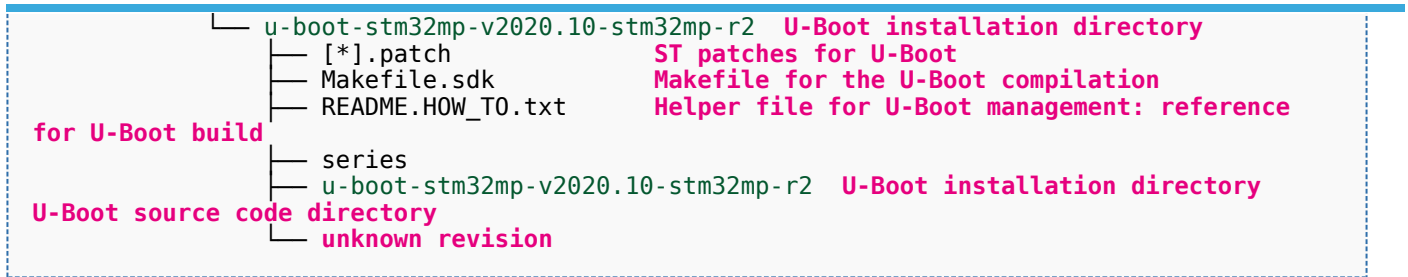
Set of examples and applications → STM32MP157C-EV1

Release note for STM32CubeMP1 Package





Example of directory structure for Packages



Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.

```

Distribution-Package
├── openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution
│   └── layers
│       ├── meta-openembedded  Collection of layers for the OpenEmbedded-Core universe (Op
│       │                       enEmbedded standard)
│       │   └── [...]
│       └── meta-qt5           QT5 layer for OpenEmbedded (standard)
│           └── [...]

```

```

├── meta-st
│   ├── meta-st-openstlinux  STMicroelectronics layer that
│   │                       contains the settings of the frameworks and images for the OpenSTLinux distribution
│   │
│   ├── recipes-st
│   │   ├── images
│   │   │   └── st-image-core.bb  Core image for OpenSTLinux
│   │   └── st-image-weston.bb   Weston image with basic Wayland
│   │                               support for OpenSTLinux distribution: recommended setup
│   └── [...]
│       ├── packagegroups
│       │   └── [...]

```

```

├── meta-st-stm32mp  STMicroelectronics layer that contains
│   │               the description of the BSP for the STM32 MPU devices
│   │
│   ├── recipes-bsp
│   │   ├── alsa
│   │   │   └── [...]
│   │   └── drivers
│   │       └── [...]
│   │
│   ├── recipes-extended
│   │   ├── m4projects
│   │   │   └── [...]
│   │   └── [...]
│   │
│   ├── recipes-graphics
│   │   ├── gcnano-userland
│   │   │   └── [...]
│   │   └── [...]
│   │
│   ├── recipes-kernel
│   │   ├── linux
│   │   │   └── [...]
│   │   └── linux-firmware
│   │       └── [...]
│   │
│   └── [...]

```

drivers

within the OpenSTLinux distribution

ES, OpenVG and EGL (multi backend)

Bluetooth firmware)

