



life.augmented

OpenSTLinux : Starter package_classroom

Johnson Zhang

March 2021

Agenda

1 Software Packages

5 References

2 Starter Package structure

3 Boot chain

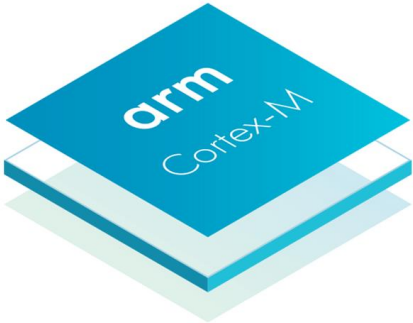
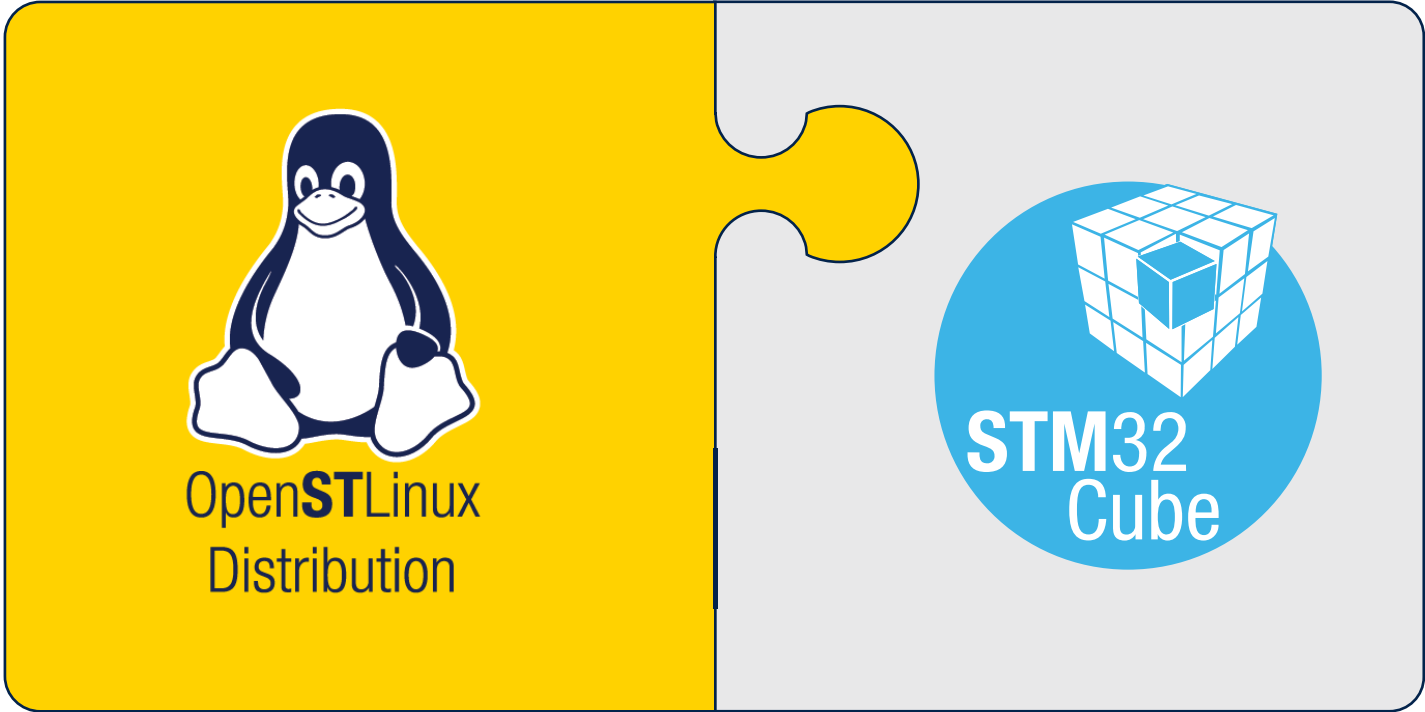
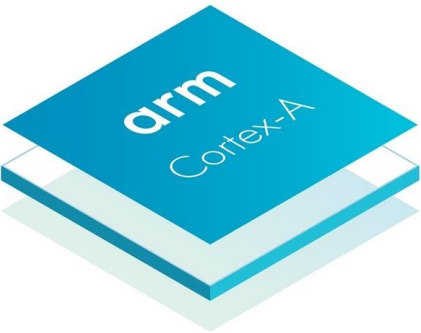
4 STM32MP15x Programmer

Software Packages



STM32MP1 Embedded Software Distribution

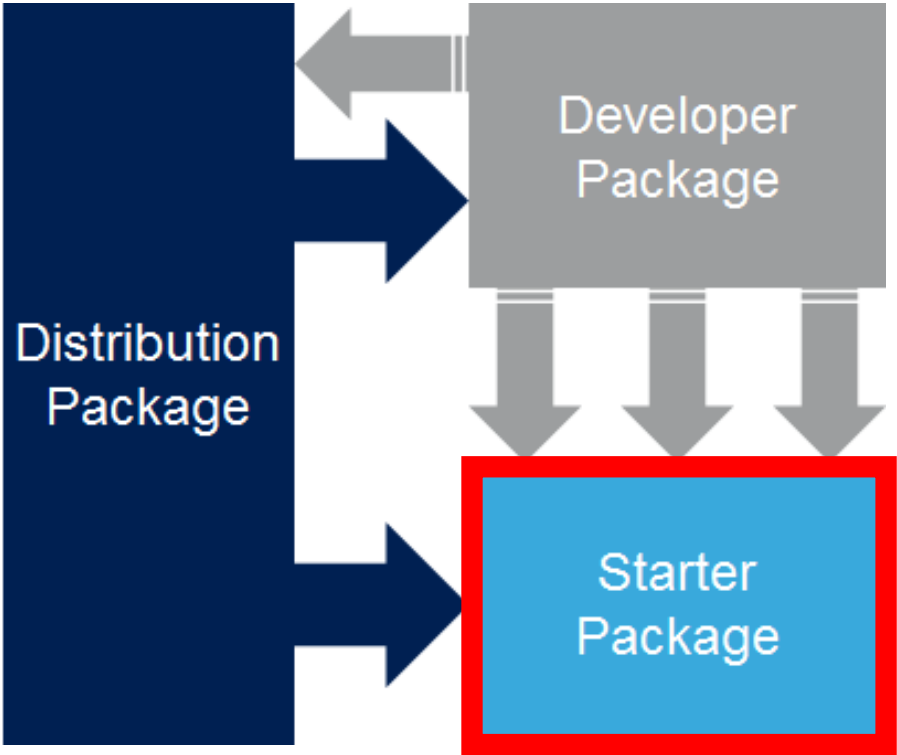
A fully Integrated Design Suite Leveraging the STM32Cube Environment



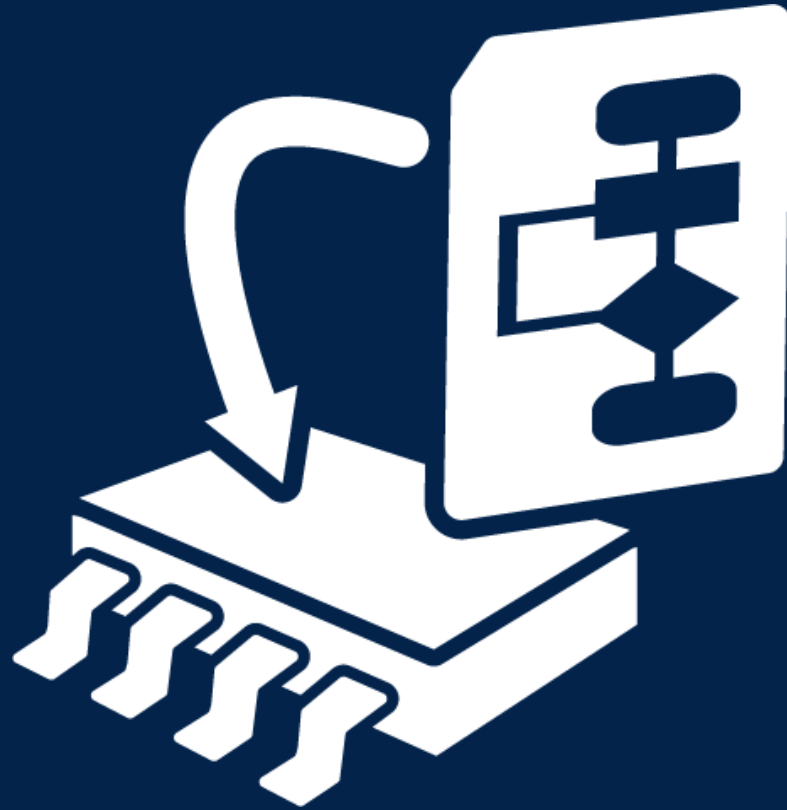
One distribution, three packages



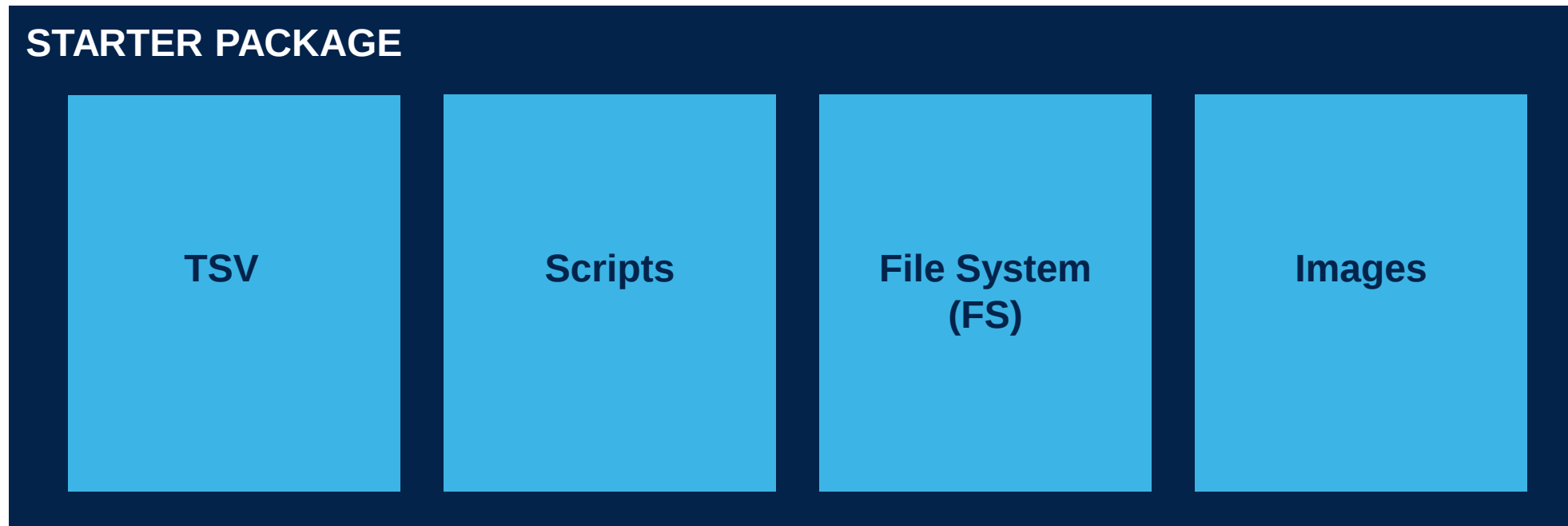
- Open STLinux
 - Starter Package
 - To quickly and easily start with any STM32MP1
 - Developer Package
 - To add your own developments on top of the STM32MP1 Embedded Software distribution
 - Distribution Package
 - To create your own Linux® distribution, your own Starter Package and your own Developer Package



Starter Package



- The starter package is composed by four kind of files.



