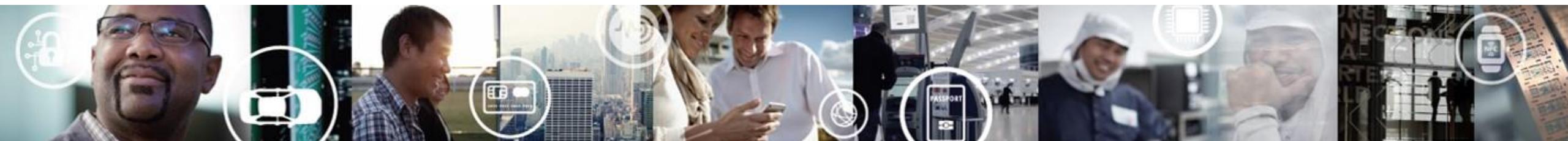


SWUpdate OTA I.MX8MM EVK

BIYONG SUN

REV 3.0
10 AUG 2021



EXTERNAL USE



SECURE CONNECTIONS
FOR A SMARTER WORLD

Embedded System Updating Introduction



Updating an Embedded System

Embedded Systems become more and more complex.

Software for Embedded Systems have new features and fixes can be updated in a reliable way.

Like Android has its own update system. Linux also need an update system.

Linux also has several projects for update system.

Linux popular update systems

Mechanism	Type	Disk layout	Rootfs	Updates from	Updates what	Code stability	OE/Yocto integration	Resource requirements		Failure resilience	Complexity	Downtime	Security
								on server	on client				
swupd	file-based	flexible	read/write	HTTP(S) server, local media	depends on setup	relatively stable, under active development	meta-swupd	moderate, suitable for frequent updates	minimal download, needs sufficient free space in rootfs	favors fast updates over failure resilience	some planning required	minimal, reboot optional	Compatible with Linux IMA, Smack, SELinux. Signed update data, HTTPS transfer protection.
sbabic's swupdate	block-based / file based	flexible	depends on setup (read-only supported)	local and remote (plain HTTP(S) or custom server)	depends on setup	Code relatively stable, 6 months release cycle	meta-swupdate	archives full image per build	download and write full (compressed) image, zero-copy	integrated rollback (requires bootloader support)	easy to use (but requires customization!?)	reboot required	signed and encrypted images, HTTPS
Mender	block-based / file based	flexible (minimum four partitions), U-Boot or GRUB as boot loader	supports read-write and read-only	remote using Mender management server (managed mode) or local using CLI (standalone mode)	Complete rootfs, including kernel (built-in). Customizable with Upgrade Modules	relatively stable, fully supported and tested upgrade path	meta-mender	compressed rootfs per build	download and write compressed rootfs	integrated rollback	easy when using meta-mender	reboot required	HTTPS enforced, signed images
OSTree	file-based	flexible, but supports only limited set of bootloaders	read/write, OS trees bind mounted read-only, /etc and /var writable	local and remote repositories	kernel and filesystem	relatively stable, significant user base, under active development	meta-ostree (WIP), meta-updater (public)	generating commits based on new builds, storing them in repository	updating local repository, hard links for sharing unchanged content between deployments	rollback to a different deployed OS tree	some work required	reboot required	GPG-signed commits
RAUC	block based / file-based (tar)	flexible (block-device/MTD)	depends on setup (read-only supported)	depends on setup	depends on setup (any storage device)	relatively stable, under active development	meta-rauc	archives full (compressed) image per build	download and write full (compressed) image	integrated rollback (requires bootloader support)	some customization required	reboot required	X509-signed update bundles

SWUpdate Introduction



Wide used

SWUpdate is already in many devices in field, providing a reliable way to update your products. Just check a short list of company having integrated SWUpdate in their products – you will be surprised how many devices you can yourself find that are running SWUpdate as OTA updater !



Full Open Source

SWUpdate is a full open-source project – 100% of the code is released under open-source license and SWUpdate is integrated and works with other FOSS(Free/Open-Source Software) projects.

Commercial support is also available. <http://swupdate.org/services>

SUWupdate Main Features

Update everything

- SWUpdate is able to update all components of your device:
- Bootloader
- Linux Kernel
- Root filesystem
- Application
- FPGAs
- FW on separate microcontroller

Hardware compatibility

SWUpdate verifies that delivered software can be applied to the device. A hardware compatibility map is applied to any release.

Rescue System

SWUpdate's small footprint allows to build a rescue system that runs automatically if your device does not boot.

100% Open Source

SWUpdate is 100% Open Source : development goes directly to mainline, and due to large community you profit of the advantages of open-source projects.

Extensible

SWUpdate integrates a Lua interpreter. Project van extend SWUpdate functionalities adding an own custom installer.



SUWupdate Main Features (cont.)

Atomic Update

Atomic update avoids that power-cut and network loss can brick your device. New software is always full installed or update is interrupted letting the device running. There is no partial software update as with package-based update.

Zero copy

SWUpdates has a small footprint – if configured, it does not create temporary copies and the incoming packets are directly installed on the storage. The whole process (decompression, decryption) is done on chunks in memory, making life harder for attacker.

Authenticity of update

Update packages are signed and verified by SWUpdate to authenticate an authorized and trusted update. SWUpdate supports signing with RSA keys and with certificates using an own PKI infrastructure.

Embedded Media

SWUpdates is suitable for all embedded storage – it supports NOR, NAND, eMMC, SPI Flash, and UBI volumes in native mode. There is no constraint about your memory layout and you have full control to define where your software is installed.

Rollback

SWUpdate together with the bootloader detects if the installed software is running correctly and rollbacks to previous installed version in case of failure.

SUWpdate Main Features(cont.)

Multiple interfaces

It does not matter how you plan your updates. SWUPdate supports offline (USB,SD, etc.) as well as remote (OTA) updates. SWUpdate has an integrated Webserver to upgrade devices like SOHO routers, but it connects to backend (Hawkbit) for fleet deployments. The modular design in SWUpdate allows to add further backends or interfaces in future.

SWUpdate provides also a library that can be linked to your application if you have a custom way to get the update package. The library is licensed under a more permissive license (LGPLv2.1) and can be used with close source applications.

Reliable

Atomic update avoids that power-cut and network loss can brick your device

SUWpdate build system support



Buildsystem

SWUpdate is well supported and integrated in modern embedded linux buildsystem



Buildroot package

SWUpdate is integrated as package into buildroot.

meta-swupdate

A meta-swupdate layer provides the best way to integrate SWUpdate in your Yocto based project.



deb package

Official debian package is integrated into debian (experimental). A branch in SWUpdate's source allows to build a package from last sources.

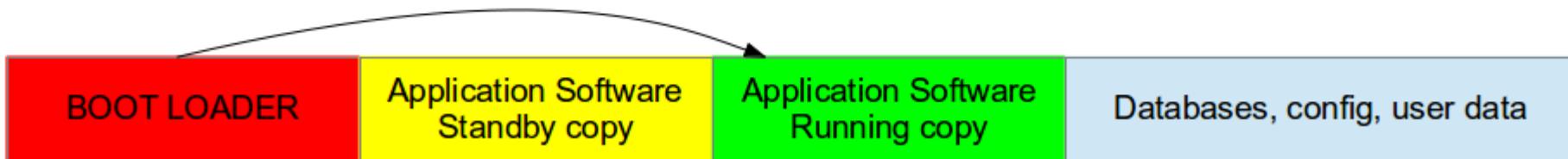
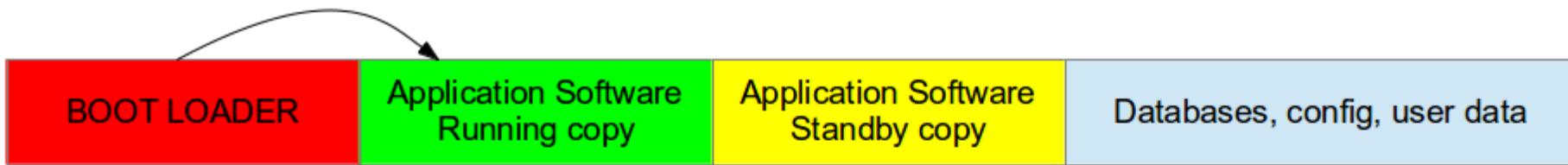


Update Strategy



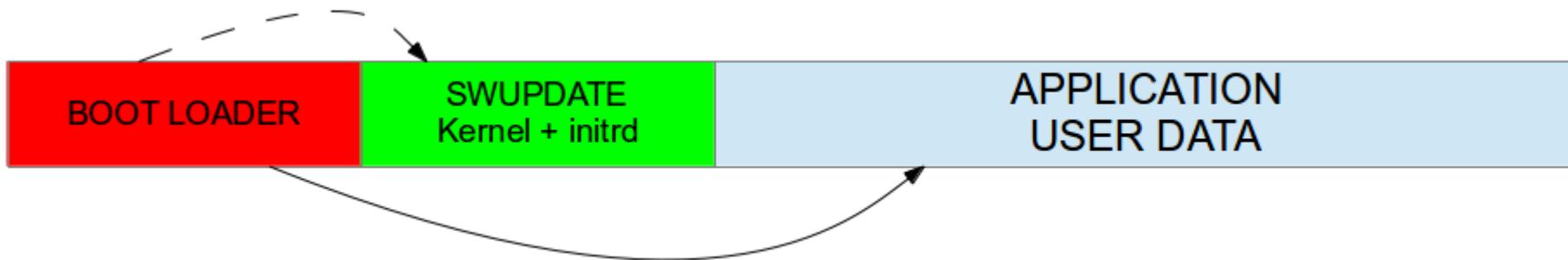
Double copy with fall-back

Each copy must contain the kernel, the root file system, and each further component that can be updated



Single copy - running as standalone image

The software upgrade application consists of kernel (maybe reduced dropping not required drivers) and a small root file system. The whole size is much less than a single copy of the system software.