Recipes for ALSA control configuration

Recipes for Vivante GCNANO GPU kernel

Recipes for TF-A

Recipes for U-Boot

Recipes for STM32Cube MPU Package

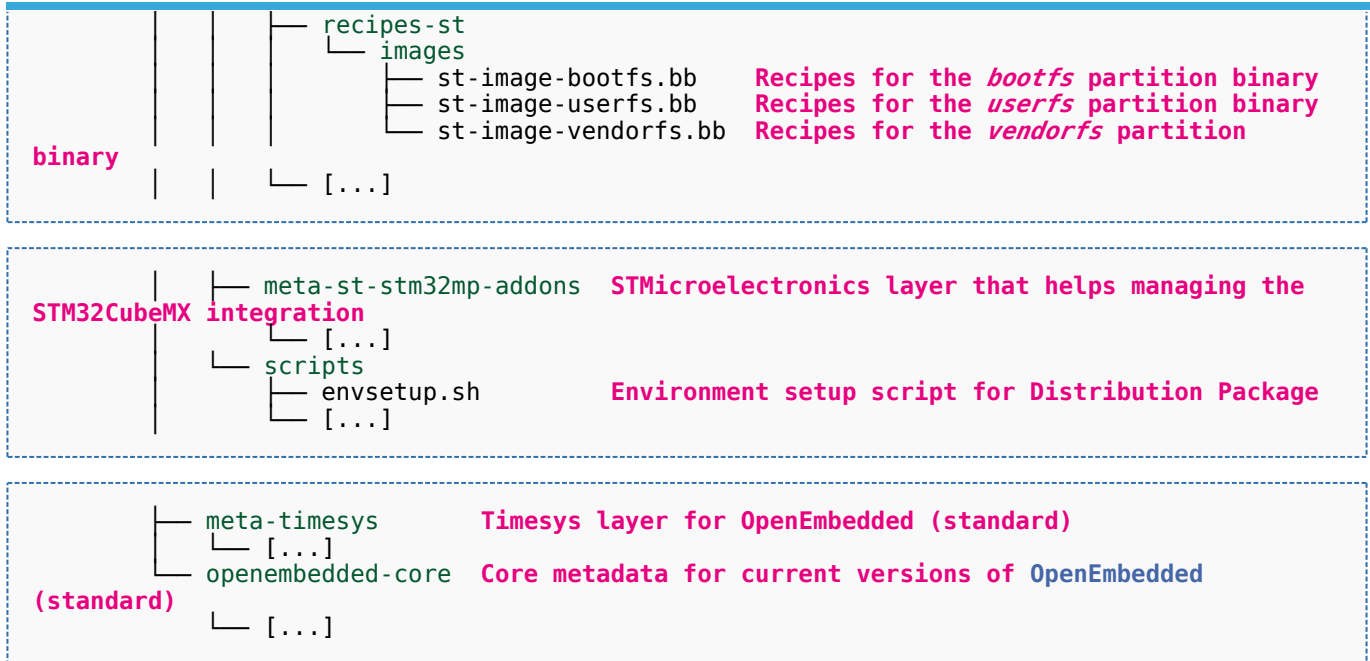
Recipes for Vivante libraries OpenGL

Recipes for Linux kernel

Recipes for Linux firmwares (example,



Example of directory structure for Packages

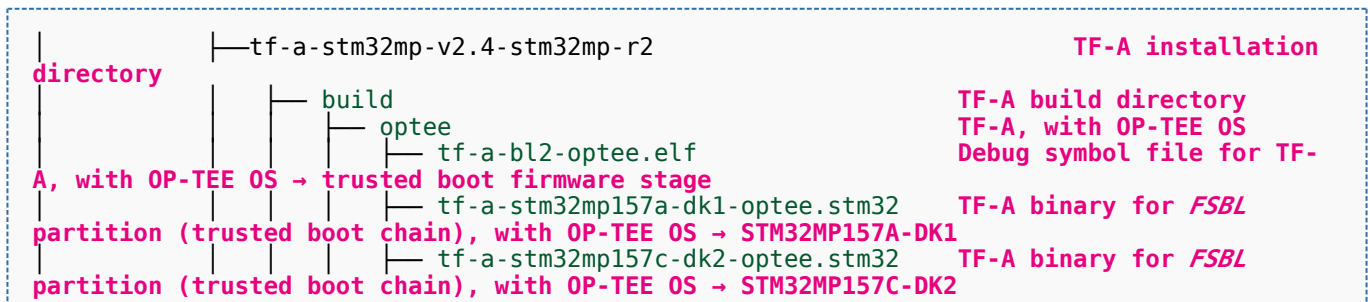
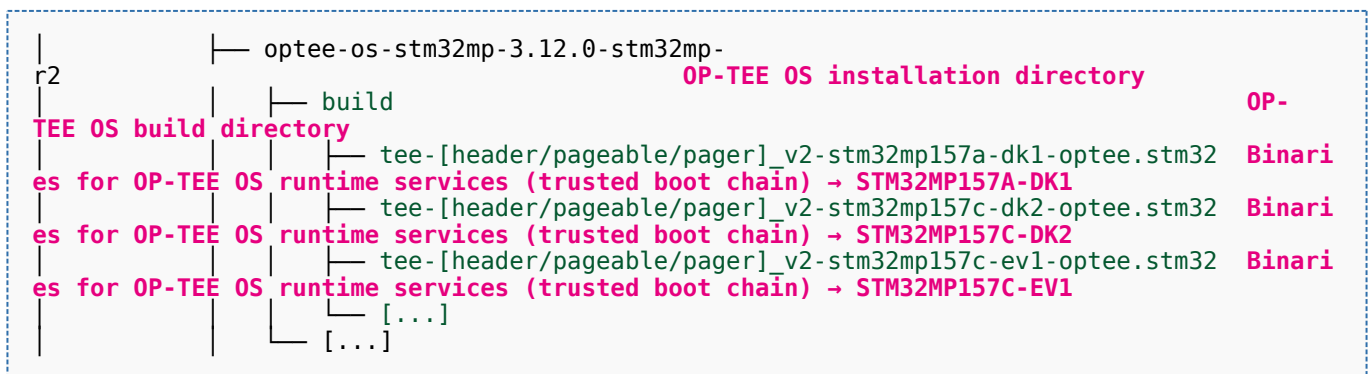
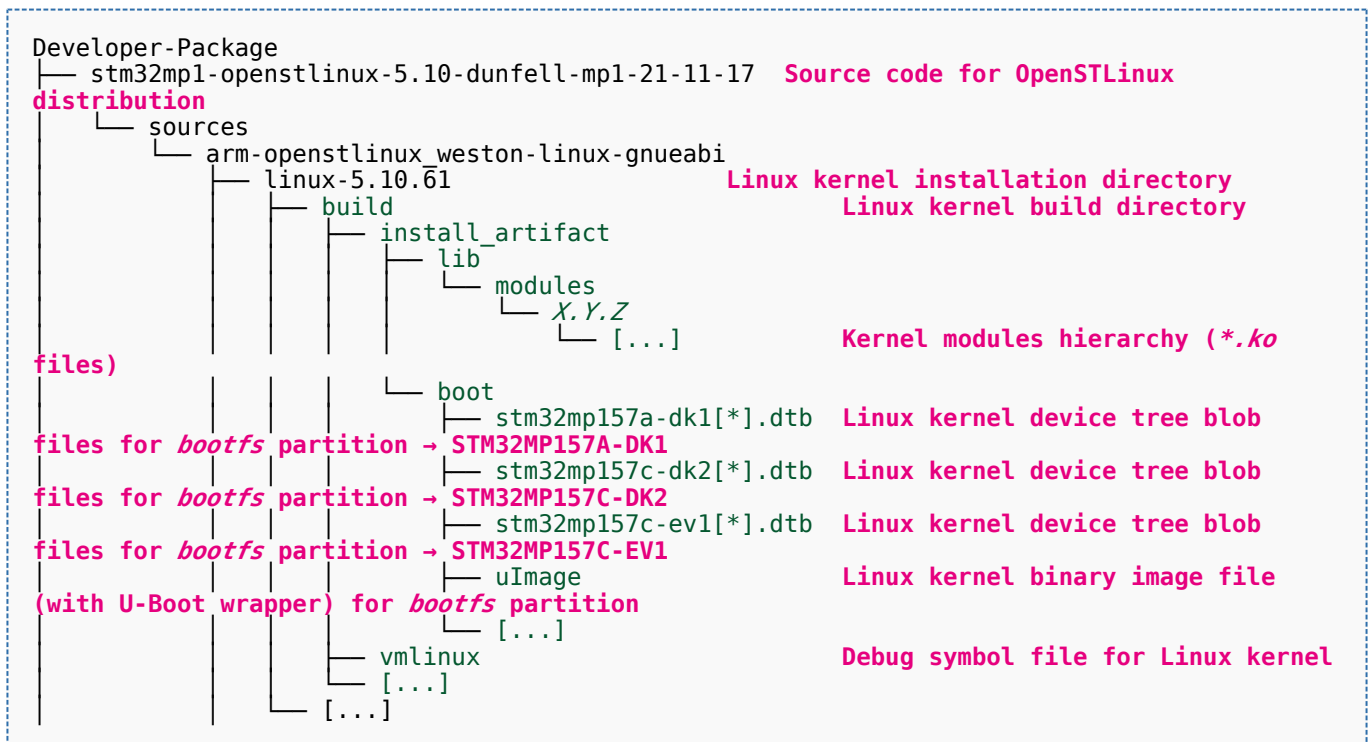


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages

partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-optee.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	trusted	TF-A, without OP-TEE OS
A → trusted boot firmware stage	tf-a-bl2-trusted.elf	Debug symbol file for TF-
A → trusted boot firmware stage	tf-a-bl32-trusted.elf	Debug symbol file for TF-
partition (trusted boot chain) → STM32MP157A-DK1	tf-a-stm32mp157a-dk1-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	tf-a-stm32mp157c-dk2-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-trusted.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	[...]	

installation directory	u-boot-stm32mp-v2020.10-stm32mp-r2	U-Boot
for basic boot chain	build-basic	U-Boot build directory
partition (basic boot chain) → STM32MP157A-DK1	u-boot-spl.stm32-stm32mp157a-dk1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-spl.stm32-stm32mp157c-dk2-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-spl.stm32-stm32mp157c-ev1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for <i>SSBL</i>
for trusted boot chain, with OP-TEE OS	build-optee	U-Boot build directory
Boot, with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
for trusted boot chain	build-trusted	U-Boot build directory
Boot → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
	[...]	



Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv         Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv       Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv         Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv         Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv         Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv         Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv       Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv         Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv         Flash layout file
```



Example of directory structure for Packages

```

for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv      Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition ─── st-image-bootfs-openstlinux-weston-stm32mp1.ext4      Binary for bootfs
partition ─── st-image-userfs-openstlinux-weston-stm32mp1.ext4     Binary for userfs
s partition ─── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4  Binary for vendorfs
partition ─── st-image-weston-openstlinux-weston-stm32mp1.ext4     Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1     Linux kernel
├── stm32mp157a-dk1[*].dtb
device tree blob files for bootfs partition → STM32MP157C-DK2     Linux kernel
├── stm32mp157c-dk2[*].dtb
device tree blob files for bootfs partition → STM32MP157C-EV1     Linux kernel
├── stm32mp157c-e[*].dtb
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1     Binaries for OP-
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2     Binaries for OP-
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1     Binaries for OP-
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32
for TF-A, with OP-TEE OS → trusted boot firmware stage             Debug symbol file
├── tf-a-bl2-optee.elf
for TF-A → trusted boot firmware stage                             Debug symbol file
├── tf-a-bl2-trusted.elf
for TF-A → runtime software stage                                 Debug symbol file
├── tf-a-bl32-trusted.elf
TF-A partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1  TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32
BL partition (trusted boot chain) → STM32MP157A-DK1                TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32
TF-A partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2  TF-A binary for FS
├── tf-a-stm32mp157c-dk2-trusted.stm32
BL partition (trusted boot chain) → STM32MP157C-DK2                TF-A binary for FS
├── tf-a-stm32mp157c-dk2-trusted.stm32
TF-A partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1  TF-A binary for FS
├── tf-a-stm32mp157c-ev1-trusted.stm32
BL partition (trusted boot chain) → STM32MP157C-EV1                U-Boot binary for
├── u-boot-spl.stm32-stm32mp157a-dk1-basic
FSBL partition (basic boot chain) → STM32MP157A-DK1                U-Boot binary for
├── u-boot-spl.stm32-stm32mp157a-dk1-basic
FSBL partition (basic boot chain) → STM32MP157C-DK2                U-Boot binary for
├── u-boot-spl.stm32-stm32mp157c-dk2-basic
FSBL partition (basic boot chain) → STM32MP157C-EV1                U-Boot binary for
├── u-boot-spl.stm32-stm32mp157c-ev1-basic
SSBL partition (basic boot chain) → STM32MP157A-DK1                Debug symbol file
├── u-boot-stm32mp157a-dk1-optee.elf
for U-Boot, with OP-TEE OS → STM32MP157A-DK1                      U-Boot binary for
├── u-boot-stm32mp157a-dk1-trusted.stm32
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1  Debug symbol file
├── u-boot-stm32mp157a-dk1-trusted.elf
for U-Boot → STM32MP157A-DK1                                       U-Boot binary for
├── u-boot-stm32mp157a-dk1-trusted.stm32
SSBL partition (trusted boot chain) → STM32MP157C-DK2                U-Boot binary for
├── u-boot-stm32mp157c-dk2-basic.img
SSBL partition (basic boot chain) → STM32MP157C-DK2                Debug symbol file
├── u-boot-stm32mp157c-dk2-optee.elf

```



for U-Boot, with OP-TEE OS → STM32MP157C-DK2	
— u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	
— u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file
for U-Boot → STM32MP157C-DK2	
— u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157C-DK2	
— u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file
for U-Boot, with OP-TEE OS → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file
for U-Boot → STM32MP157C-EV1	
— u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157C-EV1	
— uImage	Linux kernel
binary image file (with U-Boot wrapper) for <i>bootfs</i> partition	
— vmlinux	Debug symbol file
for Linux kernel	
— [...]	
— [...]	

Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))

Stable: 19.10.2021 - 13:54 / Revision: 19.10.2021 - 13:54

Contents

1 Article purpose	124
2 Creating the structure	125
3 Focus on the Starter Package directory	126
4 Focus on the Developer Package directory	128
5 Focus on the Distribution Package directory	131
6 Appendix A: directory structure after build (Developer Package)	133
7 Appendix B: directory structure after build (Distribution Package)	136



1 Article purpose

This article aims at proposing a way to organize, on the host PC, the software packages of the different Packages (Starter, Developer and Distribution) for a given release of the STM32MPU Embedded Software distribution.

The main objective of the proposed organization is to keep together the software packages corresponding to a given release because there are links between them. For example:

- Flashing the image from the Starter Package on the board is mandatory before modifying the source code from the Developer Package. Both the image and the source code must belong to the same software release.
- The SDK (Developer Package) and the image (Starter Package) have both been generated from the Distribution Package. A software release thus guarantees that there is no misalignment between the different software packages.

Information

The objective of this article is to describe one organization among all the possible organizations. Feel free to organize the delivered Packages in any other way that would better match your way of working.

Information

In practice, this article uses the release **STM32MP15-Ecosystem-v3.1.0** for the STM32MPU Embedded Software distribution as an example to illustrate the proposed organization. If you are using a different release, the names of the directories and files might differ.

The directories are shown in green, while the files are in black.



2 Creating the structure

- Create your <working directory> and assign a unique name to it (for example by including the release name):

```
PC $> mkdir STM32MP15-Ecosystem-v3.1.0
PC $> cd STM32MP15-Ecosystem-v3.1.0
```

- Create the first-level directories that will host the software packages delivered through the STM32MPU Embedded Software distribution release note:

```
PC $> mkdir Starter-Package
PC $> mkdir Developer-Package
PC $> mkdir Distribution-Package
```

- The resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
├── Distribution-Package
└── Starter-Package
```

STM32MPU Embedded Software release
Developer Package installation directory
Distribution Package installation directory
Starter Package installation directory

Once all software packages have been installed according to the instructions given in the STM32MPU Embedded Software distribution release note, the resulting directory structure looks as follows:

```
STM32MP15-Ecosystem-v3.1.0
├── Developer-Package
│   ├── SDK
│   ├── STM32Cube_FW_MP1_V1.5.0
│   └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│       ├── TEE OS source code (OpenSTLinux distribution)
│       └── Distribution-Package
│           ├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│           └── Starter-Package
│               ├── source code and OpenEmbedded-based build framework
│               └── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│                   ├── OpenSTLinux distribution (full
│                   │   source code and OpenEmbedded-based build framework)
│                   └── Starter Package installation
│                       ├── Software image (binaries)
```

STM32MPU Embedded Software release
Developer Package installation
SDK for OpenSTLinux distribution
STM32CubeMP1 Package
Linux kernel, U-Boot, TF-A and OP-
TEE OS source code (OpenSTLinux distribution)
Distribution Package installation
OpenSTLinux distribution (full
source code and OpenEmbedded-based build framework)
Starter Package installation
Software image (binaries)



3 Focus on the Starter Package directory

The *Starter-Package* directory contains the software image for the STM32MPU Embedded Software distribution.