TSV Files

A TSV file is the flash layout describing the flash mapping. STM32CubeProgrammer uses the TSV file as an input configuration file to be able to flash a product.

One TSV file describe one product flash configuration, you have to use one TSV for each flash configuration/product.

TSV file can be modified and edited as a TXT file.

TSV file can also be used/modified thanks to the STM32Programmer GUI.

#Opt	Id	Name	Type	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-ev1-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-openstlinux-weston-stm32mp1.ext4

#Opt	Id	Name	Type	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-ev1-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-openstlinux-weston-stm32mp1.ext4

OPT field :

- '-' : no action
- 'P' : update = program the partition or the flash device
- 'PE' : do not update (also 'EP') : allow GPT partitioning with empty partition for block device but equivalent to '-' for RAW flash device
- 'PD' : delete and update (also 'DP')
- 'PDE' : delete and keep empty (also 'PED' / 'DPE' / 'DEP' / 'EPD' / 'EDP')

#Opt	Id	Name	Type	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-ev1-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-openstlinux-weston-stm32mp1.ext4

ID fields :

- Two ranges :
 - 0x01 to 0x0F Boot partitions with STM32 header: SSBL, FSBL, other (TEE, Cortex-M4 firmware)
 - 0x10 to 0xF0 user partitions programmed without header (uimage, dtb, rootfs, vendorfs, userfs)
- Reserved Id :
 - 0x00 FlashLayout (used internally, cannot be used in FlashLayout file)
 - 0x01 FSBL (first copy) : used by ROM code (load in RAM)
 - 0x03 SSBL : used by FSBL=TF-A (load in RAM)
 - 0xF1 to 0xFD "virtual partition": used internally

#Opt	Id	Name	Type	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-ev1-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-openstlinux-weston-stm32mp1.ext4

Name of the alternate setting of the USB DFU for U-Boot enumeration. This is a string descriptor that indicates the target memory segment. This is also the name of the Block device GPT partition: SD card /EMMC.

The requirements for the GPT partitions are:

- FSBL for SD card boot: the name must start with "fsbl"= fsbl, fsbl1, fsbl2... (ROM code requirement)
- SSBL for eMMC/SD card boot: the name must be "ssbl" (TF-A requirement)
- TEE partition names (when present) must use the names expected by TF-A ("teeh", "teed" and "teex")
- a partition named "rootfs".

#Opt	Id	Name	Type	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-ev1-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-openstlinux-weston-stm32mp1.ext4

Type fields :

1. Binary : raw data
2. Binary(N) : raw data duplicate N times (used for MTD only)
3. FileSystem :
 - GPT : Linux filesystem data (EXT4, EXT2, FAT)
 - MTD : raw data
4. System :
 - GPT : Linux file system marked as bootable by u-boot
 - MTD : UBI file system
5. RAWImage :
 - Write a raw image on all the device.

#Opt	Id	Name	Type	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-ev1-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-openstlinux-weston-stm32mp1.ext4

IP fields, select the targeted device and the instance (starting at 0) as defined by U-Boot device tree.

Values :

- mmc + instance : 'mmc0' : EMMC or SD card on SDMMC.
- nor + instance : 'nor0' : NOR on QUADSPI.
- nand + instance : 'nand0' : Parallel NAND Flash memories on FMC.
- none : Partition only used to load the binary in RAM.

Two or more devices can be mixed in the same TSV file as for example : fsbl1 in nor0 and rootfs in mmc0.

#Opt	Id	Name	Type	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-ev1-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-openstlinux-weston-stm32mp1.ext4

Offset field :

- The supported values are:
 - boot1: first boot area partition of eMMC (offset is 0x0)
 - boot2: second boot area partition of eMMC (offset is 0x0)
 - Offset in bytes: offset in Flash memory area (in the user data area for eMMC)

The partitions are contiguous (no holes in Flash memory).

The last partition continues until the end of the selected Flash memory.

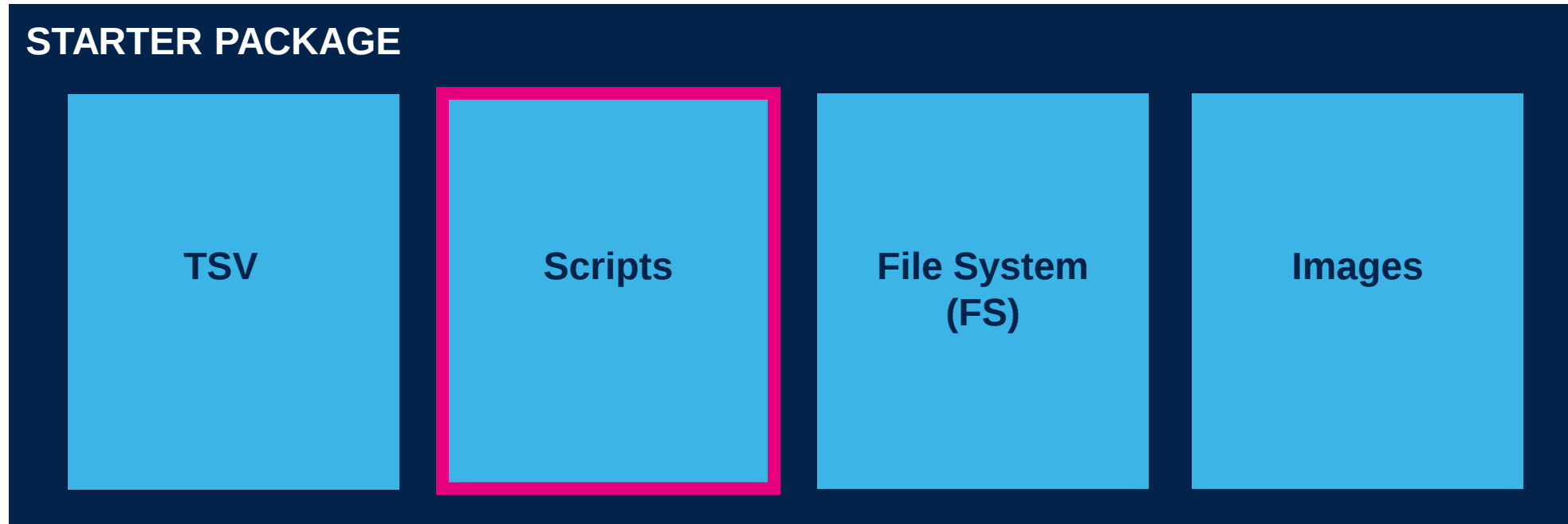
To reduce the size of the last partition, use an 'Empty' partition and leave it unused.

All the partitions need to be present in the FlashLayout file, even if they are not selected or empty.

Then the offset and size of each partition are compared with:

- pre-existing GPT partitioning, for updates on block devices (eMMC or SD card).
- predefined partitioning for MTD devices (NOR and NAND): see mtdparts environment variable in U-Boot for more information.

In case of partition size error, compare the existing partition size in U-Boot with the offset in the FlashLayout file.



One script is available to generate a full SDcard image (raw image).

File system

Four file systems are created in the starter package : bootfs, vendorfs, rootfs and userfs.

Two partition method are used depending of the flash type:

- GPT for Emmc and Sdcard
- UBI for NAND.
- No partition with NOR.

Two types of file system are used : EXT4 (with GPT) and UBIFS (with UBI).

For NOR, FSBL and SSBL are used in raw binaries (STM32 files) at a given offset with no partition and no file system.

Flash partitions (minimal)

Size	Component	Comment
Remaining area	userfs	The user file system contains user data and examples
768MB	rootfs	Linux root file system contains all user space binaries (executable, libraries, ...) and kernel modules
16MB	vendorfs	This partition is preferred to the rootfs to put third parties proprietary binaries and ensure that they are not contaminated by any open source licence, such as GPL v3
64MB	bootfs	The boot file system contains: - (option) the init ram file system, that can be copied to the external RAM and used by Linux before mounting a fatter rootfs - Linux kernel device tree (can be in a Flattened Image Tree - FIT) - Linux kernel U-Boot image (can be in a Flattened Image Tree - FIT) - For all flashes but the NOR: the boot loader splash screen image, displayed by U-Boot - U-Boot distro config file extlinux.conf (can be in a Flattened Image Tree – FIT)
2MB	ssbl	The Second Stage Boot Loader (SSBL) is U-Boot, with its device tree blob (dtb) appended at the end
256kB to 512kB (*)	fsbl	The First Stage Boot Loader is ARM Trusted Firmware (TF-A) or U-Boot Secondary Program Loader (SPL), with its device tree blob (dtb) appended at the end. At least two copies are embedded. Note: due to ROM code RAM needs, FSBL payload is limited to 247kB.

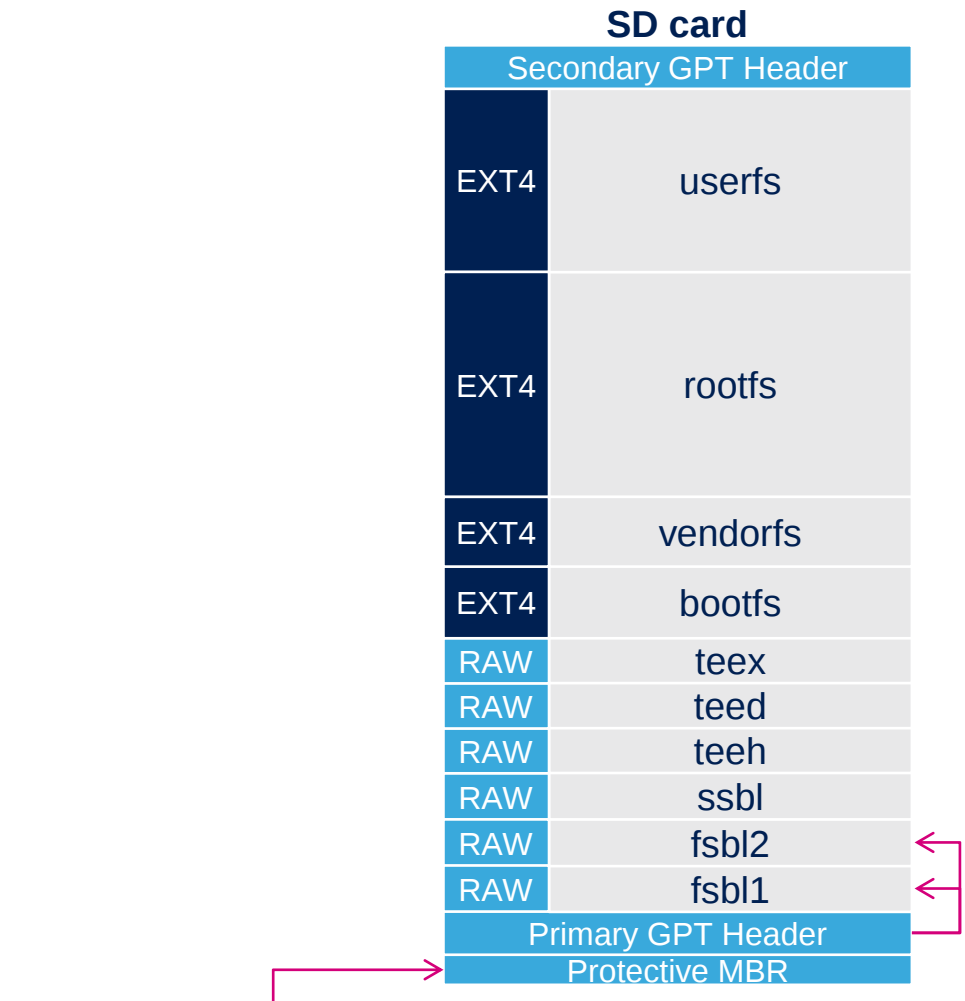
(*): the partition size depends on the flash technology, to be aligned on block erase size for NOR (256kB) / NAND (512kB)