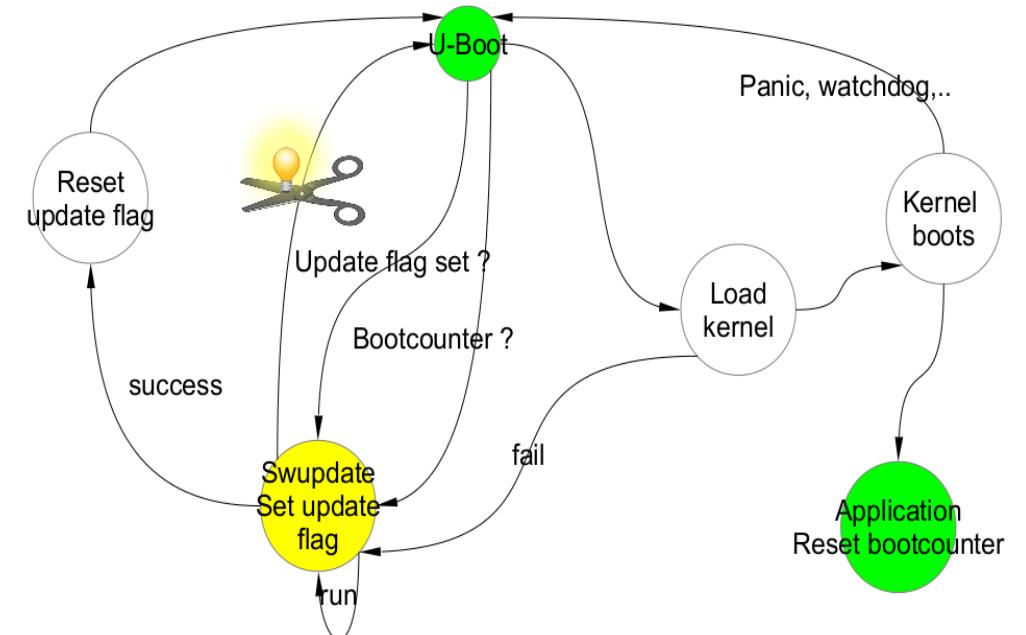
Recovery

Generally, the Recovery behavior can be split according to the chosen scenario:

single copy: SWUpdate is interrupted, and the update transaction did not end with a success. The boot loader is able to start SWUpdate again, having the possibility to update the software again.

double copy: SWUpdate did not switch between standby and current copy. The same version of software, that was not touched by the update, is started again.

Recovery from failures



Upgrading the Boot loader

Updating the boot loader is in most cases a one-way process. On most SOCs, there is no possibility to have multiple copies of the boot loader, and when boot loader is broken, the board does not simply boot. Some SOCs allow one to have multiple copies of the boot loader. But again, there is no general solution for this because it is *very* hardware specific. Most targets do not allow one to update the boot loader. It is very uncommon that the boot loader must be updated when the product is ready for production.

i.MX8MM and i.MX8QXP have secondary boot for two copied of bootloader.

If no necessary, please don't upgrade the bootloader. It is a no return way.

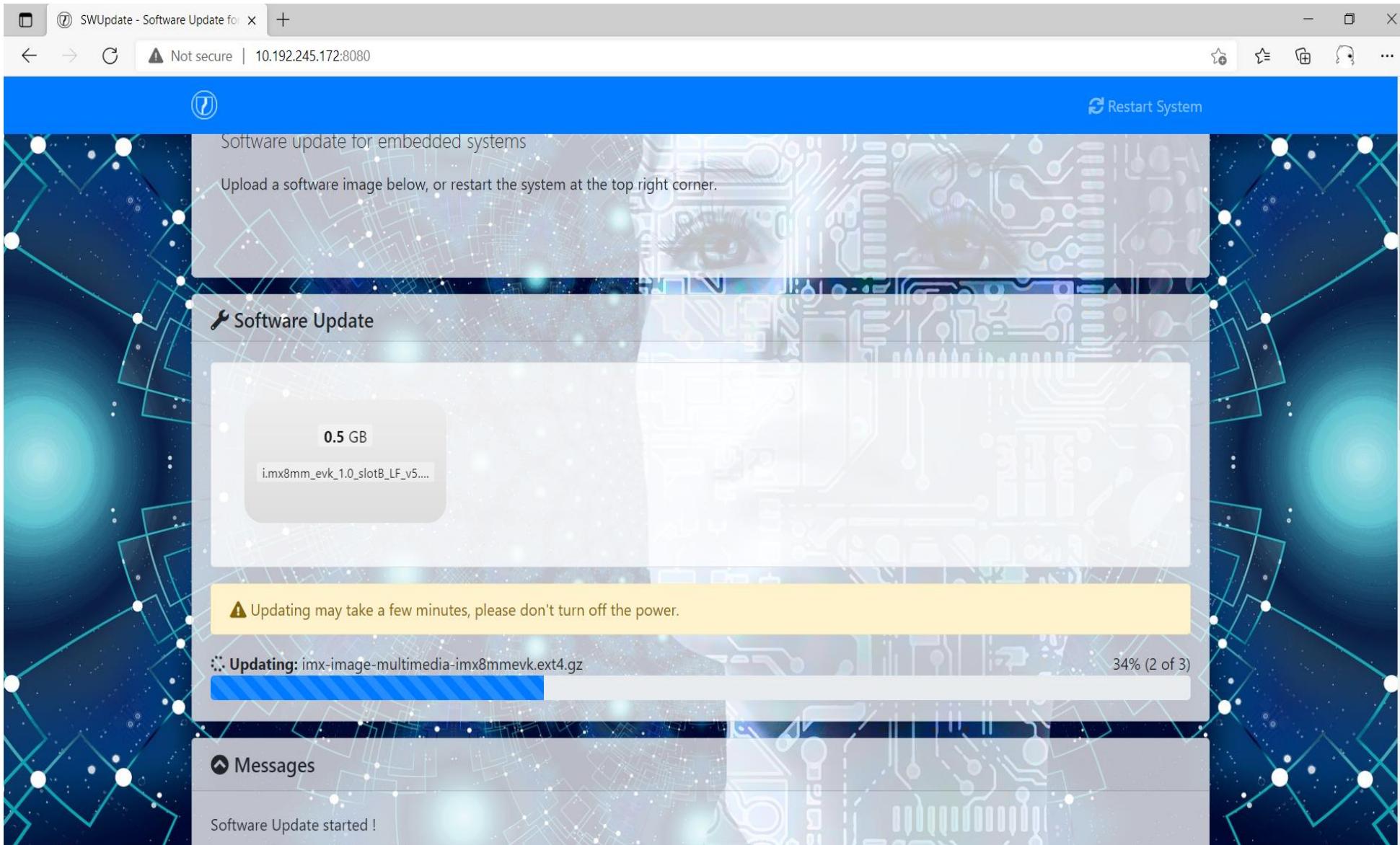
SWUpdate Mode



Mongoose daemon mode

Mongoose is a daemon mode of SWUpdate that provides a web server, web interface and web application.

Mongoose is running on the target board(i.MX8MM EVK/i.MX8QXP MEK).Using Web browser to access it.



Suricatta daemon mode

Suricatta regularly polls a remote server for updates, downloads, and installs them. Thereafter, it reboots the system and reports the update status to the server.

The screenshot is SWUpdate scuricatta working with hawkbit server.

The screenshot shows the hawkBit Update Server interface. On the left, a sidebar menu includes options like Deployment, Rollout, Target Filters, Distributions, Upload, System Config, and Documentation. The main area is titled "Deployment Management". It features two tables: "Targets" and "Distributions". The "Targets" table lists "iMX8QXP_c0_MEK" and "iMX8MM_EVK". The "Distributions" table lists "i.MX8MM_EVK:1.0" and "iMX8QXP_MEK:1.0". A "Status In Sync" message is displayed between the two tables. Below these tables are detailed sections for "Target: iMX8MM_EVK" and "Distribution set: i.MX8MM_EVK:1.0". The "Target" section shows details like Controller Id, Last poll, Address, and Security token. The "Distribution set" section shows Type (App(s) only) and Required Migration Step (No). To the right, an "Action history for iMX8MM_EVK" table is shown, listing a single action entry. The top navigation bar indicates the page is "hawkBit Update Server" at "Not secure | 192.168.35.3:8080/UI/#!deployment".

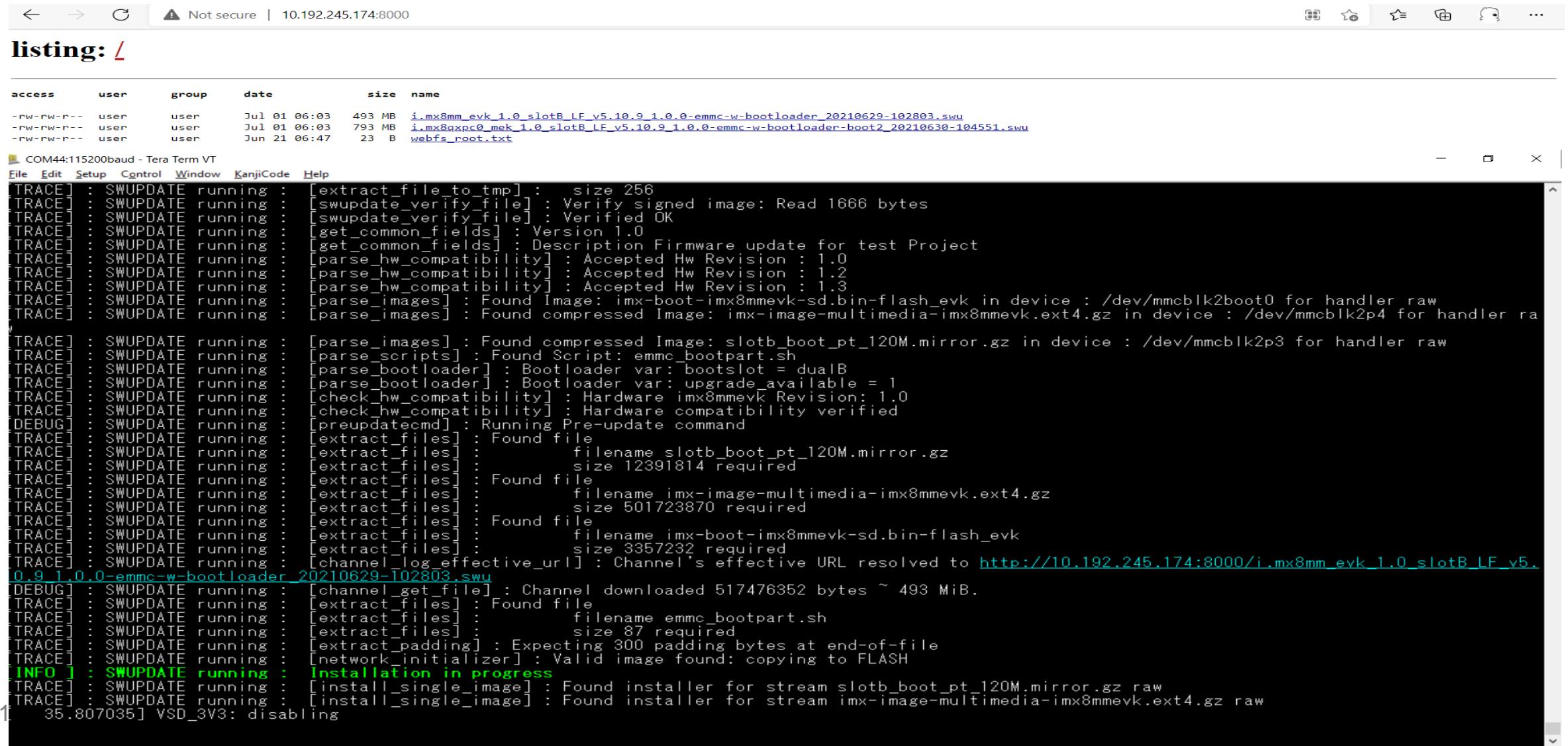
Name	Status	Actions
iMX8QXP_c0_MEK	Green	Details Checkmark Download Delete
iMX8MM_EVK	Green	Details Checkmark Download Delete