The trusted boot chain is the default solution delivered by STMicroelectronics. It includes the superset of features (for example, all Flash memory devices are supported). The basic boot chain is also upstreamed by STMicroelectronics, with a limited number of features (for example microSD card memory boot only). Refer to the [Boot chains overview](#) article for details.

Flash memory partitions (e.g. rootfs, bootfs...) are explained in the [STM32MP15 Flash mapping](#) article.

```

Starter-Package
├── stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
│   └── images
│       └── stm32mp1
│           ├── flashlayout_st-image-weston Flash layout
│           └── files (description of the partitions) for the supported Flash devices and boards
│               ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│               ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv Flash layout
│               │   └── file for eMMC and trusted boot chain → STM32MP157C-EV1
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv Flash layout
│               │   └── file for microSD card and basic boot chain → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv Flash layout
│               │   └── file for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│               ├── FlashLayout_sdcard_stm32mp157c-dk2-extensible.tsv Flash layout
│               │   └── file for microSD card with no userfs partition but a rootfs partition extended to sdcard size (recommended setup for package repository service) → STM32MP157C-DK2
│               └── [...]
│           ├── scripts
│           │   ├── create_sdcard_from_flashlayout.sh
│           │   └── st-image-bootfs-openstlinux-weston-stm32mp1.ext4 Binary for boo
│           └── tfs partition
│               ├── st-image-bootfs-openstlinux-weston-stm32mp1.manifest
│               ├── st-image-userfs-openstlinux-weston-stm32mp1.ext4 Binary for use
│               └── rfs partition
│                   ├── st-image-userfs-openstlinux-weston-stm32mp1.manifest
│                   ├── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4 Binary for ven
│                   └── dorfs partition
│                       ├── st-image-weston-openstlinux-weston-stm32mp1.ext4 Binary for roo
│                       └── tfs partition
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.license
│                           ├── st-image-weston-openstlinux-weston-stm32mp1-license_content.html
│                           ├── st-image-weston-openstlinux-weston-stm32mp1.manifest
│                           ├── [...]
│                           └── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32 Binaries for
│                               └── OP-TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
│                                   ├── [...]
│                                   └── tf-a-stm32mp157c-dk2-optee.stm32 TF-A binary
│                                       └── for FSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
│                                           ├── tf-a-stm32mp157c-dk2-trusted.stm32 TF-A binary
│                                           └── for FSBL partition (trusted boot chain) → STM32MP157C-DK2
│                                               ├── [...]
│                                               └── u-boot-spl.stm32-stm32mp157c-dk2-basic U-Boot binary
│                                                   ├── for FSBL partition (basic boot chain) → STM32MP157C-DK2
│                                                   └── u-boot-spl.stm32-stm32mp157c-ev1-basic U-Boot binary
│                                                       └── for FSBL partition (basic boot chain) → STM32MP157C-EV1

```



Example of directory structure for Packages

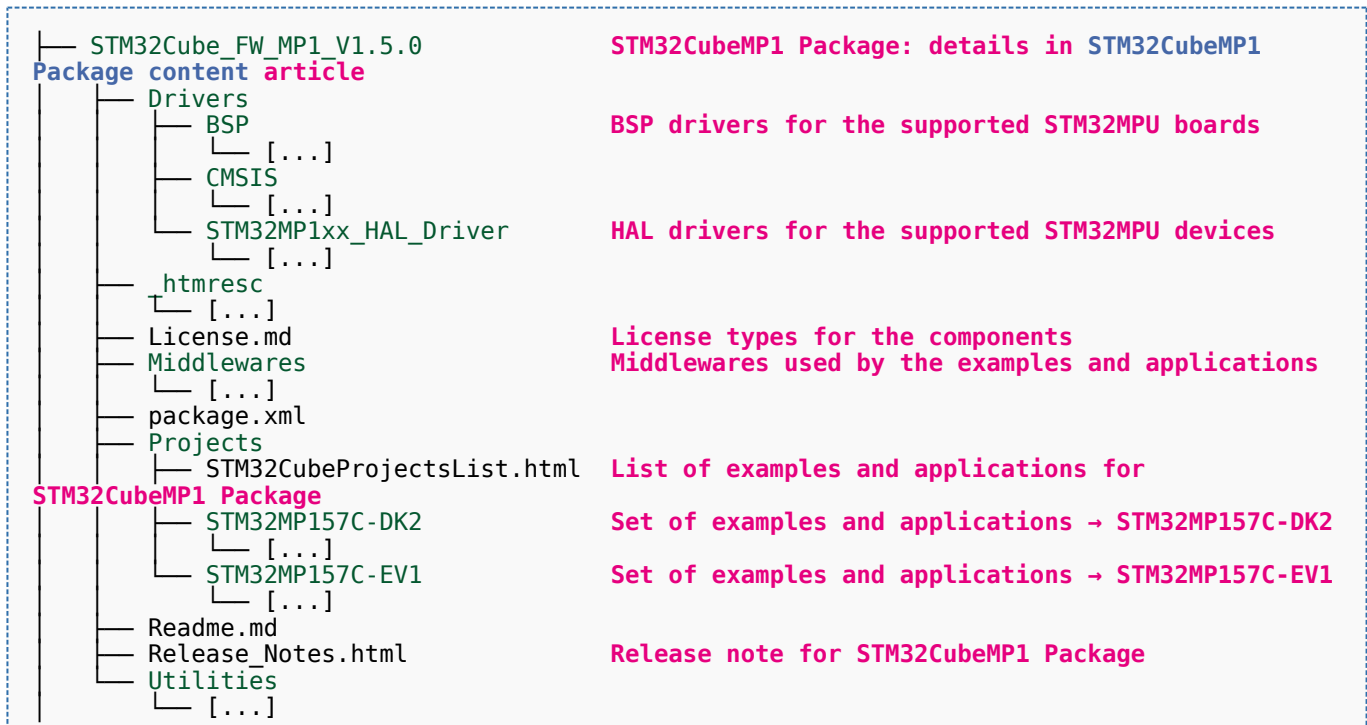
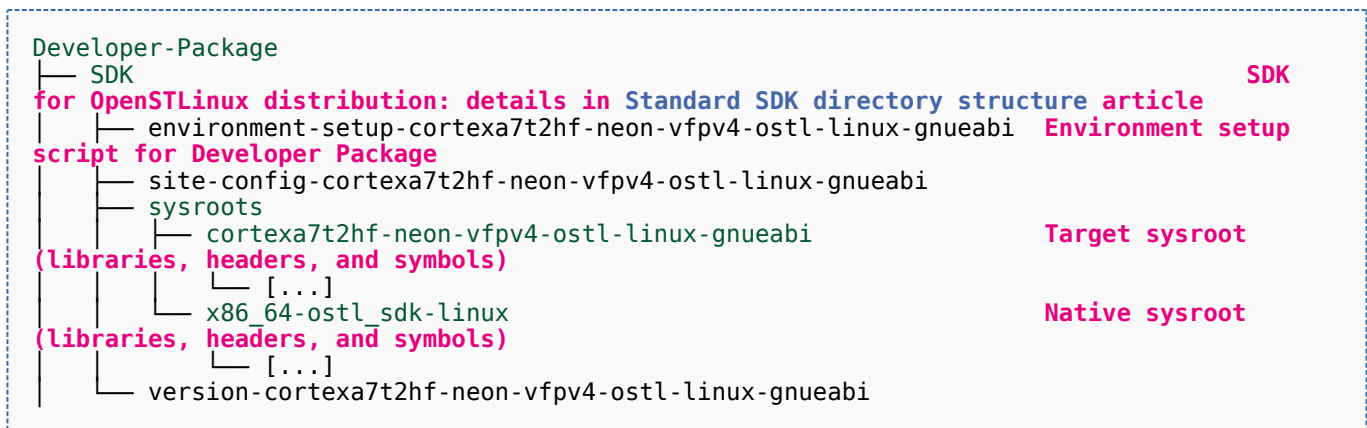
```
├─ u-boot-stm32mp157c-dk2-basic.img           U-Boot binary
for SSBL partition (basic boot chain) → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-optee.stm32       U-Boot binary
for SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
└─ [...]
```