Flash partitions (optional)

Size	Component	Comment
256kB (*)	logo	This partition contains the boot loader splash screen image while booting on NOR flash (for all other flashes, the image is stored in the bootfs partition)
256kB to 512kB (*)	teeh	OP-TEE header
256kB to 512kB (*)	teed	OP-TEE pageable code and data
256kB to 512kB (*)	teex	OP-TEE pager

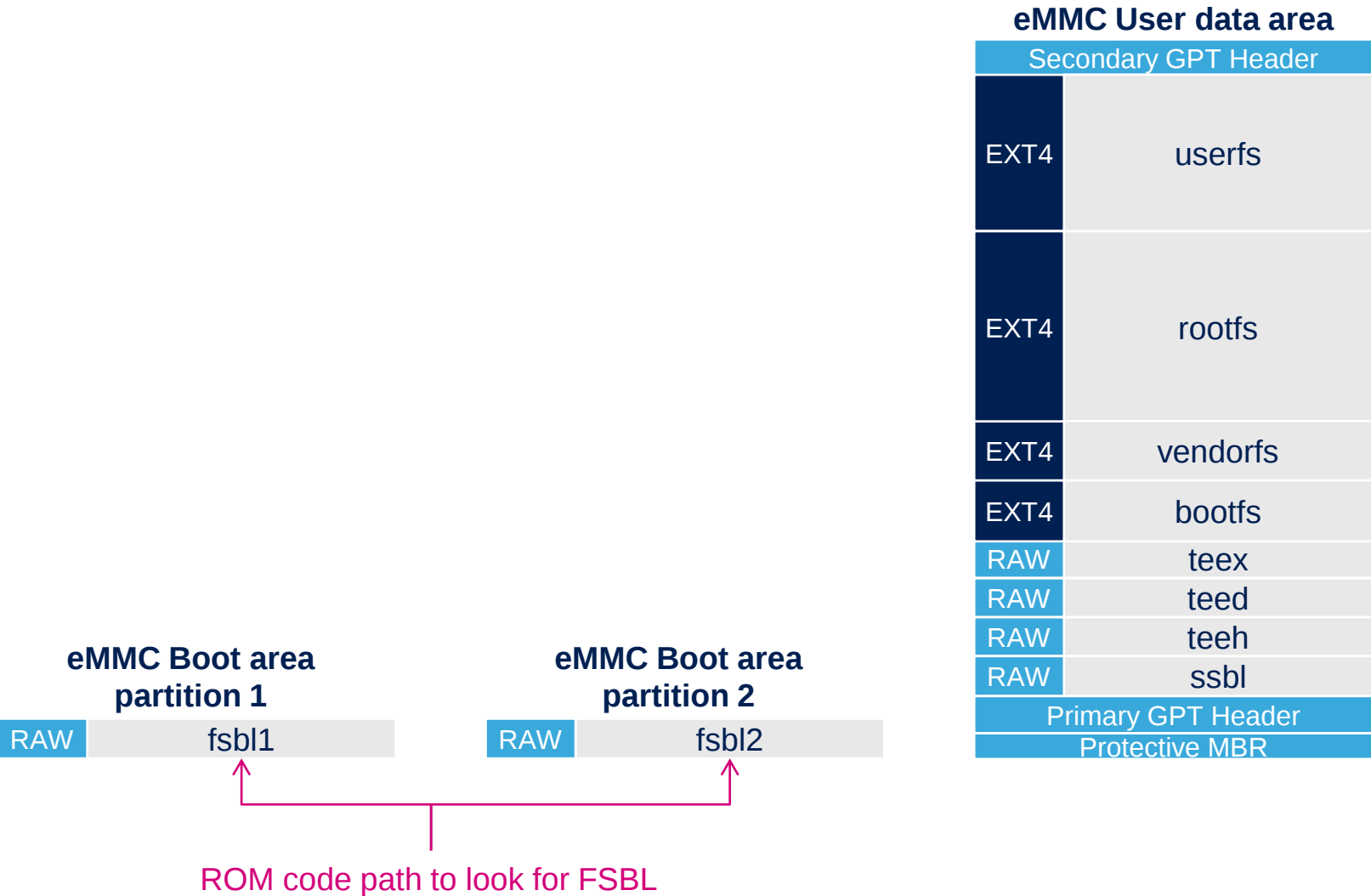
(*): the partition size depends on the flash technology, to be aligned on block erase size for NOR (256kB) / NAND (512kB)

SD card memory mapping



ROM code path to look for FSBL

eMMC memory mapping



NOR memory mapping

SD card	
Secondary GPT Header	
EXT4	userfs
EXT4	rootfs
EXT4	vendorfs
EXT4	bootfs
Primary GPT Header	
Protective MBR	

Note: SD card used as second stage boot device because the NOR flash is too small to contain Linux file systems. It is possible to use another second stage boot device, like eMMC or NAND.

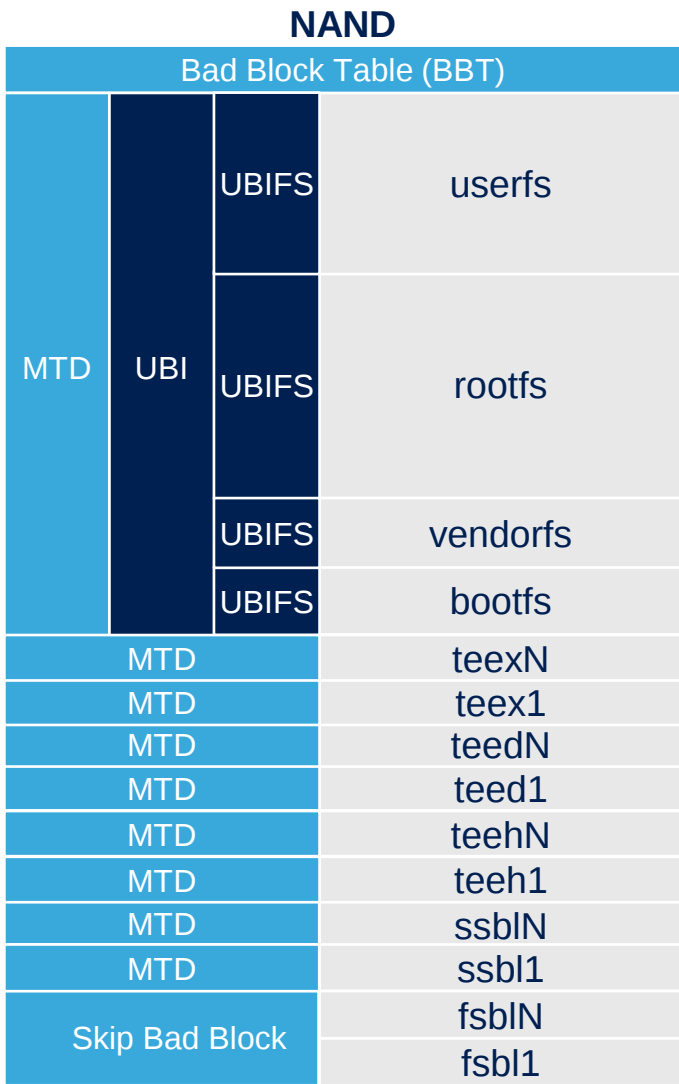
QSPI NOR	
RAW	teex
RAW	teed
RAW	teeh
RAW	logo
RAW	ssbl
RAW	fsbl2
RAW	fsbl1

Offset 256kB

Offset 0

ROM code path to look for FSBL

NAND memory mapping



Note: SSBL and OP-TEE partitions may move to UBI format when FSBL supports it

Note: in the Skip Bad Block area, the number of copies and the margin have to be defined in STM32CubeProgrammer flash layout, depending on the product expected life time and firmware update strategy

ROM code path to look for FSBL

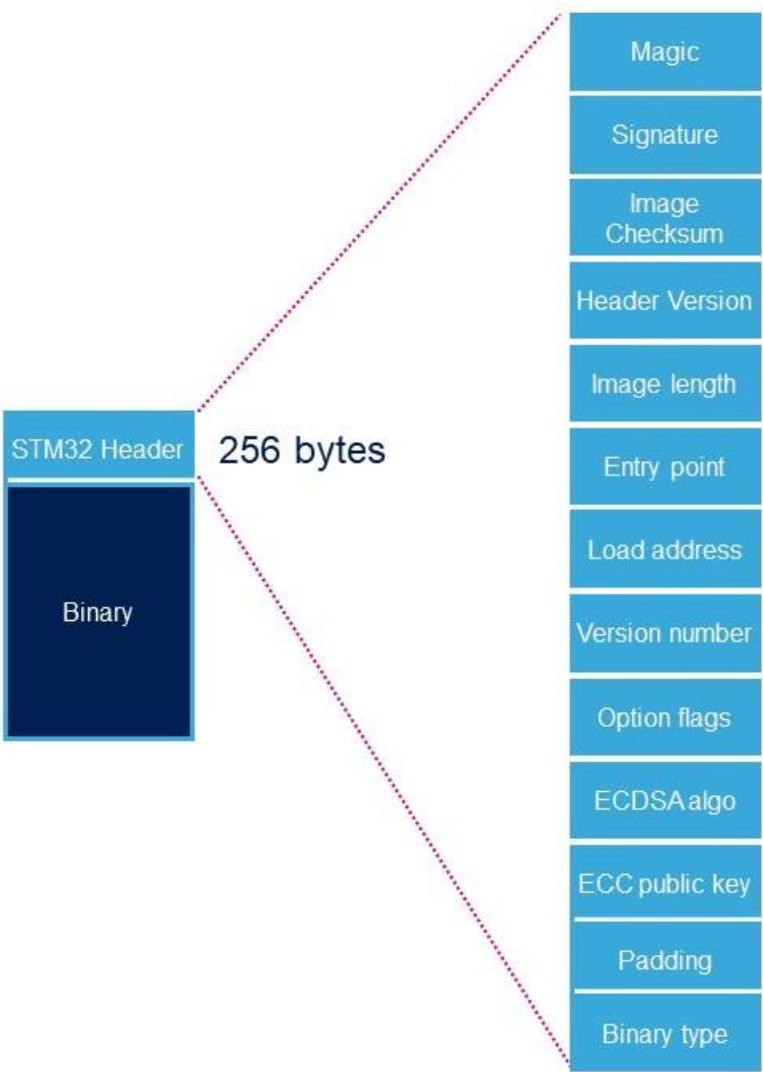
Binaries format

Two kind of raw binaries :

- STM32 binary (.stm32)
- Raw binary format (.img)

STM32 binary is a ST proprietary format consisting of the concatenation of STM32 Header with a Raw binary.

The STM32 binary format is the only format supported by the bootrom and TF-A (N.B.: TF-A in OSL 3.0.0 will support FIT format and not any more STM32 format).