Name	Version	Actions
i.MX8MM_EVK	1.0	Details Delete
iMX8QXP_MEK	1.0	Details Delete

Active	Distribution set	Date and time	Status	Type	Actions
<input checked="" type="checkbox"/>	i.MX8MM_EVK:1.0	Jul 1 10:48 SGT 2021	Green	Success	Details Download Cancel Retry Remove

Other mode

SWUpdate could get the update resources from http server, usb disk, sdcard.



The screenshot shows a dual-pane interface. The top pane is a web browser displaying a list of update files from an http://10.192.245.174:8000/ directory. The bottom pane is a terminal window titled 'COM44:115200baud - Tera Term VT' showing the log output of the SWUPDATE command.

listing: /

access	user	group	date	size	name
-rw-rw-r--	user	user	Jul 01 06:03	493 MB	i.mx8mm_evk_1.0_slotB_LF_v5.10.9_1.0.0-emmc-w-bootloader_20210629-102803.swu
-rw-rw-r--	user	user	Jul 01 06:03	793 MB	i.mx8gxpco_mek_1.0_slotB_LF_v5.10.9_1.0.0-emmc-w-bootloader-boot2_20210630-104551.swu
-rwxrwxr--	user	user	Jun 21 06:47	23 B	webfs_root.txt

```
File Edit Setup Control Window KanjiCode Help
[TRACE]: : SWUPDATE running : [extract_file_to_tmp] : size 256
[TRACE]: : SWUPDATE running : [swupdate_verify_file] : Verify signed image: Read 1666 bytes
[TRACE]: : SWUPDATE running : [swupdate_verify_file] : Verified OK
[TRACE]: : SWUPDATE running : [get_common_fields] : Version 1.0
[TRACE]: : SWUPDATE running : [get_common_fields] : Description Firmware update for test Project
[TRACE]: : SWUPDATE running : [parse_hw_compatibility] : Accepted Hw Revision : 1.0
[TRACE]: : SWUPDATE running : [parse_hw_compatibility] : Accepted Hw Revision : 1.2
[TRACE]: : SWUPDATE running : [parse_hw_compatibility] : Accepted Hw Revision : 1.3
[TRACE]: : SWUPDATE running : [parse_images] : Found Image: imx-boot-imx8mmevk-sd.bin-flash_evk in device : /dev/mmcblk2boot0 for handler raw
[TRACE]: : SWUPDATE running : [parse_images] : Found compressed Image: imx-image-multimedia-imx8mmevk.ext4.gz in device : /dev/mmcblk2p4 for handler ra
[TRACE]: : SWUPDATE running : [parse_images] : Found compressed Image: slotb_boot_pt_120M.mirror.gz in device : /dev/mmcblk2p3 for handler raw
[TRACE]: : SWUPDATE running : [parse_scripts] : Found Script: emmc_bootpart.sh
[TRACE]: : SWUPDATE running : [parse_bootloader] : Bootloader var: bootslot = dualB
[TRACE]: : SWUPDATE running : [parse_bootloader] : Bootloader var: upgrade_available = 1
[TRACE]: : SWUPDATE running : [check_hw_compatibility] : Hardware imx8mmevk Revision: 1.0
[TRACE]: : SWUPDATE running : [check_hw_compatibility] : Hardware compatibility verified
[DEBUG]: : SWUPDATE running : [preupdatecmd] : Running Pre-update command
[TRACE]: : SWUPDATE running : [extract_files] : Found file
[TRACE]: : SWUPDATE running : [extract_files] : filename slotb_boot_pt_120M.mirror.gz
[TRACE]: : SWUPDATE running : [extract_files] : size 12391814 required
[TRACE]: : SWUPDATE running : [extract_files] : Found file
[TRACE]: : SWUPDATE running : [extract_files] : filename imx-image-multimedia-imx8mmevk.ext4.gz
[TRACE]: : SWUPDATE running : [extract_files] : size 501723870 required
[TRACE]: : SWUPDATE running : [extract_files] : Found file
[TRACE]: : SWUPDATE running : [extract_files] : filename imx-boot-imx8mmevk-sd.bin-flash_evk
[TRACE]: : SWUPDATE running : [extract_files] : size 3357232 required
[TRACE]: : SWUPDATE running : [channel_log_effective_url] : Channel's effective URL resolved to http://10.192.245.174:8000/i.mx8mm\_evk\_1.0\_slotB\_LF\_v5.10.9\_1.0.0-emmc-w-bootloader\_20210629-102803.swu
[DEBUG]: : SWUPDATE running : [channel_get_file] : Channel downloaded 517476352 bytes ~ 493 MiB.
[TRACE]: : SWUPDATE running : [extract_files] : Found file
[TRACE]: : SWUPDATE running : [extract_files] : filename emmc_bootpart.sh
[TRACE]: : SWUPDATE running : [extract_files] : size 87 required
[TRACE]: : SWUPDATE running : [extract_padding] : Expecting 300 padding bytes at end-of-file
[TRACE]: : SWUPDATE running : [network_initializer] : Valid image found: copying to FLASH
[INFO]: : SWUPDATE running : Installation in progress
[TRACE]: : SWUPDATE running : [install_single_image] : Found installer for stream slotb_boot_pt_120M.mirror.gz raw
[TRACE]: : SWUPDATE running : [install_single_image] : Found installer for stream imx-image-multimedia-imx8mmevk.ext4.gz raw
1 35.807035] VSD_3V3: disabling
```