4 Focus on the Developer Package directory

The *Developer-Package* directory contains:

- The source code for the following OpenSTLinux software packages (development for Arm[®] Cortex[®]-A processor):
 - Linux[®] kernel
 - U-Boot
 - TF-A
 - OP-TEE OS
- The debug symbol files for Linux kernel, U-Boot, TF-A and OP-TEE OS
- The SDK (for cross-development on an host PC)
- The STM32Cube MPU Package (developed for Arm[®] Cortex[®]-M processor)





```

└─ stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17
distribution
├─ images
│ └─ stm32mp1
directory
├─ tf-a-bl2-optee.elf           Debug symbol files installation
TEE OS → trusted boot firmware stage
├─ tf-a-bl2-trusted.elf       Debug symbol file for TF-A, with OP-
boot firmware stage
├─ tf-a-bl32-trusted.elf     Debug symbol file for TF-A → trusted
software stage
├─ u-boot-stm32mp157a-dk1-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157A-DK1
├─ u-boot-stm32mp157a-dk1-trusted.elf  Debug symbol file for U-Boot →
STM32MP157A-DK1
├─ u-boot-stm32mp157c-dk2-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-DK2
├─ u-boot-stm32mp157c-dk2-trusted.elf  Debug symbol file for U-Boot →
STM32MP157C-DK2
├─ u-boot-stm32mp157c-ev1-optee.elf  Debug symbol file for U-Boot, with OP-
TEE OS → STM32MP157C-EV1
├─ u-boot-stm32mp157c-ev1-trusted.elf  Debug symbol file for U-Boot →
STM32MP157C-EV1
├─ vmlinux
└─ [...]

```

```

└─ sources
├─ arm-openstlinux_weston-linux-gnueabi
│ └─ linux-5.10.61
│   ├── [*].patch           ST patches for Linux kernel
│   ├── fragment-[*].config  ST configuration fragments for Linux kernel
│   ├── linux-5.10.61       Linux kernel source code directory
│   ├── linux-5.10.61.tar.xz
│   ├── README.HOW_TO.txt   Helper file for Linux kernel management: referenc
│   └─ series
code directory
└─ e for Linux kernel build

```

```

└─ optee-os-stm32mp-3.12.0-stm32mp-r2  OP-TEE OS installation directory
│ ├── [*].patch           ST patches for OP-TEE OS
│ ├── optee-os-stm32mp-3.12.0-stm32mp-r2
│ ├── Makefile.sdk         Makefile for the OP-TEE OS compilation
│ └─ optee-os-stm32mp-3.12.0-stm32mp-r2-r0.tar.gz  OP-TEE OS source
code directory
└─ for OP-TEE OS build
├─ README.HOW_TO.txt       Helper file for OP-TEE OS management: reference
└─ series

```

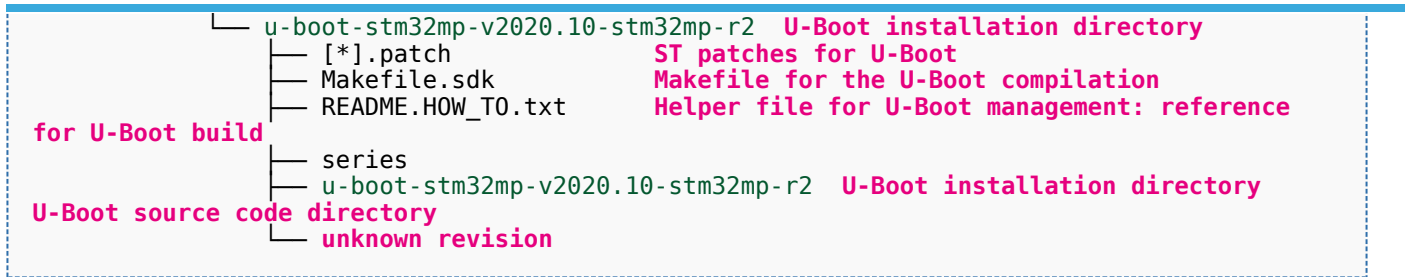
```

└─ tf-a-stm32mp-v2.4-stm32mp-r2  TF-A installation directory
│ ├── [*].patch           ST patches for TF-A
│ ├── tf-a-stm32mp-v2.4-stm32mp-r2  TF-A source code directory
│ ├── Makefile.sdk         Makefile for the TF-A compilation
│ └─ README.HOW_TO.txt     Helper file for TF-A management: reference
code directory
└─ for TF-A build
├─ series
└─ tf-a-stm32mp-v2.4-stm32mp-r2-r0.tar.gz

```



Example of directory structure for Packages



Appendix A shows the structure of the Linux kernel, U-Boot, TF-A and OP-TEE OS installation directories after these software packages have been built.



5 Focus on the Distribution Package directory

The *Distribution-Package* directory contains all the OpenEmbedded layers required to get the source code of any STM32MPU Embedded Software component, as well as a build framework based on OpenEmbedded.

```

Distribution-Package
├── openstlinux-5.10-dunfell-mp1-21-11-17  OpenSTLinux distribution
│   └── layers
│       ├── meta-openembedded  Collection of layers for the OpenEmbedded-Core universe (Op
│       │   enEmbedded standard)
│       │   ├── [...]
│       │   └── meta-qt5
│       │       └── [...]  QT5 layer for OpenEmbedded (standard)

```

```

├── meta-st
│   ├── meta-st-openstlinux  STMicroelectronics layer that
│   │   contains the settings of the frameworks and images for the OpenSTLinux distribution
│   │   └── recipes-st
│   │       ├── images
│   │       │   └── st-image-core.bb  Core image for OpenSTLinux
│   │       └── st-image-weston.bb  Weston image with basic Wayland
│   │           support for OpenSTLinux distribution: recommended setup
│   │               ├── packagegroups
│   │               │   ├── [...]
│   │               └── [...]