Transfer file to the board

There are several methods for transferring file to the board:

- Transfer the image using the CubeProgrammer.
- Transfer the image using the Ethernet link or USB Ethernet gadget:
 - `scp filename root@<board ip address>:/usr/local`
- Transfer the image thanks to USB mass storage feature in u-boot.
 - https://wiki.st.com/stm32mpu/wiki/How_to_use_USB_mass_storage_in_U-Boot
- You can also plug the SD-card in your Linux laptop or add the file before to flash the files system.
- Transfer the image using the UART link : (not recommended)
 - https://wiki.st.com/stm32mpu/wiki/How_to_transfer_a_file_over_serial_console

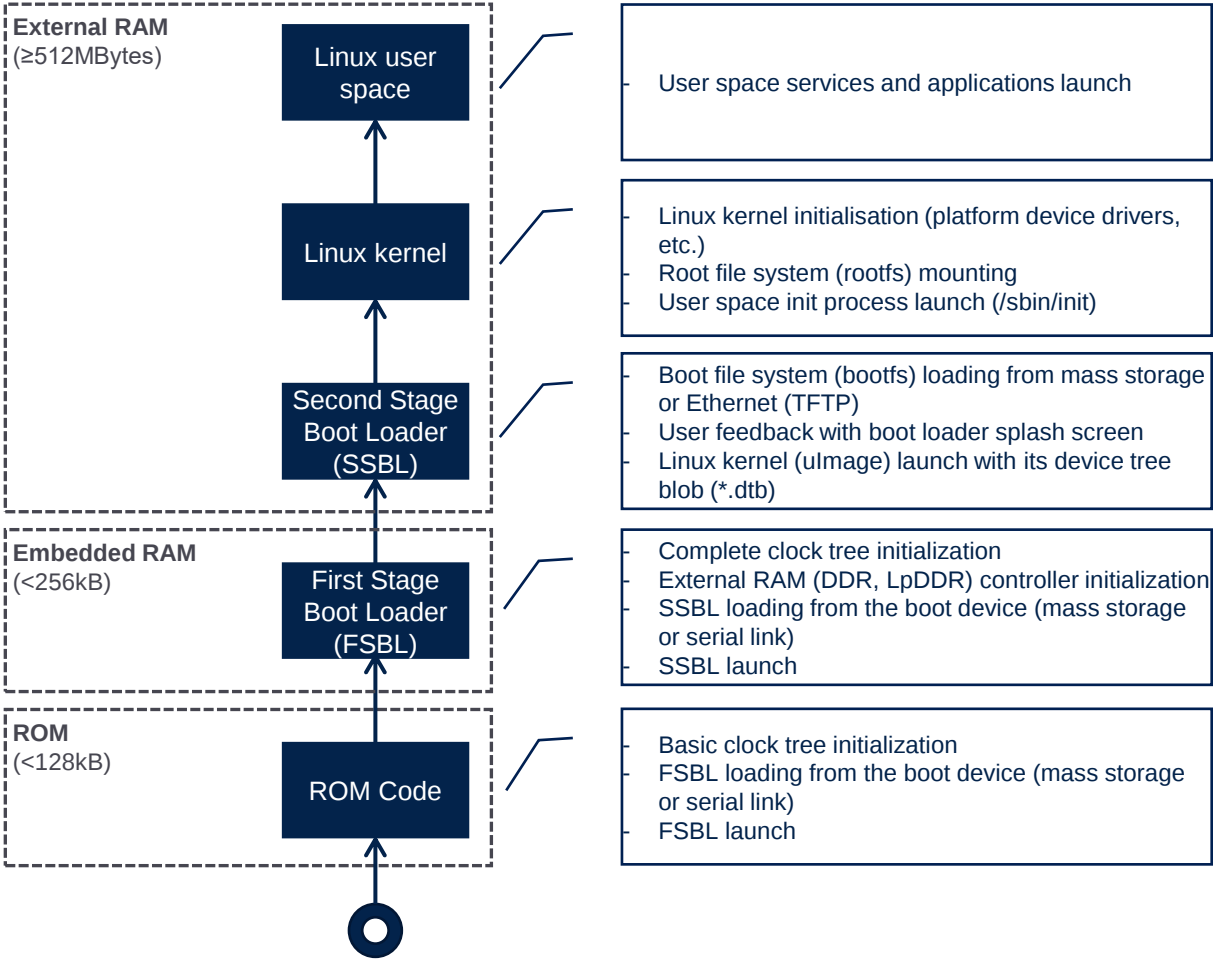
Boot chain



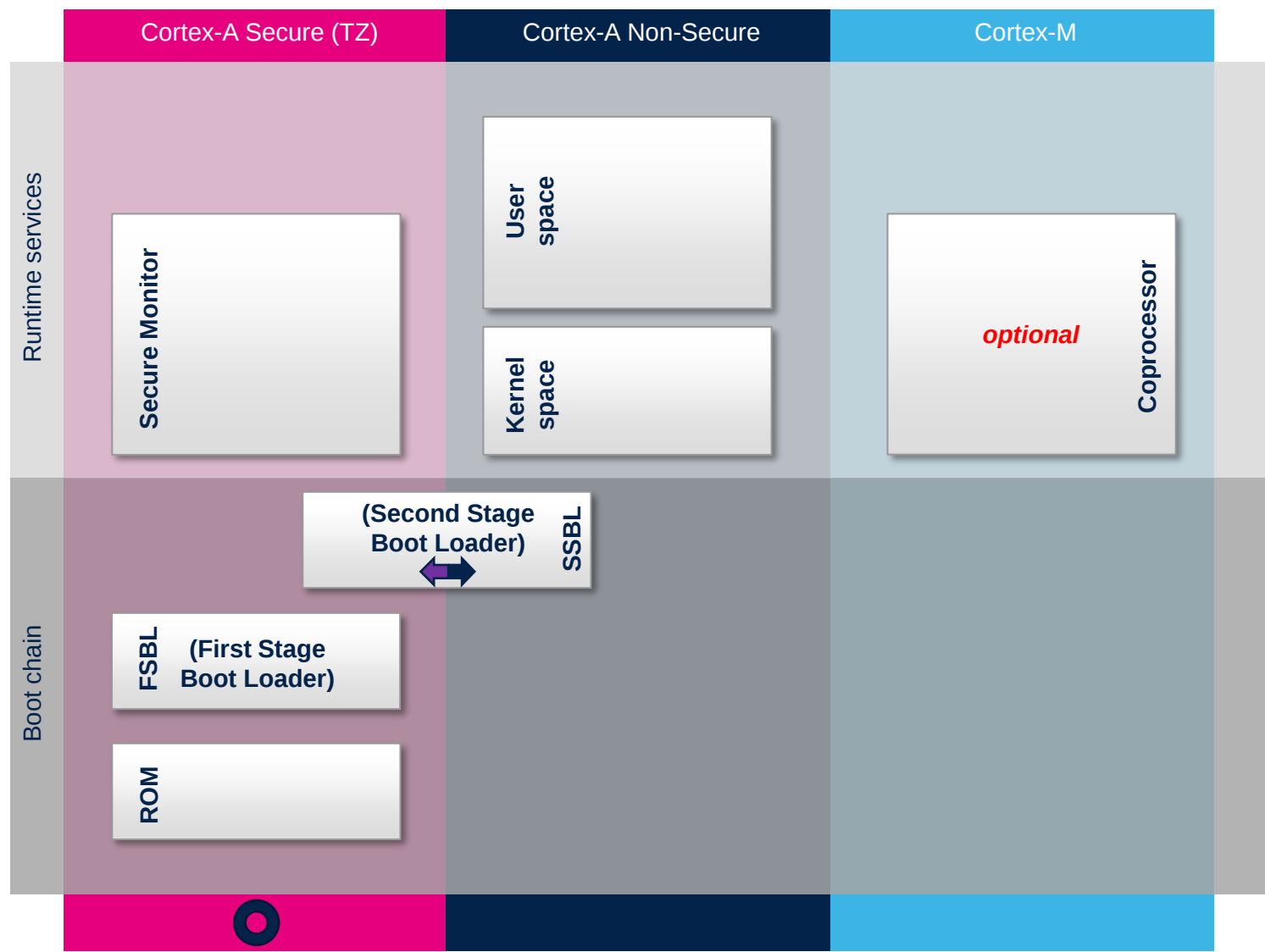
Standard Linux boot chain



Rocket image source:
https://fr.wikipedia.org/wiki/Ariane_6



STM32MP1 boot chain



3rd Party

STCommunity

Community + ST adds-on

→ Loads

→ (Loads &) Calls

→ (Loads &) Calls options

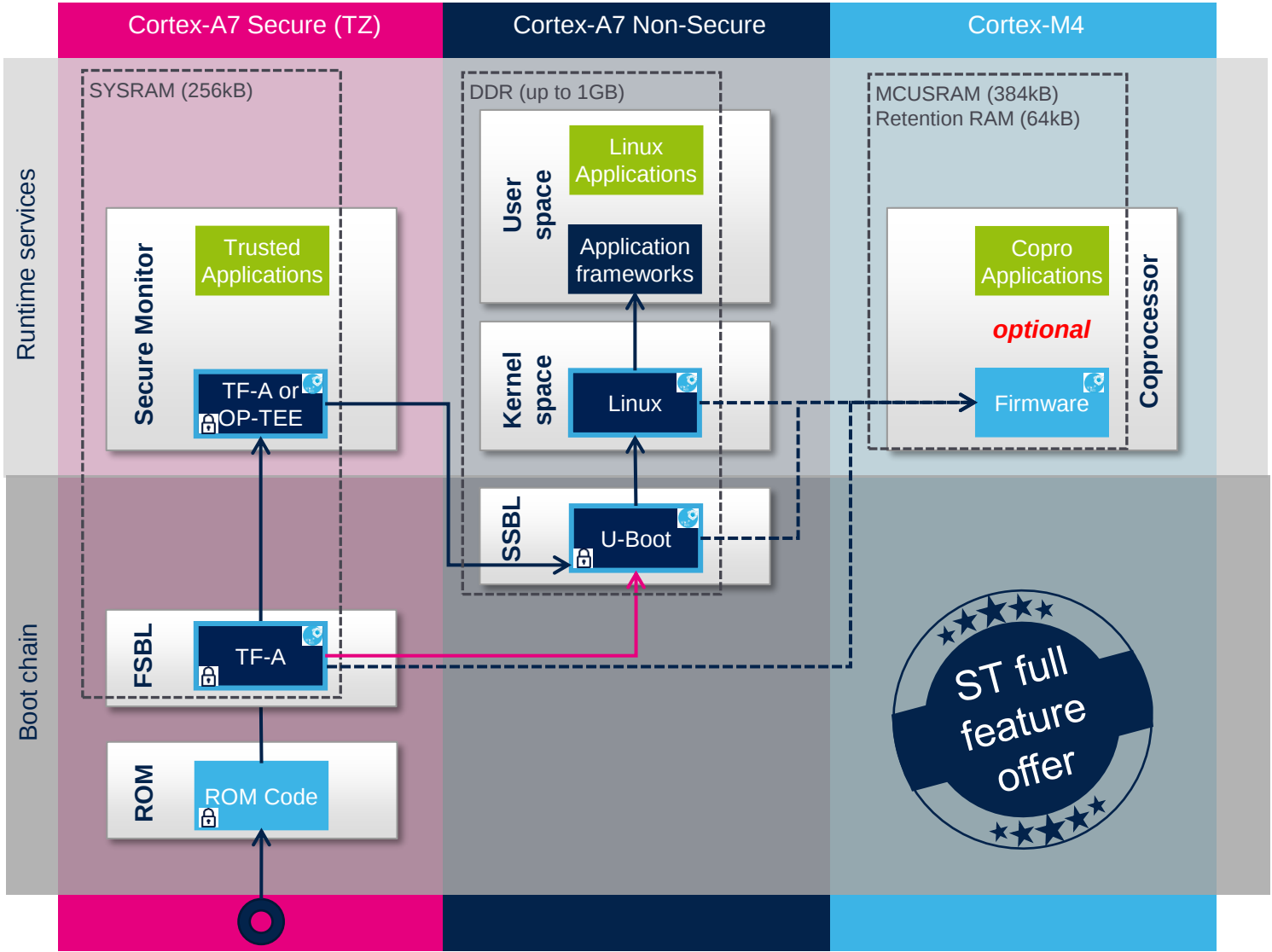
⚙️ : Configured via STM32CubeMX

🔒 : Authentication (optional)