SWUpdate Demo



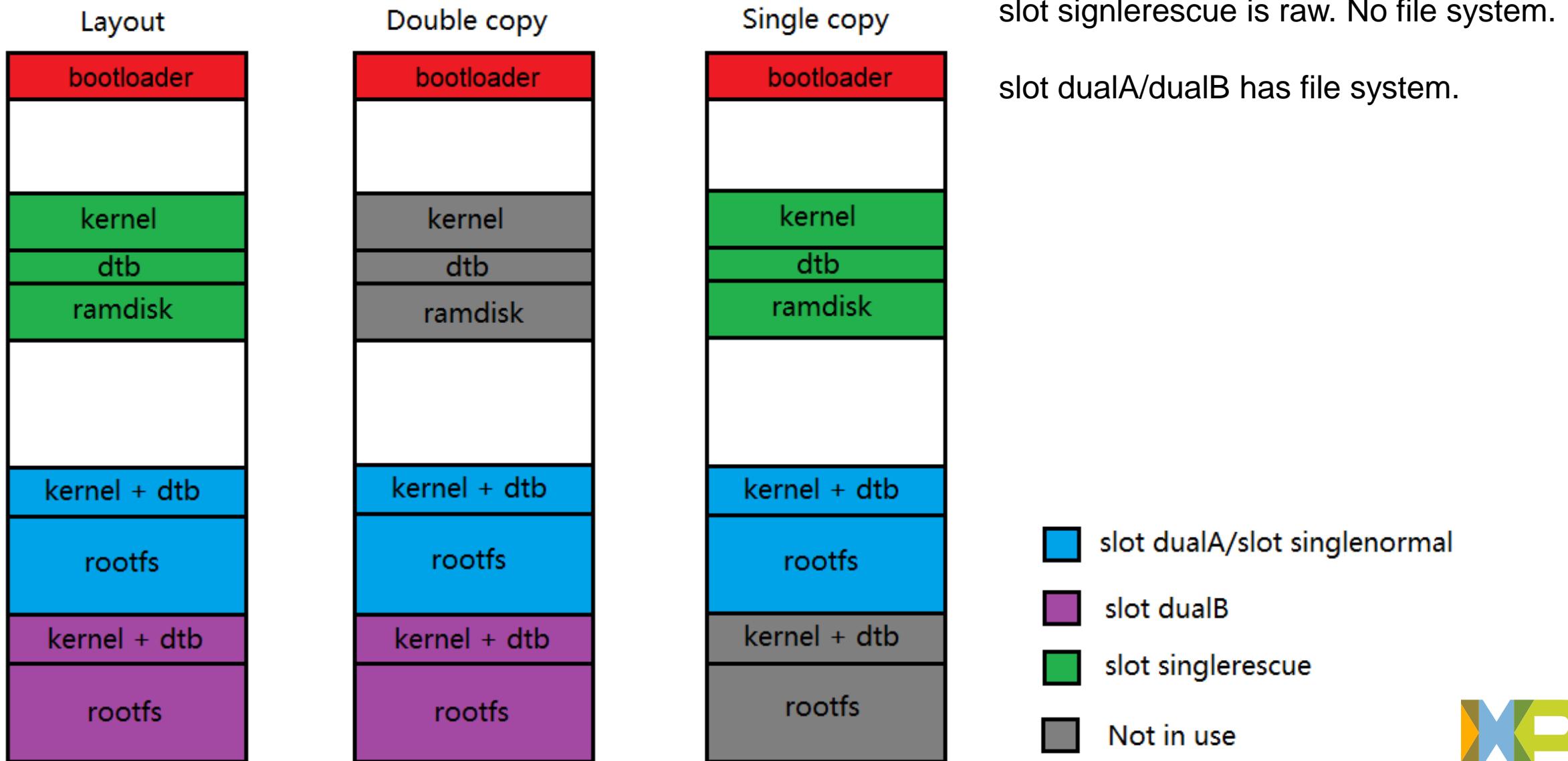
SWUpdate Demo Environment

HW: i.MX8MM EVK

SW: L5.4.70-2.3.0

LF_v5.10.9_1.0.0

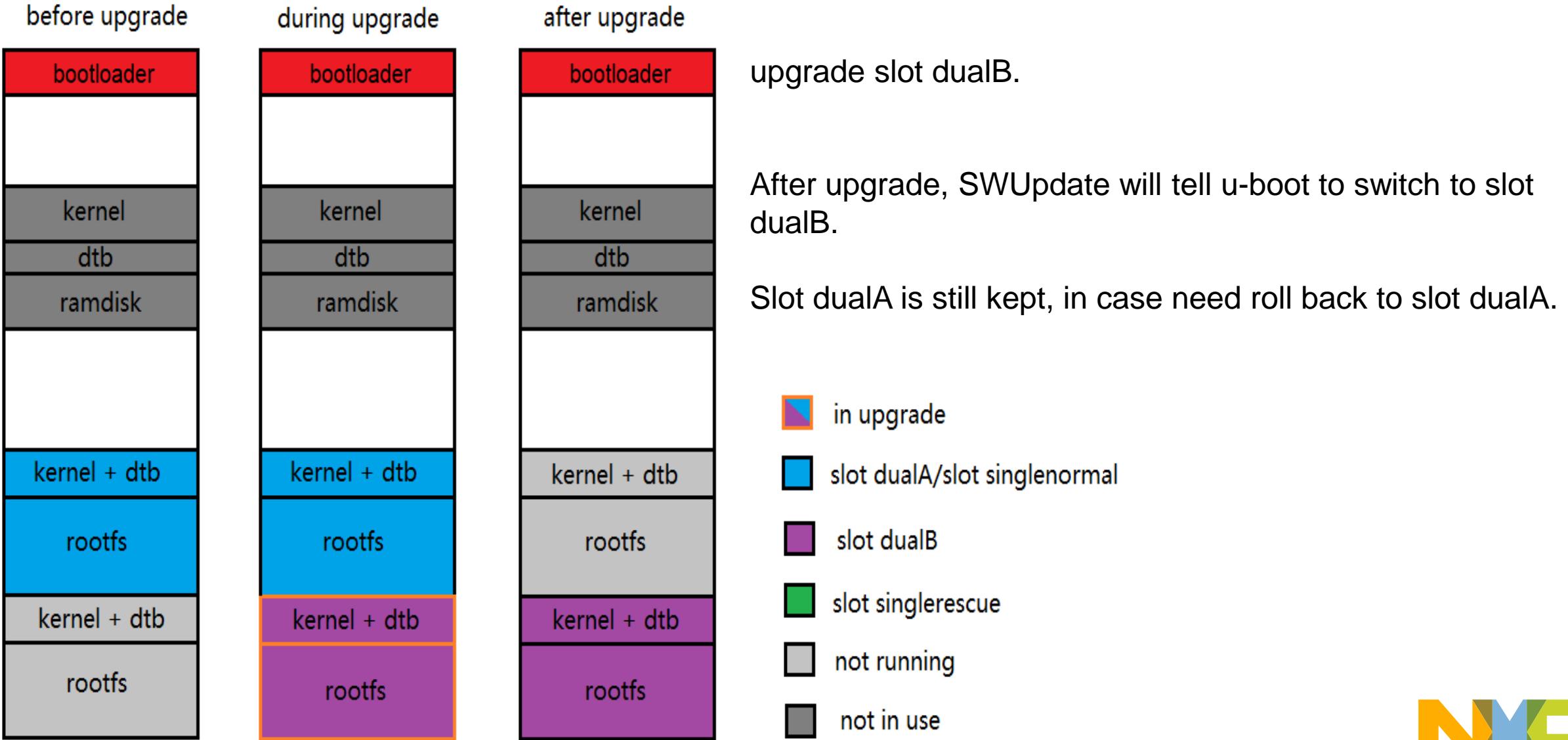
SWUpdate Demo Image Layout



Mongoose Daemon Mode Double Copy Upgrade



i.MX8MM EVK Mongoose daemon mode Double copy Demo

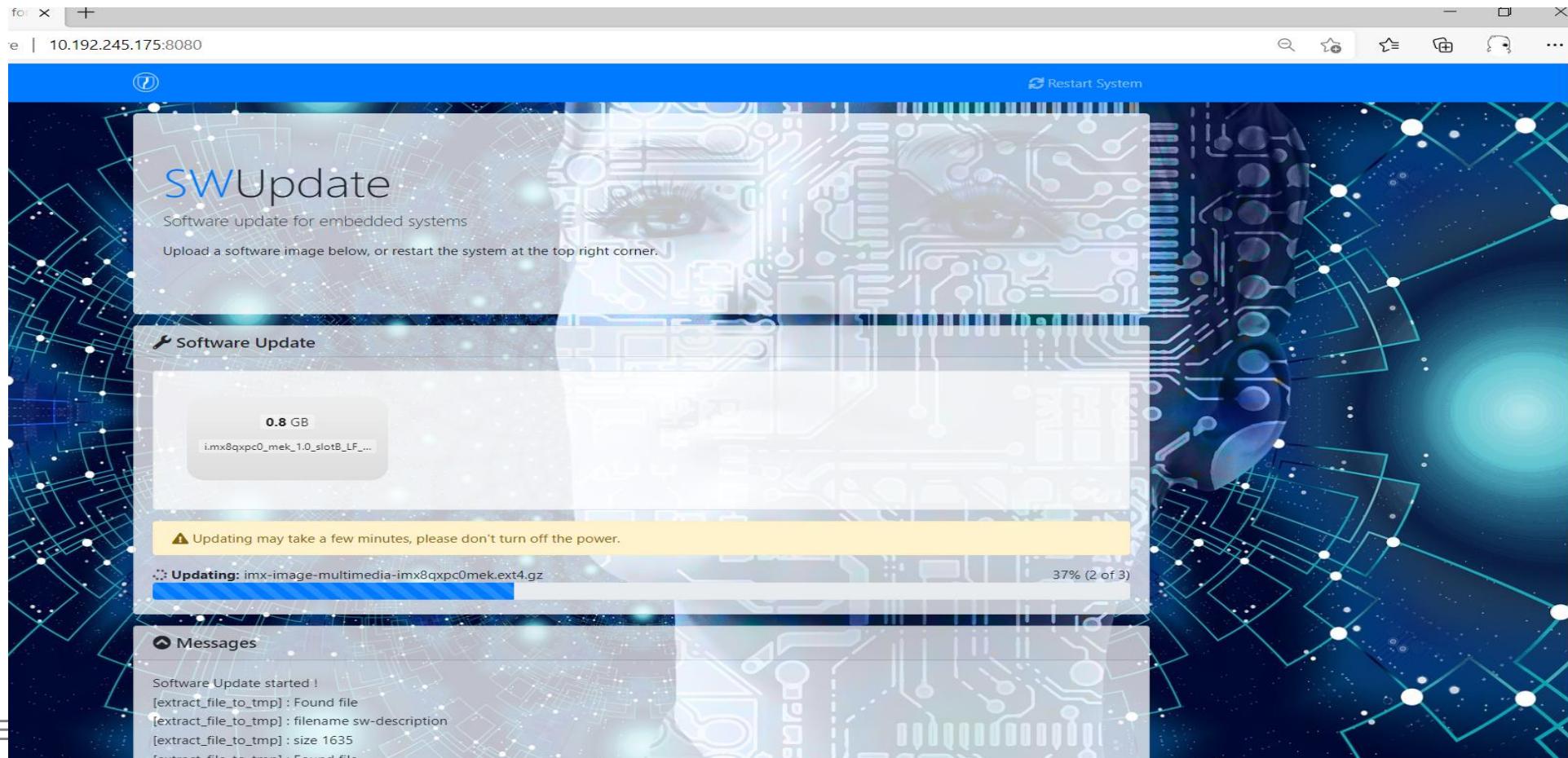


i.MX8MM EVK Mongoose daemon mode Double copy Demo

Upgrade with

i.mx8mm_evk_1.0_slotB_LF_v5.10.9_1.0.0-doublecopy-emmc-full_20210713-104718.swu

Upgrade the bootloader, kernel, dtb, rootfs.



i.MX8MM EVK Mongoose daemon mode Double copy Demo(cont.)

Before upgrade

<http://10.192.245.175:8080/sysinfo.cgi>

```
Linux imx8mmevk 5.4.70-2.3.0+g4f2631b022d8 #1 SMP PREEMPT Sun Jun 27 18:06:47 UTC 2021 aarch64 aarch64 aarch64 GNU/Linux  
console=ttyMxc1,115200 root=/dev/mmcblk2p2 rootwait rw cur_slot=dualA U-Boot_ver=2020.04-5.4.70-2.3.0+ge42dee801e(Jun 27 2021-18:23:27)
```

After upgrade

<http://10.192.245.175:8080/sysinfo.cgi>

```
Linux imx8mmevk 5.10.9-1.0.0+g32513c25d8c7 #1 SMP PREEMPT Tue Mar 9 02:17:18 UTC 2021 aarch64 aarch64 aarch64 GNU/Linux  
console=ttyMxc1,115200 root=/dev/mmcblk2p4 rootwait rw cur_slot=dualB U-Boot_ver=2020.04-5.10.9-1.0.0+gad7b74b415(Mar 05 2021-07:05:56)
```

Double Copy Recovery(Roll Back)



i.MX8MM EVK Double Copy Recovery(Roll Back)

Simulate a boot fail.