```

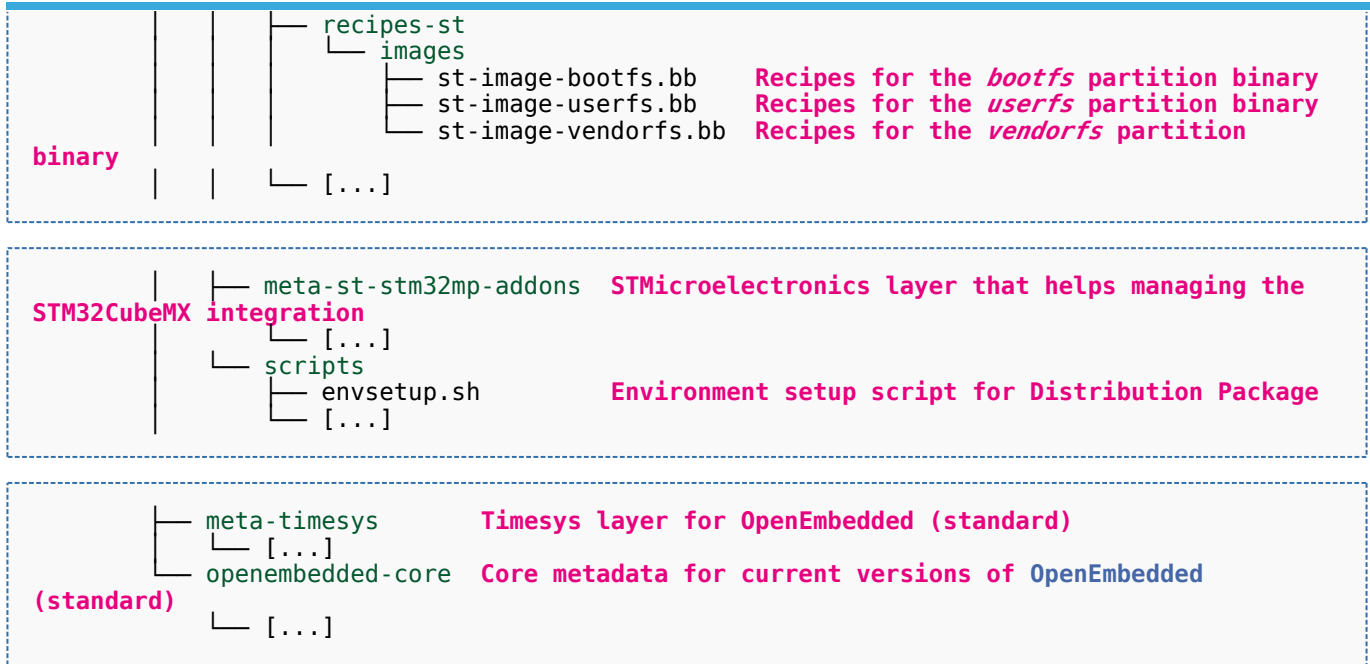
```

├── meta-st-stm32mp  STMicroelectronics layer that contains
│   │   the description of the BSP for the STM32 MPU devices
│   │   └── recipes-bsp
│   │       ├── alsa
│   │       │   └── [...]  Recipes for ALSA control configuration
│   │       └── drivers
│   │           └── [...]  Recipes for Vivante GCNANO GPU kernel
│   │               ├── trusted-firmware-a
│   │               │   ├── [...]  Recipes for TF-A
│   │               │   └── [...]
│   │               └── u-boot
│   │                   └── [...]  Recipes for U-Boot
│   │   └── recipes-extended
│   │       └── m4projects
│   │           ├── [...]  Recipes for STM32Cube MPU Package
│   │           └── [...]
│   │   └── recipes-graphics
│   │       ├── gcnano-userland
│   │       │   └── [...]  Recipes for Vivante libraries OpenGL
│   │       └── [...]
│   │   └── recipes-kernel
│   │       ├── linux
│   │       │   ├── [...]  Recipes for Linux kernel
│   │       │   └── [...]
│   │       └── linux-firmware
│   │           └── [...]  Recipes for Linux firmwares (example,
│   │               Bluetooth firmware)