Legend

STM32MP1 boot chain

TF-A is:
-BSD licence
-Trusted writing
-ARMv8 future proof



3rd Party

STCommunity

Community + ST adds-on

→ Loads

→ (Loads &) Calls

→ (Loads &) Calls options

⚙️ : Configured via STM32CubeMX

🔒 : Authentication (optional)

Legend

Boot mode selection

BOOT pins	TAMP_REG[20] (Force Serial)	OTP WORD 3 Primary boot source	OTP WORD 3 Secondary boot source	Boot source #1	Boot source #2 if #1 fails	Boot source if #2 fails
b000	x (don't care)	x (don't care)	x (don't care)	Serial	-	-
b001	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NOR	Serial	-
b010	!= 0xFF	0 (virgin)	0 (virgin)	eMMC	Serial	-
b011	!= 0xFF	0 (virgin)	0 (virgin)	FMC NAND	Serial	-
b100	x (don't care)	x (don't care)	x (don't care)	NoBoot	-	-
b101	!= 0xFF	0 (virgin)	0 (virgin)	SD-Card	Serial	-
b110	!= 0xFF	0 (virgin)	0 (virgin)	Serial	-	-
b111	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NAND	Serial	-
!= b100	!= 0xFF	Primary ¹	0 (virgin)	Primary ¹	Serial	-
!= b100	!= 0xFF	0 (virgin)	Secondary ¹	Secondary ¹	Serial	-
!= b100	!= 0xFF	Primary ¹	Secondary ¹	Primary ¹	Secondary ¹	Serial
!= b100	0xFF	x (don't care)	x (don't care)	Serial	-	-

Boot behavior can be modified using three IPs :

- Boot Pins (three pins)
- Tamper register (security)
- OTP (User configuration)

¹Primary and Secondary are fields of **OTP WORD3**.

Boot mode selection

BOOT pins	TAMP_REG[20] (Force Serial)	OTP WORD 3 Primary boot source	OTP WORD 3 Secondary boot source	Boot source #1	Boot source #2 if #1 fails	Boot source if #2 fails
b000	x (don't care)	x (don't care)	x (don't care)	Serial	-	-
b001	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NOR	Serial	-
b010	!= 0xFF	0 (virgin)	0 (virgin)	eMMC	Serial	-
b011	!= 0xFF	0 (virgin)	0 (virgin)	FMC NAND	Serial	-
b100	x (don't care)	x (don't care)	x (don't care)	NoBoot	-	-
b101	!= 0xFF	0 (virgin)	0 (virgin)	SD-Card	Serial	-
b110	!= 0xFF	0 (virgin)	0 (virgin)	Serial	-	-
b111	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NAND	Serial	-
!= b100	!= 0xFF	Primary ¹	0 (virgin)	Primary ¹	Serial	-
!= b100	!= 0xFF	0 (virgin)	Secondary ¹	Secondary ¹	Serial	-
!= b100	!= 0xFF	Primary ¹	Secondary ¹	Primary ¹	Secondary ¹	Serial
!= b100	0xFF	x (don't care)	x (don't care)	Serial	-	-

Depending of the configuration these three IPs the boot selected is not the same

¹Primary and Secondary are fields of OTP WORD3.

Boot mode selection

BOOT pins	TAMP_REG[20] (Force Serial)	OTP WORD 3 Primary boot source	OTP WORD 3 Secondary boot source	Boot source #1	Boot source #2 if #1 fails	Boot source if #2 fails
b000	x (don't care)	x (don't care)	x (don't care)	Serial	-	-
b001	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NOR	Serial	-
b010	!= 0xFF	0 (virgin)	0 (virgin)	eMMC	Serial	-
b011	!= 0xFF	0 (virgin)	0 (virgin)	FMC NAND	Serial	-
b100	x (don't care)	x (don't care)	x (don't care)	NoBoot	-	-
b101	!= 0xFF	0 (virgin)	0 (virgin)	SD-Card	Serial	-
b110	!= 0xFF	0 (virgin)	0 (virgin)	Serial	-	-
b111	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NAND	Serial	-
!= b100	!= 0xFF	Primary ¹	0 (virgin)	Primary ¹	Serial	-
!= b100	!= 0xFF	0 (virgin)	Secondary ¹	Secondary ¹	Serial	-
!= b100	!= 0xFF	Primary ¹	Secondary ¹	Primary ¹	Secondary ¹	Serial
!= b100	0xFF	x (don't care)	x (don't care)	Serial	-	-

The one with the highest priority is the boot pins, when these one are set at **b100**, in this case the STM32MP15x **doesn't** boot, No Boot.

¹Primary and Secondary are fields of OTP WORD3.

Boot mode selection

BOOT pins	TAMP_REG[20] (Force Serial)	OTP WORD 3 Primary boot source	OTP WORD 3 Secondary boot source	Boot source #1	Boot source #2 if #1 fails	Boot source if #2 fails
b000	x (don't care)	x (don't care)	x (don't care)	Serial	-	-
b001	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NOR	Serial	-
b010	!= 0xFF	0 (virgin)	0 (virgin)	eMMC	Serial	-
b011	!= 0xFF	0 (virgin)	0 (virgin)	FMC NAND	Serial	-
b100	x (don't care)	x (don't care)	x (don't care)	NoBoot	-	-
b101	!= 0xFF	0 (virgin)	0 (virgin)	SD-Card	Serial	-
b110	!= 0xFF	0 (virgin)	0 (virgin)	Serial	-	-
b111	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NAND	Serial	-
!= b100	!= 0xFF	Primary ¹	0 (virgin)	Primary ¹	Serial	-
!= b100	!= 0xFF	0 (virgin)	Secondary ¹	Secondary ¹	Serial	-
!= b100	!= 0xFF	Primary ¹	Secondary ¹	Primary ¹	Secondary ¹	Serial
!= b100	0xFF	x (don't care)	x (don't care)	Serial	-	-

The second with the highest priority is the tamper detection, when this one is set at 0xFF, in this case the STM32MP15x **allow a boot only in serial mode, USB or UART.**

¹Primary and Secondary are fields of OTP WORD3.

Boot mode selection

BOOT pins	TAMP_REG[20] (Force Serial)	OTP WORD 3 Primary boot source	OTP WORD 3 Secondary boot source	Boot source #1	Boot source #2 if #1 fails	Boot source if #2 fails
b000	x (don't care)	x (don't care)	x (don't care)	Serial	-	-
b001	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NOR	Serial	-
b010	!= 0xFF	0 (virgin)	0 (virgin)	eMMC	Serial	-
b011	!= 0xFF	0 (virgin)	0 (virgin)	FMC NAND	Serial	-
b100	x (don't care)	x (don't care)	x (don't care)	NoBoot	-	-
b101	!= 0xFF	0 (virgin)	0 (virgin)	SD-Card	Serial	-
b110	!= 0xFF	0 (virgin)	0 (virgin)	Serial	-	-
b111	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NAND	Serial	-
!= b100	!= 0xFF	Primary ¹	0 (virgin)	Primary ¹	Serial	-
!= b100	!= 0xFF	0 (virgin)	Secondary ¹	Secondary ¹	Serial	-
!= b100	!= 0xFF	Primary ¹	Secondary ¹	Primary ¹	Secondary ¹	Serial
!= b100	0xFF	x (don't care)	x (don't care)	Serial	-	-

¹Primary and Secondary are fields of **OTP WORD3**.

The third with the highest priority is the **OTP WORD 3**, Primary boot is used first if it's failed (or primary empty) the secondary is used, if it's failed the serial is used.

	0	No secondary boot source is defined
	1	FMC NAND
0		No primary boot source is defined
1		FMC NAND
2		QSPI NOR
3		eMMC
4		SD
5		QSPI NAND

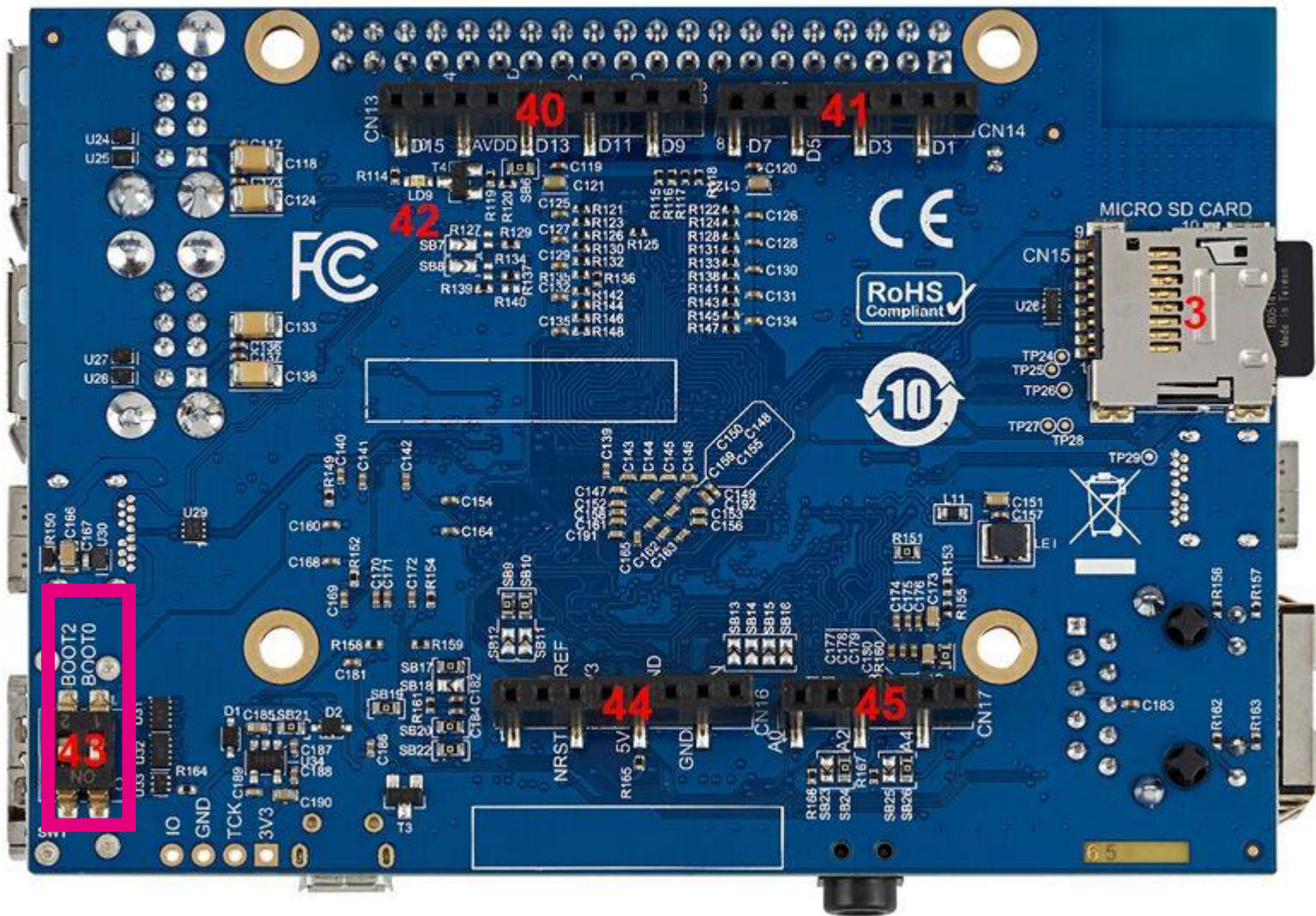
Boot mode selection

BOOT pins	TAMP_REG[20] (Force Serial)	OTP WORD 3 Primary boot source	OTP WORD 3 Secondary boot source	Boot source #1	Boot source #2 if #1 fails	Boot source if #2 fails
b000	x (don't care)	x (don't care)	x (don't care)	Serial	-	-
b001	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NOR	Serial	-
b010	!= 0xFF	0 (virgin)	0 (virgin)	eMMC	Serial	-
b011	!= 0xFF	0 (virgin)	0 (virgin)	FMC NAND	Serial	-
b100	x (don't care)	x (don't care)	x (don't care)	NoBoot	-	-
b101	!= 0xFF	0 (virgin)	0 (virgin)	SD-Card	Serial	-
b110	!= 0xFF	0 (virgin)	0 (virgin)	Serial	-	-
b111	!= 0xFF	0 (virgin)	0 (virgin)	QSPI NAND	Serial	-
!= b100	!= 0xFF	Primary ¹	0 (virgin)	Primary ¹	Serial	-
!= b100	!= 0xFF	0 (virgin)	Secondary ¹	Secondary ¹	Serial	-
!= b100	!= 0xFF	Primary ¹	Secondary ¹	Primary ¹	Secondary ¹	Serial
!= b100	0xFF	x (don't care)	x (don't care)	Serial	-	-

Finally, the boot pins are used.