In u-boot:

```
setenv bootslot dualB  
setenv upgrade_available 1
```

```
saveenv
```

```
echo ${bootcount} ${bootlimit}  
      1           3
```

Every time, when u-boot boot, press the reset button on board to simulate a boot fail.

After 3(bootlimit) times, u-boot will think a boot fail and go to altbootcmd. altbootcmd will do rolling back.

```
Net: eth0: ethernet@5b040000 [PRIME]  
Warning: ethernet@5b050000 (eth1) using random MAC  
address - ba:0f:ca:f9:fa:d9  
, eth1: ethernet@5b050000  
Fastboot: Normal  
Saving Environment to MMC... Writing to redundant  
MMC(0)... OK  
Normal Boot
```

Warning: Bootlimit (3) exceeded. Using altbootcmd.

Hit any key to stop autoboot: 3 2 1 0

Rolling back to slot dualA

```
Saving Environment to MMC... Writing to MMC(2)... OK
```

```
RootFs Slot A
```

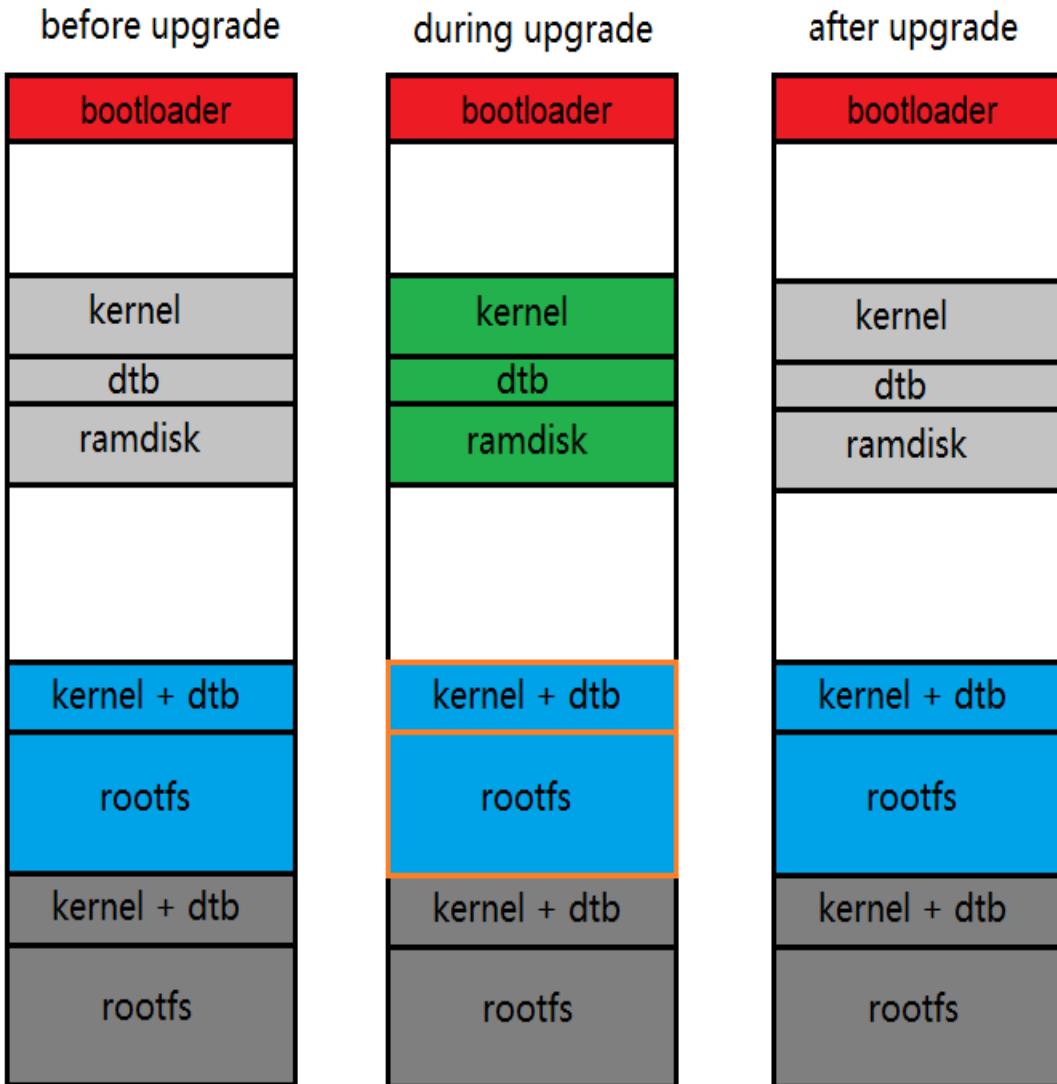
```
switch to partitions #0, OK
```

```
mmc2(part 0) is current device
```

Mongoose Daemon Mode Single Copy



i.MX8MM EVK Mongoose daemon mode Single Copy Demo



Single copy upgrade starts from switch to slot singlerescue from slot singlenormal (dualA) and getting into upgrade mode.

After upgrade, SWUpdate will tell u-boot to switch back to slot singlenormal(dualA).



i.MX8MM EVK Single Copy Demo(cont.)

Single copy upgrade starts from getting into upgrade mode.

After upgrade done, boot to the normal run.

In Linux:

```
fw_setenv bootslot singlerescue  
fw_setenv upgrade_available 1  
reboot
```

You will see u-boot switch to ram disk boot.
Once boot done, the SWUpdate Mongoose is running.

Hit any key to stop autoboot: 3 2 1 0
swuboot ramdisk

MMC read: dev # 0, block # 16384, count 61440 ... 61440
blocks read: OK

MMC read: dev # 0, block # 77824, count 512 ... 512
blocks read: OK

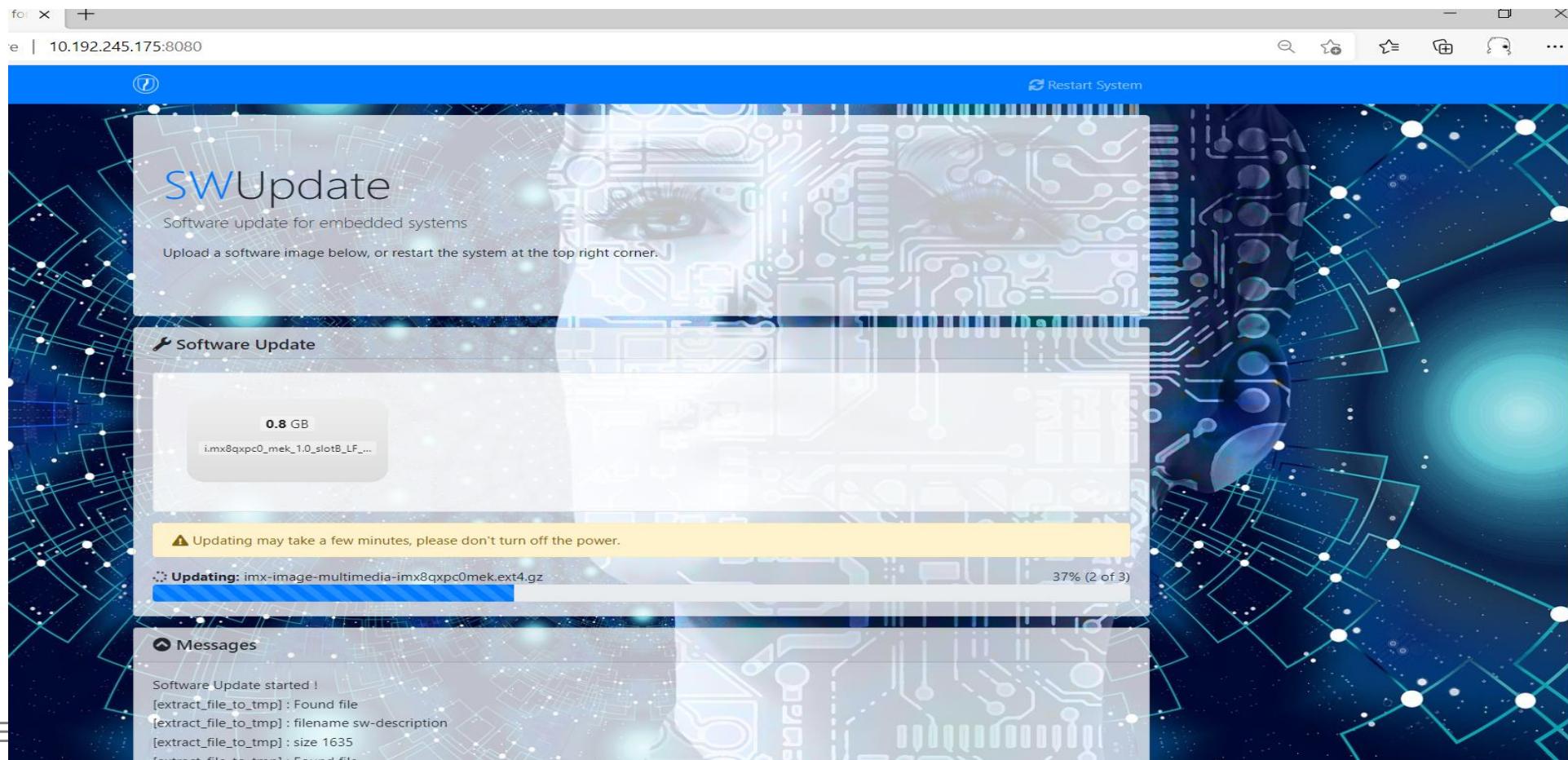
MMC read: dev # 0, block # 86016, count 61440 ... 61440
blocks read: OK
Loading init Ramdisk from Legacy Image at d2100000 ...

i.MX8MM EVK Mongoose daemon mode Single copy Demo

Upgrade with

i.mx8mm_evk_1.0_slotS_LF_v5.10.9_1.0.0-singlecopy-emmc-full_20210713-104833.swu

Upgrade the bootloader, kernel, dtb, rootfs.



i.MX8MM EVK Mongoose daemon mode Single copy Demo(cont.)