```



Example of directory structure for Packages

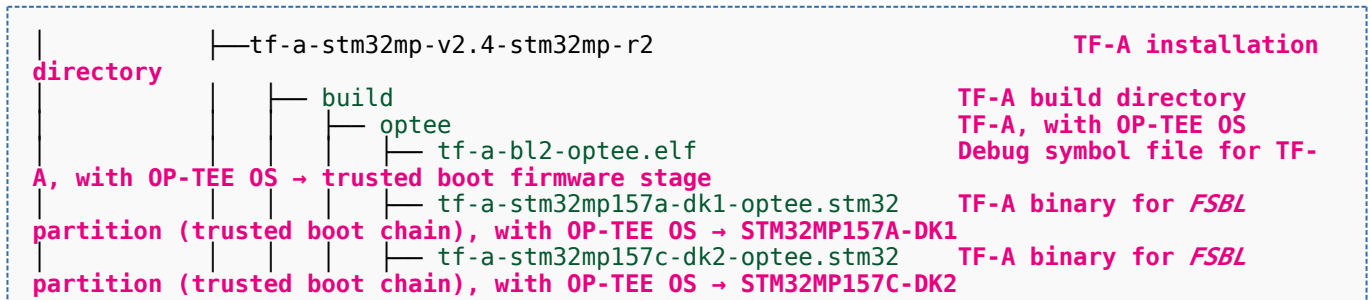
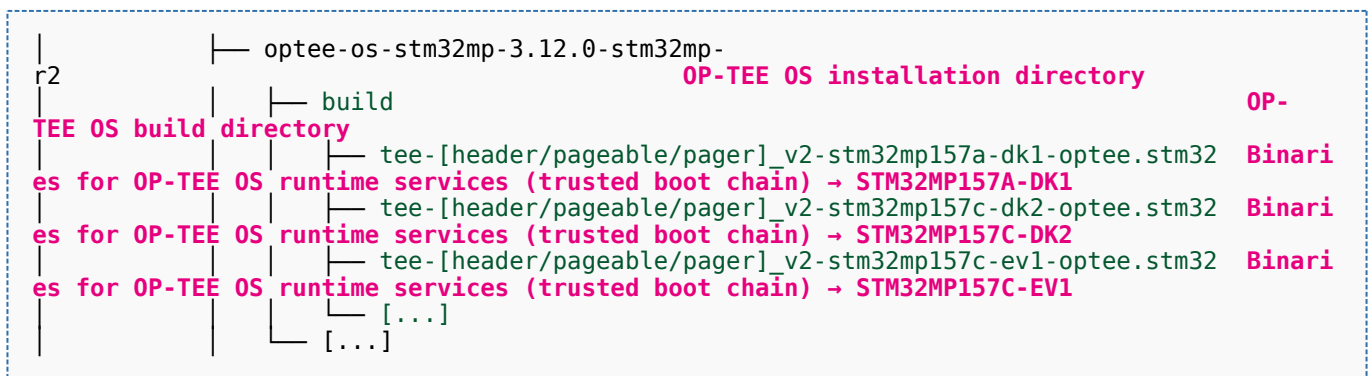
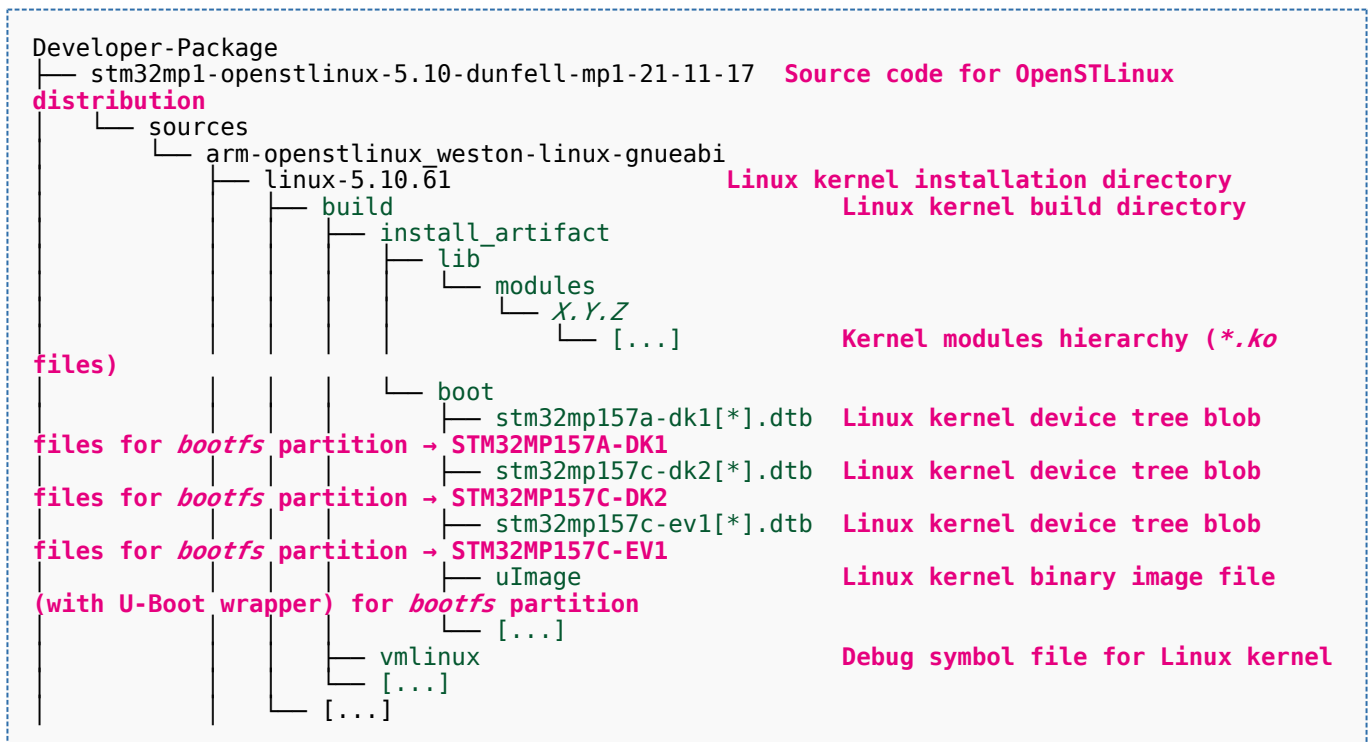


Appendix B shows the structure of the build directory.



6 Appendix A: directory structure after build (Developer Package)

Provided you have followed the recommendations of the *README.HOW_TO.txt* helper files to build the Linux kernel, the U-Boot and the TF-A, then the following new directories and files contain the build outputs.





Example of directory structure for Packages

partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-optee.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	trusted	TF-A, without OP-TEE OS
A → trusted boot firmware stage	tf-a-bl2-trusted.elf	Debug symbol file for TF-
A → trusted boot firmware stage	tf-a-bl32-trusted.elf	Debug symbol file for TF-
partition (trusted boot chain) → STM32MP157A-DK1	tf-a-stm32mp157a-dk1-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-DK2	tf-a-stm32mp157c-dk2-trusted.stm32	TF-A binary for <i>FSBL</i>
partition (trusted boot chain) → STM32MP157C-EV1	tf-a-stm32mp157c-ev1-trusted.stm32	TF-A binary for <i>FSBL</i>
	[...]	
	[...]	

	u-boot-stm32mp-v2020.10-stm32mp-r2	U-Boot
installation directory	build-basic	U-Boot build directory
for basic boot chain		
partition (basic boot chain) → STM32MP157A-DK1	u-boot-spl.stm32-stm32mp157a-dk1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-spl.stm32-stm32mp157c-dk2-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-spl.stm32-stm32mp157c-ev1-basic	U-Boot binary for <i>FSBL</i>
partition (basic boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-basic.img	U-Boot binary for <i>SSBL</i>
partition (basic boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-basic.img	U-Boot binary for <i>SSBL</i>
	build-optee	U-Boot build directory
for trusted boot chain, with OP-TEE OS		
Boot, with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-optee.stm32	U-Boot binary for <i>SSBL</i>
Boot, with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.elf	Debug symbol file for U-
partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-optee.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
for trusted boot chain	build-trusted	U-Boot build directory
Boot → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157A-DK1	u-boot-stm32mp157a-dk1-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-DK2	u-boot-stm32mp157c-dk2-trusted.stm32	U-Boot binary for <i>SSBL</i>
Boot → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.elf	Debug symbol file for U-
partition (trusted boot chain) → STM32MP157C-EV1	u-boot-stm32mp157c-ev1-trusted.stm32	U-Boot binary for <i>SSBL</i>
	[...]	
	[...]	



Example of directory structure for Packages



7 Appendix B: directory structure after build (Distribution Package)

Provided you have followed the build method explained in OpenSTLinux distribution, then the following new directories contain the build outputs.