¹Primary and Secondary are fields of **OTP WORD3**.

DK1 boot pins



Boot mode	Boot 0	Boot 2
Forced USB boot for flashing	0	0
Not supported	ON	0
Engineering boot	0	ON
microSD card	ON	ON

On DK board the Boot1 pin is grounded so always 0.

STM32MP15x Programmer

STM32 
CubeProgrammer

STM32CubeProgrammer is the official STMicroelectronics tool for creating partitions into any Flash device available on STM32 platforms.

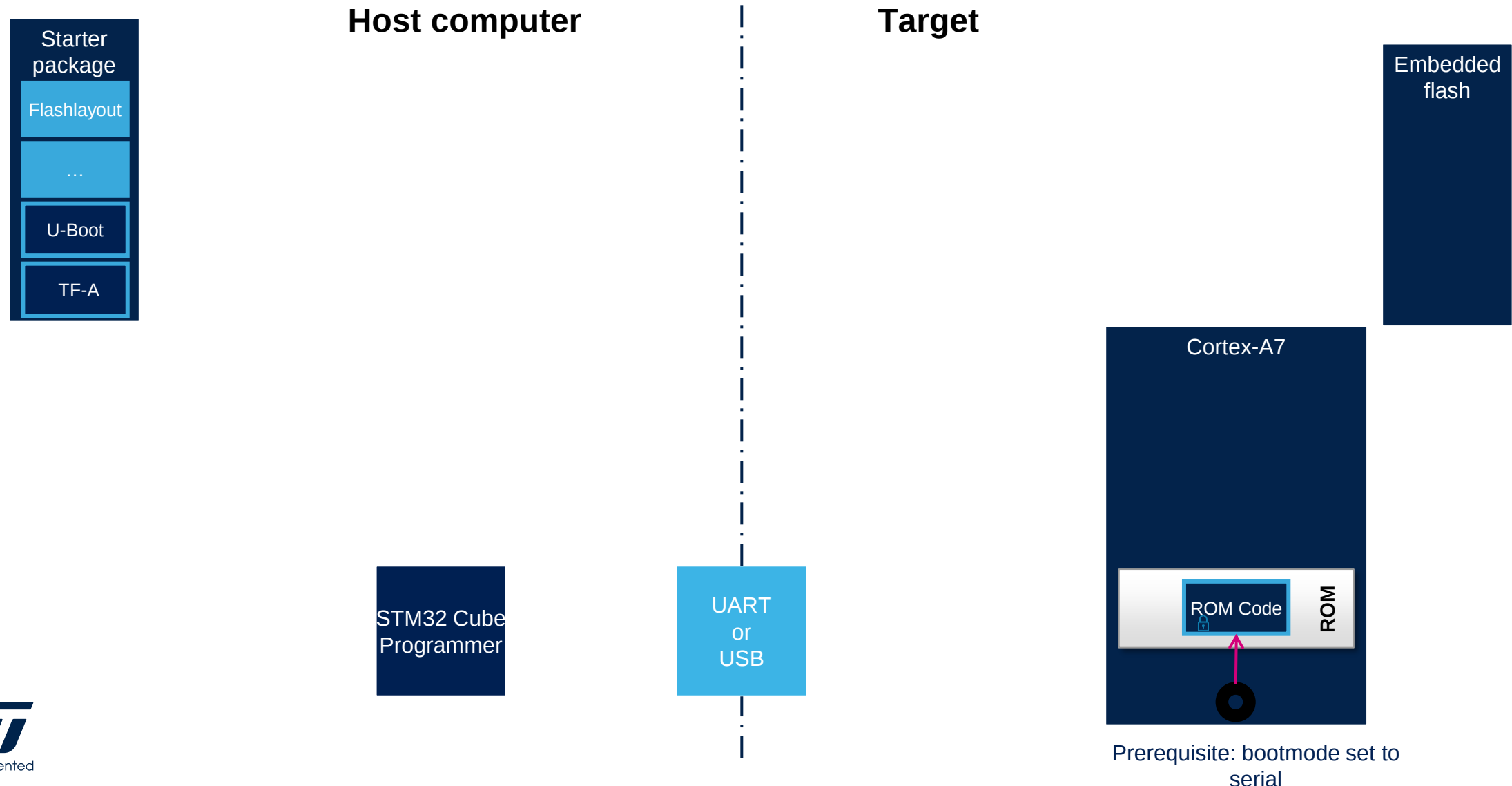
STM32CubeProgrammer allows populating and updating the partitions with the prebuilt binaries.

The connection between the host PC and the board can be done through **UART** or **USB** serial links.

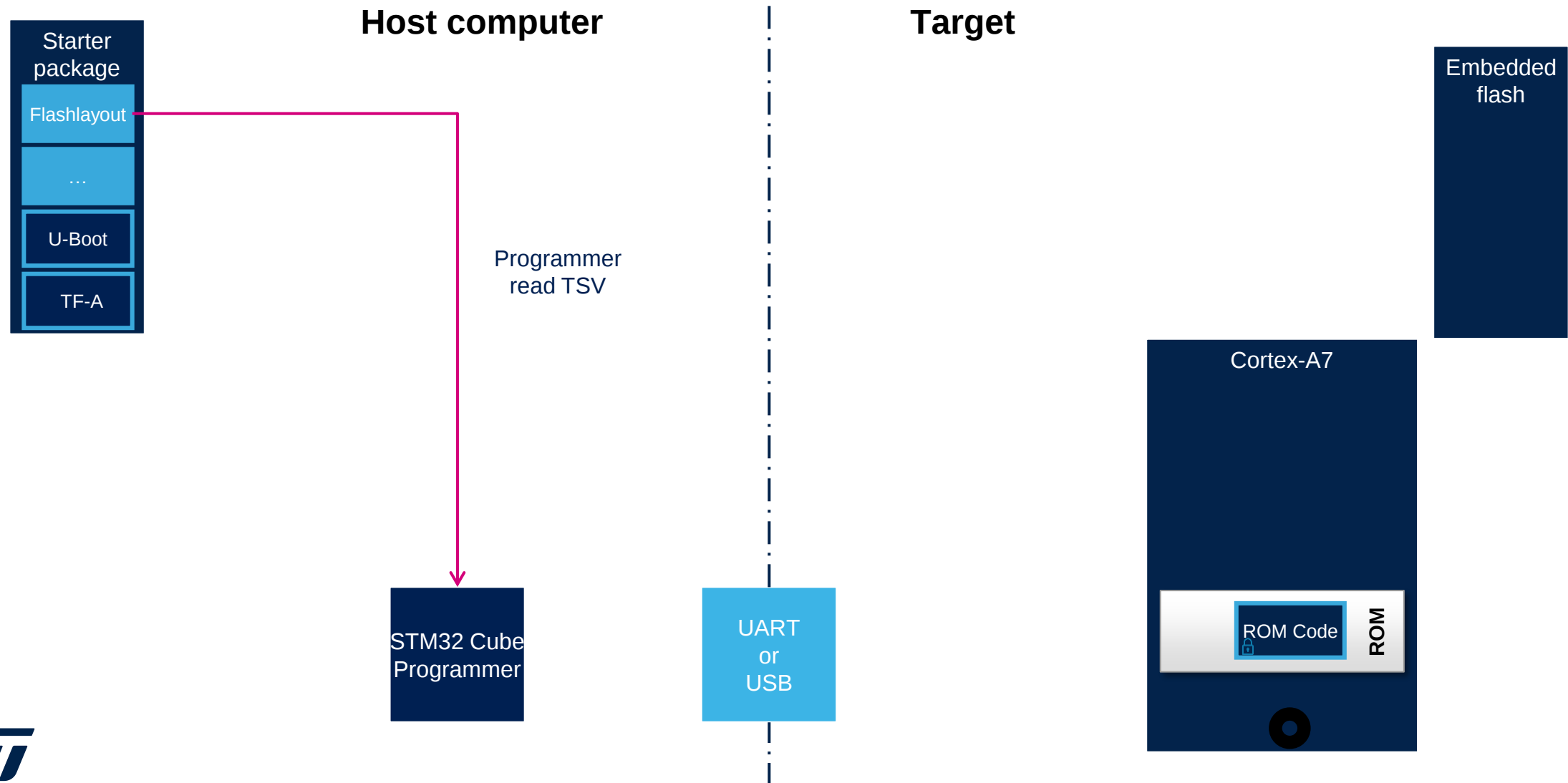
Any Flash device supported on STM32MPU15x boards can be flashed using STM32CubeProgrammer :