Before upgrade

<http://10.192.245.175:8080/sysinfo.cgi>

```
Linux imx8mmevk 5.4.70-2.3.0+g4f2631b022d8 #1 SMP PREEMPT Sun Jun 27 18:06:47 UTC 2021 aarch64 aarch64 aarch64 GNU/Linux
console=ttyMxc1,115200 root=/dev/mmcblk2p2 rootwait rw cur_slot=singlenormal U-Boot_ver=2020.04-5.4.70-2.3.0+ge42dee801e(Jun 27 2021-18:23:27)
```

During upgrade

<http://10.192.245.175:8080/sysinfo.cgi>

```
Linux imx8mmevk 5.4.70-2.3.0+g4f2631b022d8 #1 SMP PREEMPT Sun Jun 27 18:06:47 UTC 2021 aarch64 aarch64 aarch64 GNU/Linux
console=ttyMxc1,115200 root=/dev/mmcblk2p2 rootwait rw cur_slot=singlerescue U-Boot_ver=2020.04-5.4.70-2.3.0+ge42dee801e(Jun 27 2021-18:23:27)
```

After upgrade

<http://10.192.245.175:8080/sysinfo.cgi>

```
Linux imx8mmevk 5.10.9-1.0.0+g32513c25d8c7 #1 SMP PREEMPT Tue Mar 9 02:17:18 UTC 2021 aarch64 aarch64 aarch64 GNU/Linux
console=ttyMxc1,115200 root=/dev/mmcblk2p2 rootwait rw cur_slot=singlenormal U-Boot_ver=2020.04-5.10.9-1.0.0+gad7b74b415(Mar 05 2021-07:05:56)
```

Single Copy Recover



i.MX8MM EVK Mongoose daemon mode Single Copy Demo

Single copy



Single copy upgrade, the rescue part is always kept.

In this demo, designed even when the slot singlerescue boot fail.
The board can get into usb serial download mode.
We can have last chance to flash the board.

i.MX8MM EVK Mongoose daemon mode Single Copy Demo

Simulate a boot fail.

```
In u-boot:  
setenv bootslot singlerescue  
setenv upgrade_available 1  
saveenv  
  
echo ${bootcount} ${bootlimit}  
      1           3
```

Every time, when u-boot boot, press the reset button on board to simulate a boot fail.

After 3(bootlimit) times, u-boot will think a boot fail and go to altbootcmd. altbootcmd will run fastboot. Then usb download tool uuu can help to flash the board.

```
Net: eth0: ethernet@5b040000 [PRIME]  
Warning: ethernet@5b050000 (eth1) using random MAC  
address - a6:e5:09:bb:85:11  
, eth1: ethernet@5b050000  
Fastboot: Normal  
Saving Environment to MMC... Writing to MMC(2)... OK  
Normal Boot  
Warning: Bootlimit (3) exceeded. Using altbootcmd.  
Hit any key to stop autoboot: 0  
Boot Fail! Get into usb fastboot download.
```

Hardware compatibility



i.MX8MM EVK hardware compatibility Demo

The screenshot shows a software update interface with the following details:

- Software Update** section:
 - File size: 3.4 MB
 - File name: i.mx8mm_evk_1.0_slotNONE_LF_v5.10.9_1.0.0-emmc-demo-hw-check_20210707-075230.swu
- Update failed.** message
- Messages** log:
 - [swupdate_verify_file] : Verify signed image: Read 503 bytes
 - [swupdate_verify_file] : Verified OK
 - [get_common_fields] : Version 1.0
 - [get_common_fields] : Description Firmware update for test Project
 - [parse_hw_compatibility] : Accepted Hw Revision : 1.2
 - [parse_hw_compatibility] : Accepted Hw Revision : 1.3
 - [parse_images] : Found Image: imx-boot-imx8mmevk-sd.bin-flash_evk in device : /dev/mmcblk2boot0 for handler raw
 - [check_hw_compatibility] : Hardware imx8mmevk Revision: 1.0
 - ERROR : SW not compatible with hardware
 - Image invalid or corrupted. Not installing ...
 - [network_initializer1] : Main thread sleep again !

On the right side, there is a terminal window showing the following command and output:

```
root@imx8mmevk:~# cat /etc/hwrevision  
imx8mmevk 1.0
```

The terminal output is annotated with red boxes and underlines:

- hwdescription** (highlighted)
- hardware-compatibility: ["1.2", "1.3"];** (highlighted)
- imx8mmevk: {** (highlighted)
- board root@imx8mmevk:~# cat /etc/hwrevision** (highlighted)
- imx8mmevk 1.0** (highlighted)

Installed software version check



Installed software version check demo

SWUpdate searches for a file (/etc/sw-versions is the default location) containing all versions of the installed images. This must be generated before running SWUpdate. The file must contain pairs with the name of image and its version.

Because it is separated as components, not include /etc/sw-versions in the build. Need to manually create /etc/sw-versions for this demo.

The demo will show if the software version is same, will skip upgrade this component.

```
sw-description:  
images: (  
    {  
        name = "bootloader";  
        version ="5.10.9";  
        install-if-different = true;  
        filename = "imx-boot-imx8mmevk-sd.bin-flash_evk";  
        sha256 = "07fb81ac2ab0d7db375a5efe3e6057d1318acac33ccd4a64fe21abd9a4526ffa";  
        device = "/dev/mmcblk2boot0";  
        offset = "33K";  
    }  
);  
files: (  
    {  
        filename = "test.txt.gz";  
        sha256 = "b0dc9cd0435966fb4682a96d41ab8df331b15a6827d91bd4f52a8fca4a1cd4b";  
        compressed = "zlib";  
        path = "/test.txt";  
        device = "/dev/mmcblk2p2";  
        filesystem = "ext4"  
    }  
);  
EXTERNAL USE  
);
```

```
root@imx8mmevk:~# cat /etc/sw-versions  
bootloader 5.10.9
```



Installed software version check demo(cont.)

The same version of bootloader is skipped.
But the test.txt.gz is still installed.

```
[parse_hw_compatibility] : Accepted Hw Revision : 1.0
[parse_hw_compatibility] : Accepted Hw Revision : 1.2
[parse_hw_compatibility] : Accepted Hw Revision : 1.3
[parse_files] : Found compressed File: test.txt.gz --> /test.txt (/dev/mmcblk2p2)
[compare_versions] : Parsed: '1407417833816064' <-> '1407417833816064'
[is_image_installed] : bootloader(5.10.9) already installed, skipping...
[parse_images] : Found Image bootloader 5.10.9: imx-boot-imx8mmevk-sd.bin-flash_evk in device : /dev/mmcblk2boot0 for handler raw SKIPPED
[send_progress_msg] : A progress client disappeared, removing it.
[send_progress_msg] : A progress client disappeared, removing it.
[check_hw_compatibility] : Hardware imx8mmevk Revision: 1.0
```



Suricatta Daemon Mode With Hawkbit Server



Hawkbit Server

<https://www.eclipse.org/hawkbit/>

Target board registered

hawkBit Update Server +

Not secure | 192.168.35.3:8080/UI/#!deployment

DEFAULT

Deployment Management

Targets

Name	Status	Actions
iMX8MM_EVK	Online	
iMX8QXP_c0_MEK	Online	

Distributions

Name	Version	Actions

Action history for iMX8MM_EVK

Active	Distribution set	Date and time	Status	Type	Actions

Deployment

Rollout

Target Filters

Distributions

Upload

System Config

Documentation

Filters

Simple Filter

NO TAG

Filter by Status

Filter by Overdue

Custom Filter

Target: iMX8MM_EVK

Details Description Attributes

Controller Id: iMX8MM_EVK

Last poll: Thu Jul 1 10:37:58 SGT 2021

Address: http://10.192.245.172

Security token: 6c5703dc072f17a9a847

Total Targets: 2

hawkBit

Create Distribution

EF hawkBit Update Server +

Not secure | 192.168.35.3:8080/UI/#!spUpload

DEFAULT

Deployment Rollout Target Filters Distributions Upload System Config Documentation

Upload Management

Filter by type

Software Module		
	Name	Version
	i.MX8MM_EVK	1.0
	iMX8QXP_MEK	1.0

Artifact Details of i.MX8MM_EVK:1.0

File name	Size(B)	Last modified da...
i.mx8mm_evk_1.0_suricatta_hawkbit_test.swu	2048	Thu Jul 1 10:41:1...

Software Module: i.MX8MM_EVK:1.0

Details Description Logs Metadata

Vendor:

Type: Application

Assignment type: Software (SW)

Drop Files to upload

Upload File

The screenshot shows the hawkBit Update Server interface. On the left, a sidebar contains navigation links: Deployment, Rollout, Target Filters, Distributions, Upload (which is currently selected), and System Config. Below these are Documentation and a question mark icon. The main area is titled 'Upload Management'. It features a table for 'Software Module' with columns for Name, Version, and Delete. Two entries are listed: 'i.MX8MM_EVK' (version 1.0) and 'iMX8QXP_MEK' (version 1.0). To the right, a panel shows 'Artifact Details' for the selected module, including the file name, size (2048 B), and last modified date (Thu Jul 1 10:41:1...). At the bottom, there's a section for 'Software Module: i.MX8MM_EVK:1.0' with tabs for Details, Description, Logs, and Metadata. Below this, it shows Vendor: (empty), Type: Application, and Assignment type: Software (SW). A large dashed box at the bottom right is labeled 'Drop Files to upload' with an arrow pointing down, and a 'Upload File' button is located below it.