As long as you did not modify the source code:

- the files in **STPurple** are the same as the ones available in the **Starter Package**: flash layout, binaries for *bootfs*, *rootfs*, *userfs* and *vendorfs* partitions
- the files in grey are the same as the ones available in the **Starter and Developer Packages**: binaries for *FSBL* and *SSBL* partitions, and for OP-TEE OS runtime services
- the files in **orange** are the same as the ones available in the **Developer Package**: Linux kernel image and device tree blobs, and debug symbol files

```
Distribution-Package/openstlinux-5.10-dunfell-mp1-21-11-17 /build-openstlinuxweston-
stm32mp/tmp-glibc/deploy
├── images
│   └── stm32mp1
│       ├── flashlayout_st-image-weston                               Flash layout
│       └── files (description of the partitions) for the supported flash devices
│           ├── FlashLayout_emmc_stm32mp157c-ev1-optee.tsv           Flash layout file
│           ├── for eMMC and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_emmc_stm32mp157c-ev1-trusted.tsv         Flash layout file
│           ├── for eMMC and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-optee.tsv     Flash layout file
│           ├── for NAND Flash and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nand-4-256_stm32mp157c-ev1-trusted.tsv   Flash layout file
│           ├── for NAND Flash and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-optee.tsv       Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-emmc_stm32mp157c-ev1-trusted.tsv     Flash layout file
│           ├── for NOR Flash (and eMMC) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-optee.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-nand-4-256_stm32mp157c-ev1-trusted.tsv  Flash layout file
│           ├── for NOR Flash (and NAND Flash) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-optee.tsv      Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
│           ├── FlashLayout_nor-sdcard_stm32mp157c-ev1-trusted.tsv     Flash layout file
│           ├── for NOR Flash (and microSD card) and trusted boot chain → STM32MP157C-EV1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-basic.tsv          Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-optee.tsv          Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157a-dk1-trusted.tsv        Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157A-DK1
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-basic.tsv          Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv          Flash layout file
│           ├── for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-dk2-trusted.tsv        Flash layout file
│           ├── for microSD card and trusted boot chain (recommended setup) → STM32MP157C-DK2
│           ├── FlashLayout_sdcard_stm32mp157c-ev1-basic.tsv          Flash layout file
│           ├── for microSD card and basic boot chain → STM32MP157C-EV1
│           └── FlashLayout_sdcard_stm32mp157c-ev1-optee.tsv          Flash layout file
```




Example of directory structure for Packages

```

for microSD card and trusted boot chain, with OP-TEE OS → STM32MP157C-EV1
├── FlashLayout_sdcard_stm32mp157c-ev1-trusted.tsv          Flash layout file
for microSD card and trusted boot chain (recommended setup) → STM32MP157C-EV1
├── [...]
├── scripts
└── create_sdcard_from_flashlayout.sh

```

```

partition ─── st-image-bootfs-openstlinux-weston-stm32mp1.ext4          Binary for bootfs
partition ─── st-image-userfs-openstlinux-weston-stm32mp1.ext4        Binary for userfs
s partition ─── st-image-vendorfs-openstlinux-weston-stm32mp1.ext4    Binary for vendorfs
partition ─── st-image-weston-openstlinux-weston-stm32mp1.ext4        Binary for rootfs
device tree blob files for bootfs partition → STM32MP157A-DK1
├── stm32mp157a-dk1[*].dtb                                             Linux kernel
device tree blob files for bootfs partition → STM32MP157C-DK2
├── stm32mp157c-dk2[*].dtb                                             Linux kernel
device tree blob files for bootfs partition → STM32MP157C-EV1
├── stm32mp157c-e[*].dtb                                              Linux kernel
TEE OS runtime services (trusted boot chain) → STM32MP157A-DK1
├── tee-[header/pageable/pager]_v2-stm32mp157a-dk1-optee.stm32       Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-DK2
├── tee-[header/pageable/pager]_v2-stm32mp157c-dk2-optee.stm32       Binaries for OP-
TEE OS runtime services (trusted boot chain) → STM32MP157C-EV1
├── tee-[header/pageable/pager]_v2-stm32mp157c-ev1-optee.stm32       Binaries for OP-
for TF-A, with OP-TEE OS → trusted boot firmware stage
├── tf-a-bl2-optee.elf                                                 Debug symbol file
for TF-A → trusted boot firmware stage
├── tf-a-bl2-trusted.elf                                               Debug symbol file
for TF-A → runtime software stage
├── tf-a-bl32-trusted.elf                                              Debug symbol file
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── tf-a-stm32mp157a-dk1-optee.stm32                                   TF-A binary for FS
├── tf-a-stm32mp157a-dk1-trusted.stm32                                TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157A-DK1
├── tf-a-stm32mp157c-dk2-optee.stm32                                   TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── tf-a-stm32mp157c-dk2-trusted.stm32                                TF-A binary for FS
BL partition (trusted boot chain) → STM32MP157C-DK2
├── tf-a-stm32mp157c-ev1-optee.stm32                                   TF-A binary for FS
BL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── tf-a-stm32mp157c-ev1-trusted.stm32                                TF-A binary for FS
FSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-spl.stm32-stm32mp157a-dk1-basic                             U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-spl.stm32-stm32mp157c-dk2-basic                             U-Boot binary for
FSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-spl.stm32-stm32mp157c-ev1-basic                             U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-basic.img                                    U-Boot binary for
for U-Boot, with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.elf                                  Debug symbol file
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.elf                                Debug symbol file
for U-Boot → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-optee.stm32                                U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157A-DK1
├── u-boot-stm32mp157a-dk1-trusted.stm32                              U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-basic.img                                    U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.elf                                  Debug symbol file

```



Example of directory structure for Packages

```

for U-Boot, with OP-TEE OS → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-optee.stm32          U-Boot binary for
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-trusted.elf        Debug symbol file
for U-Boot → STM32MP157C-DK2
├── u-boot-stm32mp157c-dk2-trusted.stm32      U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157C-DK2
├── u-boot-stm32mp157c-ev1-basic.img          U-Boot binary for
SSBL partition (basic boot chain) → STM32MP157C-EV1
├── u-boot-stm32mp157c-ev1-optee.elf         Debug symbol file
for U-Boot, with OP-TEE OS → STM32MP157C-EV1
├── u-boot-stm32mp157c-ev1-optee.stm32      U-Boot binary for
SSBL partition (trusted boot chain), with OP-TEE OS → STM32MP157C-EV1
├── u-boot-stm32mp157c-ev1-trusted.elf      Debug symbol file
for U-Boot → STM32MP157C-EV1
├── u-boot-stm32mp157c-ev1-trusted.stm32     U-Boot binary for
SSBL partition (trusted boot chain) → STM32MP157C-EV1
├── uImage                                   Linux kernel
binary image file (with U-Boot wrapper) for bootfs partition
├── vmlinux                                  Debug symbol file
for Linux kernel
├── [...]
└── [...]

```

Das U-Boot -- the Universal Boot Loader (see [U-Boot_overview](#))