- microSDTM card
- eMMC
- NAND Flash memory
- NOR Flash memory

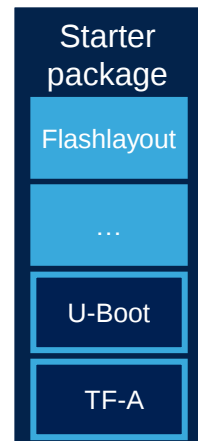
Programmer flow



Programmer flow



Programmer flow

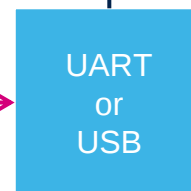


Host computer

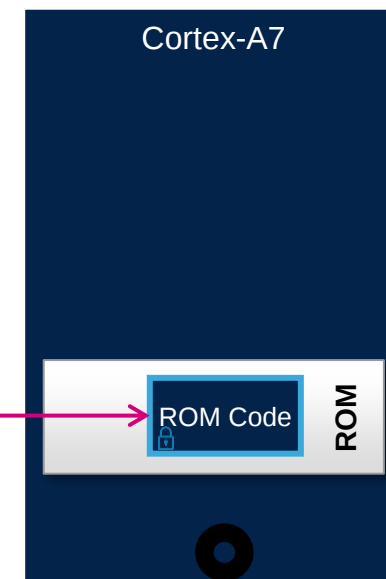
Target



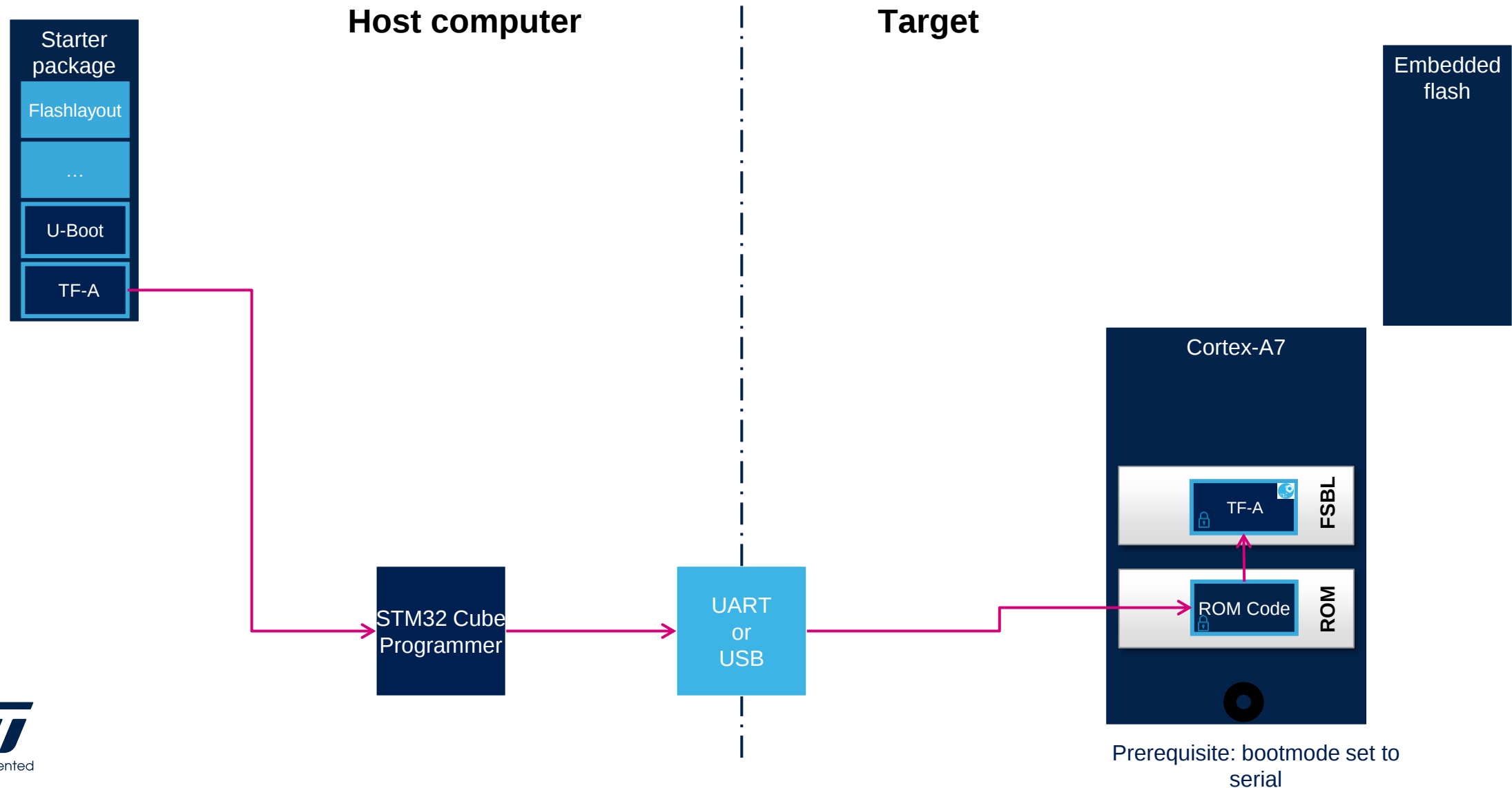
Init



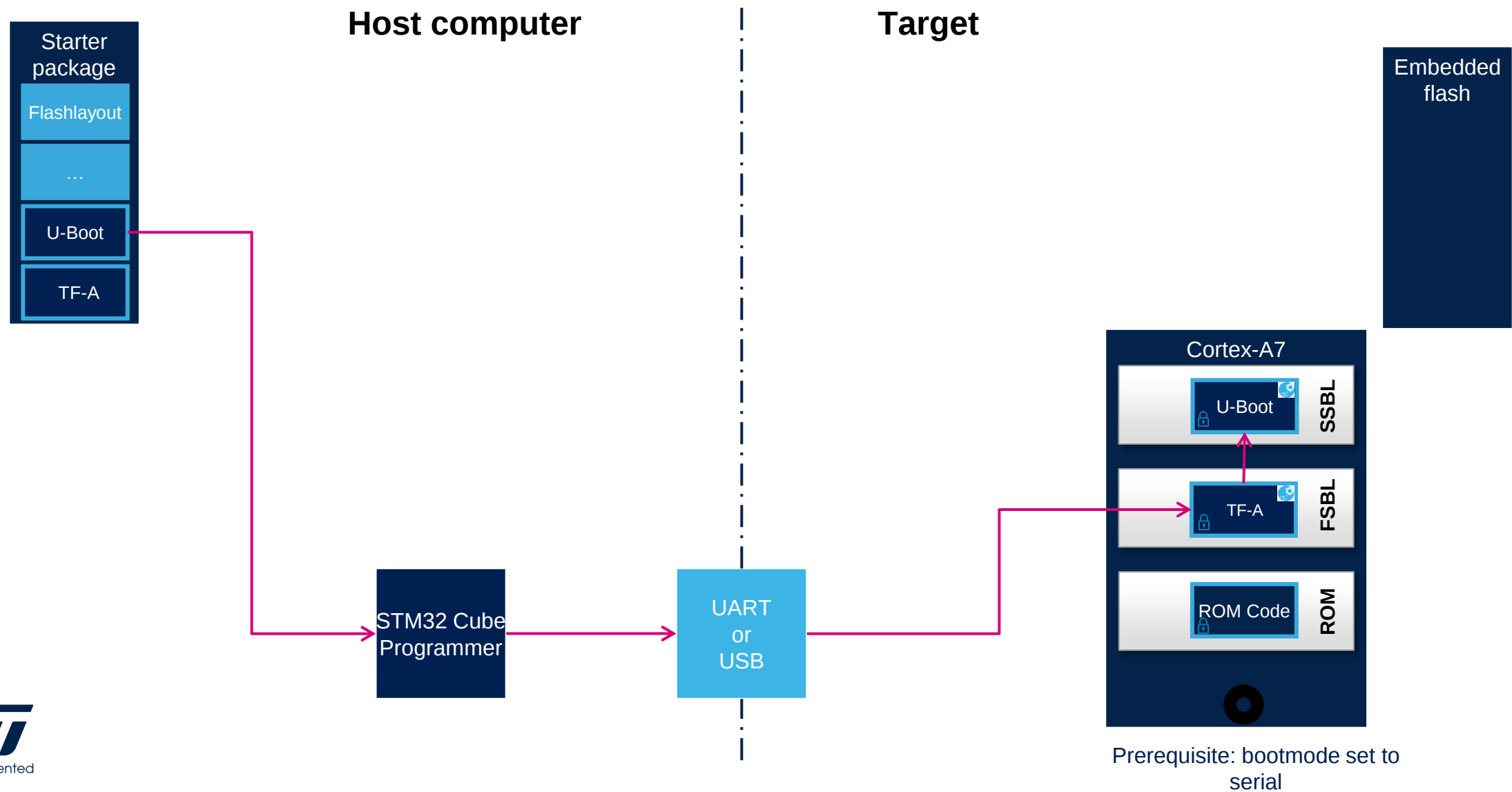
Init



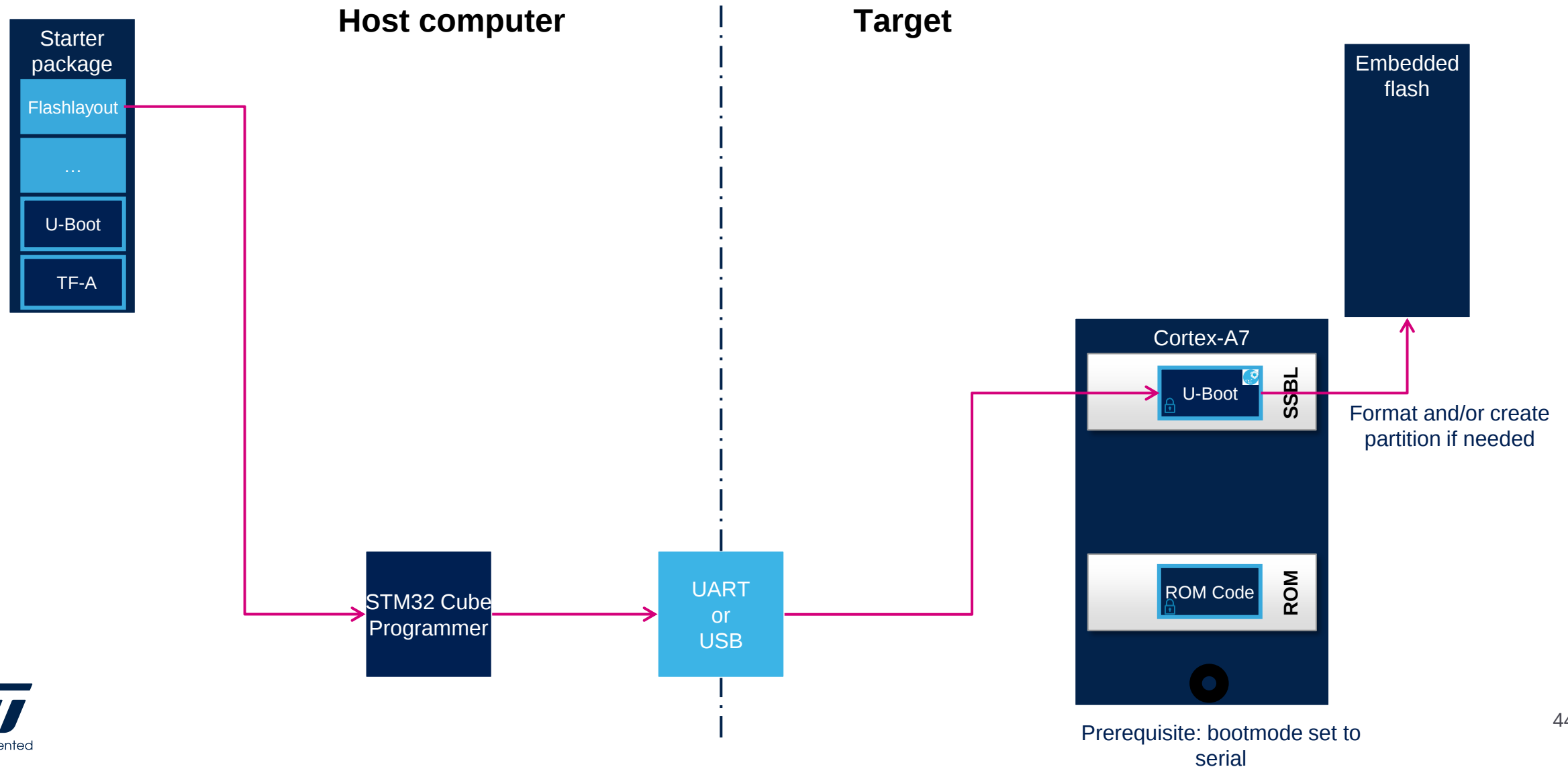
Programmer flow



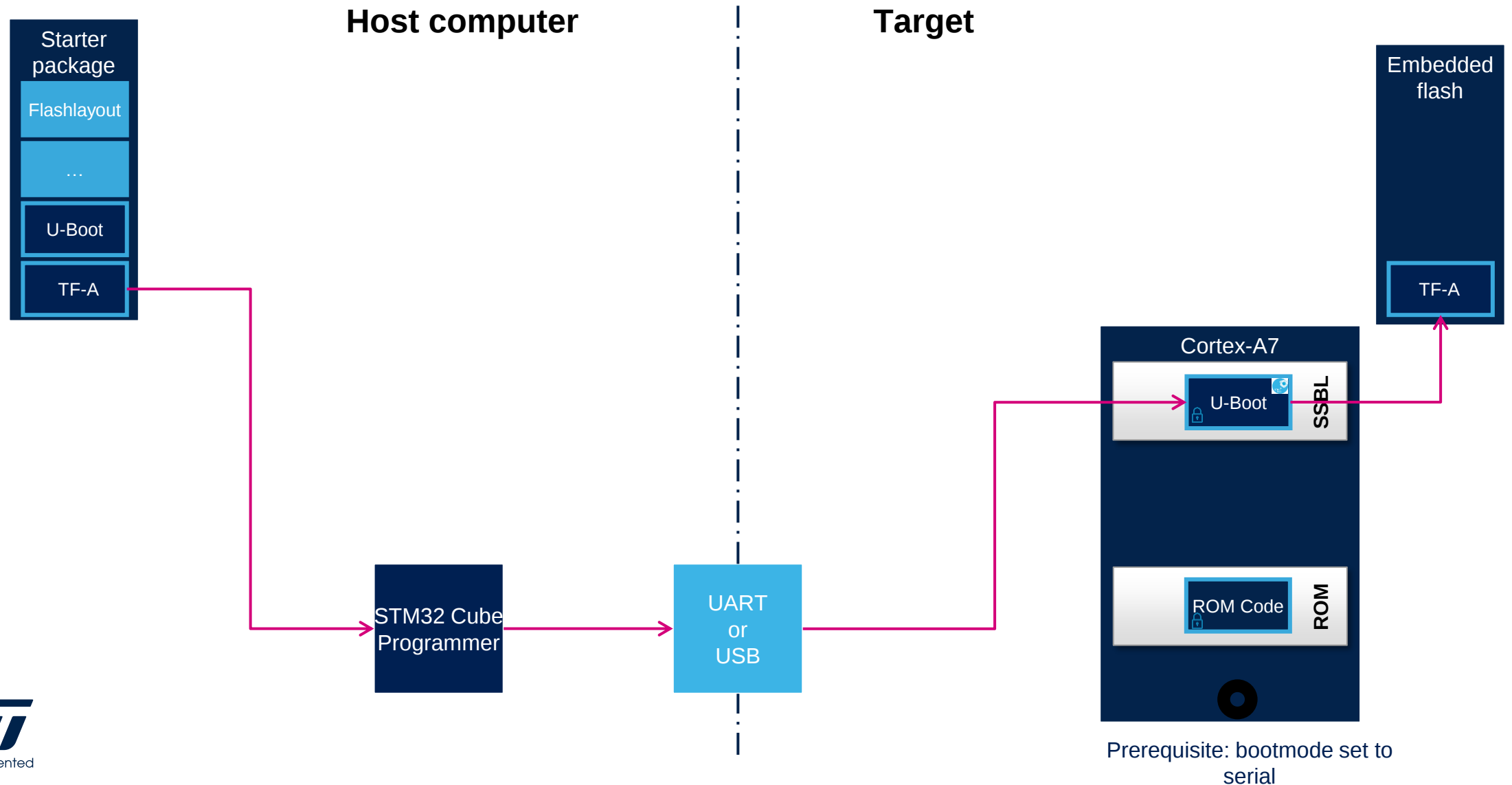
Programmer flow



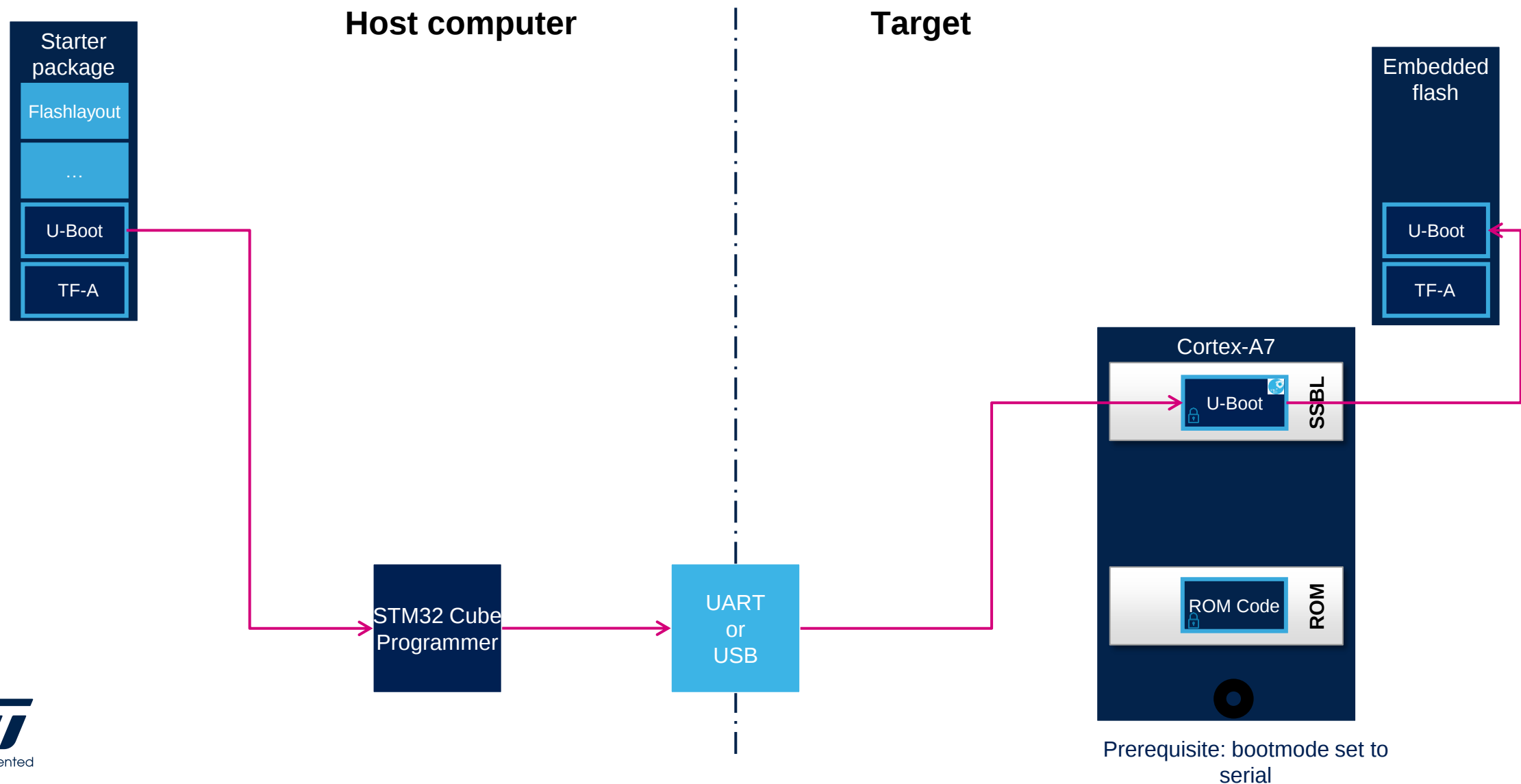
Programmer flow



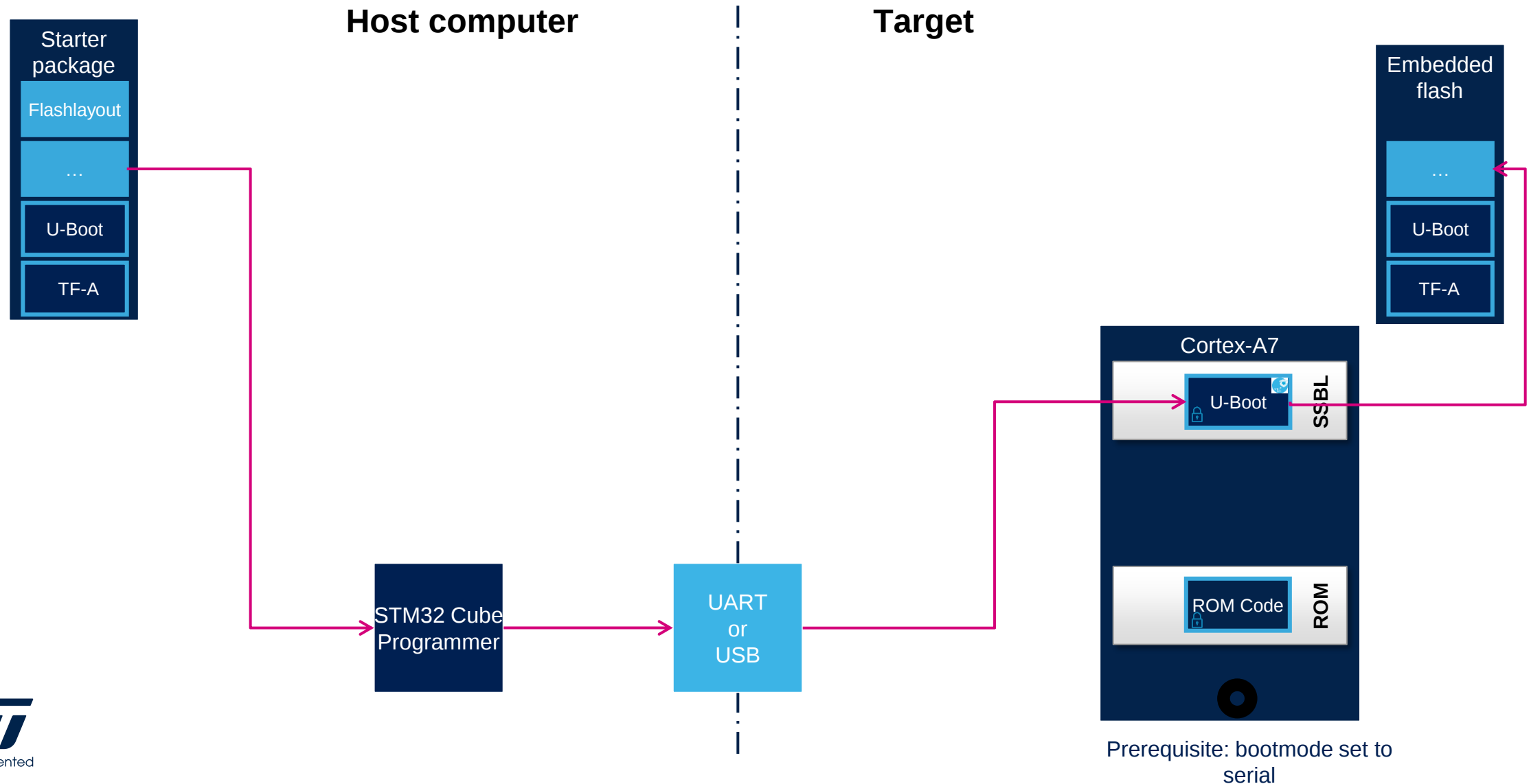
Programmer flow



Programmer flow



Programmer flow



References



TSV file :

https://wiki.st.com/stm32mpu/wiki/STM32CubeProgrammer_flashlayout

Memory Mapping :

https://wiki.st.com/stm32mpu/wiki/STM32MP15_Flash_mapping

DKx hardware description :

[https://wiki.st.com/stm32mpu/wiki/STM32MP157x-DKx - hardware description](https://wiki.st.com/stm32mpu/wiki/STM32MP157x-DKx_-_hardware_description)

DKx starter packages :

[https://wiki.stmicroelectronics.cn/stm32mpu/wiki/STM32MP15_Discovery_kits - Starter Package](https://wiki.stmicroelectronics.cn/stm32mpu/wiki/STM32MP15_Discovery_kits_-_Starter_Package)

Boot chain :

https://wiki.stmicroelectronics.cn/stm32mpu/wiki/Boot_chain_overview

Programmer :

<https://wiki.stmicroelectronics.cn/stm32mpu/wiki/STM32CubeProgrammer>

Thank you