Create Distribution(cont.)

hawkBit Update Server Not secure | 192.168.35.3:8080/UI/#!distributions

DEFAULT

Deployment Rollout Target Filters Distributions Upload System Config Documentation

Distributions Management

Filter by type

- App(s) only
- OS only
- OS with app(s)

Name	Version	Delete
i.MX8MM_EVK	1.0	trash
iMX8QXP_MEK	1.0	trash

Distribution set: iMX8QXP_MEK:1.0

Type: App(s) only

Required Migration Step: No

Software Module

Filter by type

- Application
- os

Name	Version	Delete
iMX8QXP_MEK	1.0	trash
i.MX8MM_EVK	1.0	trash

Software Module

Details Description Logs Metadata

hawkBit

Create Distribution(cont.)

hawkBit Update Server Not secure | 192.168.35.3:8080/UI/#!distributions

DEFAULT

Deployment Rollout Target Filters Distributions Upload System Config Documentation

Distributions Management

Filter by type

- App(s) only
- OS only
- OS with app(s)

Name	Version	Delete
i.MX8MM_EVK	1.0	trash
iMX8QXP_MEK	1.0	trash

Distribution set: iMX8QXP_MEK:1.0

Type: App(s) only

Required Migration Step: No

Software Module

Filter by type

- Application
- os

Name	Version	Delete
iMX8QXP_MEK	1.0	trash
i.MX8MM_EVK	1.0	trash

Software Module

Details Description Logs Metadata

hawkBit

Target Filter

EF hawkBit Update Server

Not secure | 192.168.35.3:8080/UI/#!targetFilters

DEFAULT

Deployment

Rollout

Target Filters

Distributions

Upload

System Config

Documentation

Total Filtered Targets: 0



hawkBit

Target Filter Management

Custom Filters > Create Filter

Name *

Query * controllerid==iMX8QXP_c0_MEK

Controller ID	Name	Description	Status	Created By	Created Date	Modified By	Modified Date	Actions
---------------	------	-------------	--------	------------	--------------	-------------	---------------	---------

Target Filter(cont.)

EF hawkBit Update Server

Not secure | 192.168.35.3:8080/UI/#!targetFilters

DEFAULT

Deployment

Rollout

Target Filters

Distributions

Upload

System Config

Documentation

Total Filtered Targets: 0

hawkBit

Target Filter Management

Custom Filters > Create Filter

Name * i.MX8MM_EVK

Query * controllerid==iMX8MM_EVK

Controller ID	Name	Description	Status	Created By	Created Date	Modified By	Modified Date
---------------	------	-------------	--------	------------	--------------	-------------	---------------

Target Filter(cont.)

EF hawkBit Update Server

Not secure | 192.168.35.3:8080/UI/#!targetFilters

DEFAULT

Deployment

Rollout

Target Filters

Distributions

Upload

System Config

Documentation

hawkBit

Target Filter Management

Custom Filter

Name	Created By	Created Date	Modified By	Modified Date	Auto assignment	Delete
IMX8MM_EVK	admin	Thu Jul 1 10:46:16 SGT 2021	admin	Thu Jul 1 10:46:16 SGT 2021	none	Delete
IMX8QXP_MEK	admin	Thu Jul 1 10:45:46 SGT 2021	admin	Thu Jul 1 10:45:46 SGT 2021	none	Delete

Deployment

hawkBit Update Server +

Not secure | 192.168.35.3:8080/UI/#!deployment

Settings and more (Alt+F)

hawkBit

Deployment Management

Deployment Targets Distributions Action history for iMX8MM_EVK

Simple Filter

NO TAG

Name	Status	Actions
iMX8MM_EVK	OK	↻ ⚡ 🗑
IMX8QXP_r0_MEK	OK	↻ ⚡ 🗑

Name	Version	Actions
i.MX8MM_EVK	1.0	☒ 🗑
IMX8QXP_MEK	1.0	☒ 🗑

Confirm Assignment

Are you sure you want to assign Distribution set i.MX8MM_EVK:1.0 to Target iMX8MM_EVK?

⚡ Forced ⚡ Soft Download Only ⚡ Time Forced

Use maintenance window ?

7/15/21 10:47 AM

OK Cancel

Target: iMX8MM_EVK

Details Description Attributes

Controller Id: iMX8MM_EVK
Last poll: Thu Jul 1 10:43:06 SGT 2021
Address: http://10.192.245.172
Security token: 6c5703dc072f17a9a847

Total Targets: 2

Distribution set: i.MX8MM_EVK:1.0

Details Description Modules

Type: App(s) only
Required Migration Step: No

Deployment(cont.)

hawkBit Update Server Not secure | 192.168.35.3:8080/UI/#!deployment

DEFAULT

Deployment Management

Targets

Name	Status	Actions
iMX8MM_EVK	● ○ 🔍	🔗 🗑
iMX8QXP_c0_MEK	● ○ 🔍	🔗 🗑

Distributions

Name	Version	Actions
i.MX8MM_EVK	1.0	🔗 🗑
iMX8QXP_MEK	1.0	🔗 🗑

Action history for iMX8MM_EVK

Active	Distribution set	Date and time	Status	Type	Actions
●	i.MX8MM_EVK:...	Jul 1 10:48 SGT...	●	⚡	✖ ⚡ ✖

Target: iMX8MM_EVK

Details Description Attributes 🔎

Controller Id: iMX8MM_EVK
Last poll: Thu Jul 1 10:48:06 SGT 2021
Address: http://10.192.245.172
Security token: 6c5703dc072f17a9a847

Total Targets: 2

Distribution set: iMX8QXP_MEK:1.0

Details Description Modules 🔎

Type: App(s) only
Required Migration Step: No

Filter by Status: Gray, Green, Yellow, Red, Blue
Filter by Overdue
Custom Filter

Documentation

hawkBit

Deployment(cont.)

hawkBit Update Server Not secure | 192.168.35.3:8080/UI/#!deployment

Deployment Management

Targets

Name	Status	Actions
iMX8QXP_c0_MEK	● ○ 🔍 🔞	🔗 🗑️
iMX8MM_EVK	● ○ 🔍 🔞	🔗 🗑️

Distributions

Name	Version	Actions
i.MX8MM_EVK	1.0	🔗 🗑️
iMX8QXP_MEK	1.0	🔗 🗑️

Action history for iMX8QXP_c0_MEK

Active	Distribution set	Date and time	Status	Type	Actions
○	iMX8QXP_MEK...	Jul 1 10:48 SGT...	●	⚡	✖️ ⚡️

Target: iMX8QXP_c0_MEK

Details	Description	Attributes
Controller Id: iMX8QXP_c0_MEK		🔗
Last poll: Thu Jul 1 10:47:56 SGT 2021		🔗
Address: http://10.192.245.145		🔗
Security token: f0b768735f5b5d22c078!		🔗

Distribution set: i.MX8MM_EVK:1.0

Details	Description	Modules
Type: App(s) only		🔗
Required Migration Step: No		🔗

Total Targets: 2

Deployment(cont.)

hawkBit Update Server Not secure | 192.168.35.3:8080/UI/#!deployment

Deployment Management

Targets

Name	Status	Actions
iMX8QXP_c0_MEK		
iMX8MM_EVK		

Distributions

Name	Version	Actions
i.MX8MM_EVK	1.0	
iMX8QXP_MEK	1.0	

Action history for iMX8MM_EVK

Active	Distribution set	Date and time	Status	Type	Actions
	i.MX8MM_EVK:...	Jul 1 10:48 SGT...			

Target: iMX8MM_EVK

Distribution set: i.MX8MM_EVK:1.0

Total Targets: 2

Deployment(cont.)

hawkBit Update Server Not secure | 192.168.35.3:8080/UI/#!deployment

DEFAULT

Deployment Management

Targets

Name	Status	Actions
iMX8QXP_c0_MEK		
iMX8MM_EVK		

Distributions

Name	Version	Actions
i.MX8MM_EVK	1.0	
IMX8QXP_MEK	1.0	

Action history for iMX8MM_EVK

Active	Distribution set	Date and time	Status	Type	Actions
	i.MX8MM_EVK...	Jul 1 10:48 SGT...			

Target: iMX8MM_EVK

Details	Description	Attributes
Controller Id: iMX8MM_EVK		
Last poll: Thu Jul 1 10:54:56 SGT 2021		
Address: http://10.192.245.172		
Security token: 6c5703dc072f17a9a847...		

Distribution set: i.MX8MM_EVK:1.0

Details	Description	Modules
Type: App(s) only		
Required Migration Step: No		

Total Targets: 2

hawkBit

Implementation



The major work list

- The SWUpdate Yocto integration & upgrade logic
- Image Assembling
- Upgrade Image build

Swupdate Yocto Integration Upgrade Logic



Preface

- Suppose you already have skills to build i.MX Yocto Linux BSP release.
- Suppose you already have skills to flash the board.
- The Yocto Integration goes to L5.4.70-2.3.0 and LF_v5.10.9_1.0.0.

meta-swupdate: building with Yocto

Follow the SWUpdate documents and add the SWUpdate into the i.MX Yocto Linux BSP release

<https://sbabic.github.io/swupdate>

<https://sbabic.github.io/swupdate/building-with-yocto.html>

Suppose you already have the i.MX Yocto Linux BSP release source code

L5.4.70-2.3.0:

cd sources

git clone <https://github.com/sbabic/meta-swupdate.git> -b zeus

LF_v5.10.9_1.0.0:

cd sources

git clone <https://github.com/sbabic/meta-swupdate.git> -b gatesgarth

Patches And Key Changes Review



Patches

Please read the readme.txt in patches folder and apply the patch.

```
swupdate_yocto_patches/L5.4.70-2.3.0/
|-- 0000-u-boot-default-env-L5.4.70-2.3.0.patch
|-- 1000-swupdate-zeus-enable-systemd-add mmc-utils-ext4-cpio-
    resize2fs-L5.4.70-2.3.0.patch
|-- 1001-swupdate-zeus-defconfig-w-security-L5.4.70-2.3.0.patch
|-- 1002-swupdate-zeus-add-swupdate-sysrestart-L5.4.70-2.3.0.patch
|-- 1002-swupdate-zeus-add-swupdate-sysrestart-security-L5.4.70-
    2.3.0.patch
|-- 1003-swupdate-zeus-add-swupdate-cgi-swupdate_bbappend-L5.4.70-
    2.3.0.patch
|-- 1100-swupdate-zeus-mask-recipes-devtools-L5.4.70-2.3.0.patch
|-- 1101-swupdate-zeus-remove-no-necessary-app-in-swupdate-image-
    L5.4.70-2.3.0.patch
|-- 2000-u-boot-append-emmc-L5.4.70-2.3.0.patch
|-- 2000-u-boot-append-L5.4.70-2.3.0.patch
|-- 3000-build-conf-L5.4.70-2.3.0.patch
|-- 3001-imx-setup-release.sh-L5.4.70-2.3.0.patch
|-- 3002-no-include_kernels_in_8mm_images-L5.4.70-2.3.0.patch
|-- 4000-udev-mount.blacklist-L5.4.70-2.3.0.patch
|-- 4001-remove-u-boot-fw-utils-L5.4.70-2.3.0.patch
|-- 4002-add-systemd-swusys-L5.4.70-2.3.0.patch
`-- readme.txt
```

```
swupdate_yocto_patches/LF_v5.10.9_1.0.0/
|-- 1000-swupdate-gatesgarth-enable-systemd-add mmc-utils-ext4-cpio-
    resize2fs-LF_v5.10.9_1.0.0.patch
|-- 1001-swupdate-gatesgarth-defconfig-w-security-LF_v5.10.9_1.0.0.patch
|-- 1002-swupdate-gatesgarth-add-swupdate-sysrestart-
    LF_v5.10.9_1.0.0.patch
|-- 1002-swupdate-gatesgarth-add-swupdate-sysrestart-security-
    LF_v5.10.9_1.0.0.patch
|-- 1003-swupdate-gatesgarth-add-swupdate-cgi-swupdate_bbappend-
    LF_v5.10.9_1.0.0.patch
|-- 1101-swupdate-gatesgarth-remove-no-necessary-app-in-swupdate-
    image-LF_v5.10.9_1.0.0.patch
|-- 2000-u-boot-append-emmc-LF_v5.10.9_1.0.0.patch
|-- 2000-u-boot-append-LF_v5.10.9_1.0.0.patch
|-- 3000-build-conf-LF_v5.10.9_1.0.0.patch
|-- 3001-imx-setup-release.sh-LF_v5.10.9_1.0.0.patch
|-- 3002-no-include_kernels_in_8mm_images-LF_v5.10.9_1.0.0.patch
|-- 4000-udev-mount.blacklist-LF_v5.10.9_1.0.0.patch
|-- 4002-add-systemd-swusys-LF_v5.10.9_1.0.0.patch
`-- readme.txt
```

Key Changes Review

`fw_printenv/fw_setenv` are the applications could modify the u-boot environment variables from Linux side.

In the demo, Yocto build from `u-boot-fw-utils` and it can also be built from u-boot.

This is the way to communicate between u-boot and Linux

The following command is in Linux set the u-boot environment variables bootslot to dualB:

```
fw_setenv bootslot dualB
```

Key Changes Review (cont.)

CONFIG_SYS_REDUNDAND_ENVIRONMENT gives u-boot can have two copies of environment to store, which can be a backup for each other.

u-boot=> savee

Saving Environment to MMC... Writing to MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to redundant MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to redundant MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to MMC(2)... OK

u-boot=>

Key Changes Review (cont.)

CONFIG_SYS_REDUNDAND_ENVIRONMENT gives u-boot can have two copies of environment to store, which can be a backup for each other.

u-boot=> savee

Saving Environment to MMC... Writing to MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to redundant MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to redundant MMC(2)... OK

u-boot=> savee

Saving Environment to MMC... Writing to MMC(2)... OK

u-boot=>

Key Changes Review (cont.)

Yocto build add SWUpdate mandatory configuration:

/etc/fw_env.config --- configuraton for fw_printenv/fw_setenv

/etc/hwrevision --- hardware revision, SWUpdate to check the hardware compatibility

/etc/swupdate.cfg --- SWUpdate configuration file. In this demo,just tell the public key location

/etc/swu_public.pem --- public key for sign image checking

Key Changes Review (Cont.)

u-boot CONFIG_BOOTCOUNT_BOOTLIMIT is used to trigger the u-boot to execute the altbootcmd.

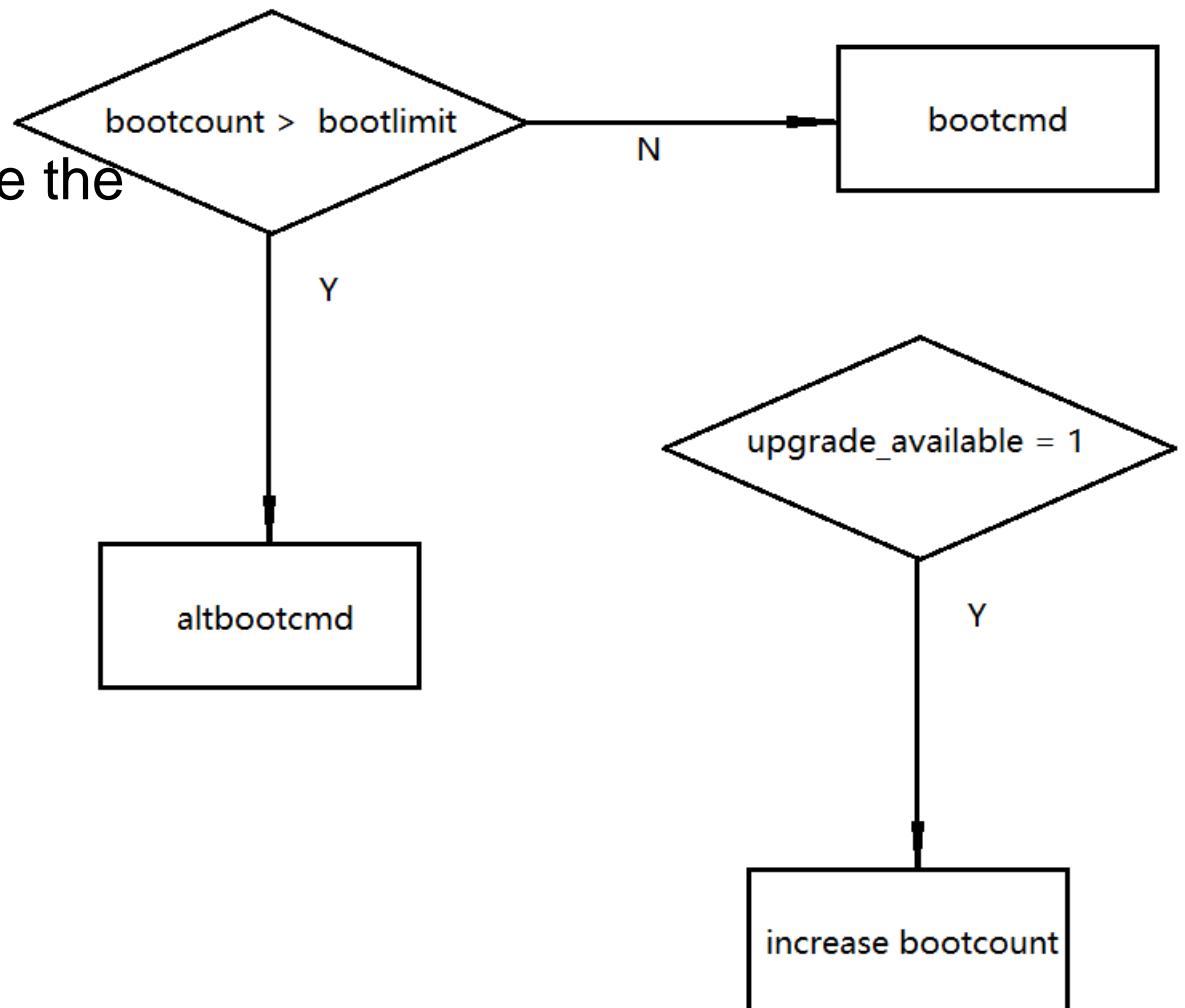
In this demo, bootlimit=3

When upgrade_available = 1, u-boot will increase the bootcount.

Patch:

2000-u-boot-append-Lxxx_1.0.0.patch

2000-u-boot-append-emmc-Lxxx.patch



Key Changes Review(Cont.)

systemd swusys(clrupstatus) service is to clean up the upgrade status by clean the upgrade_available. The same time, reset the bootcount to 0. When boot to rootfs successfully, the clrupstatus.service will be invoked to clean up the upgrade status.

Patch:

4002-add-systemd-swusys-Lxxx.patch

/etc/systemd/system/multi-user.target.wants/clrupstatus.service

[Unit]

Description=clean up update status

[Service]

Type=oneshot

ExecStart=sh -c 'if [\$(fw_printenv -n upgrade_available) -eq 1]; then fw_setenv upgrade_available 0 && fw_setenv bootcount 0 ; fi'

[Install]

WantedBy=multi-user.target

Key Changes Review(Cont.)

sysinfo.cgi is cgi to know about the system information.

Patch: 1003-swupdate-xxxxxx-add-swupdate-cgi-swupdate_2020.11.bb-Lxxx.patch

url: <http://target board ip:8080/sysinfo.cgi>

```
#!/bin/bash
```

Target board:
[/www/sysinfo.cgi](http://target board ip:8080/www/sysinfo.cgi)

```
echo "Content-type: text/html"
echo ""
echo '<html>'
echo '<head>'
echo '<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">'
echo '<title>system information</title>'
echo '</head>'
echo '<body>'
echo "$(uname -a) </br>"
echo $(cat /proc/cmdline)
echo '</body>'
echo '</html>'
exit 0
```

Build



Build

The same as build i.MX Yocto to apply build configurations and create a build

```
DISTRO=fsl-imx-xwayland MACHINE=imx8mmevk source imx-setup-release.sh -b build-xwayland-swupdate-imx8mmevk
```

For the SWUpade image(ram disk) build:

```
bitbake swupdate-image
```

After build, you will find swupdate-image-imx8mmevk.cpio.gz.u-boot.

swupdate-image-imx8mmevk.cpio.gz.u-boot ram disk will be used as single copy update and recovery.

Build(cont.)

For the double copy, you just run command as i.MX Yocto documents tells you.

Here is the command used for this demo:

```
bitbake imx-image-multimedia
```

After build, the SWUpdate has been integrated inside the images.

Image Assembling

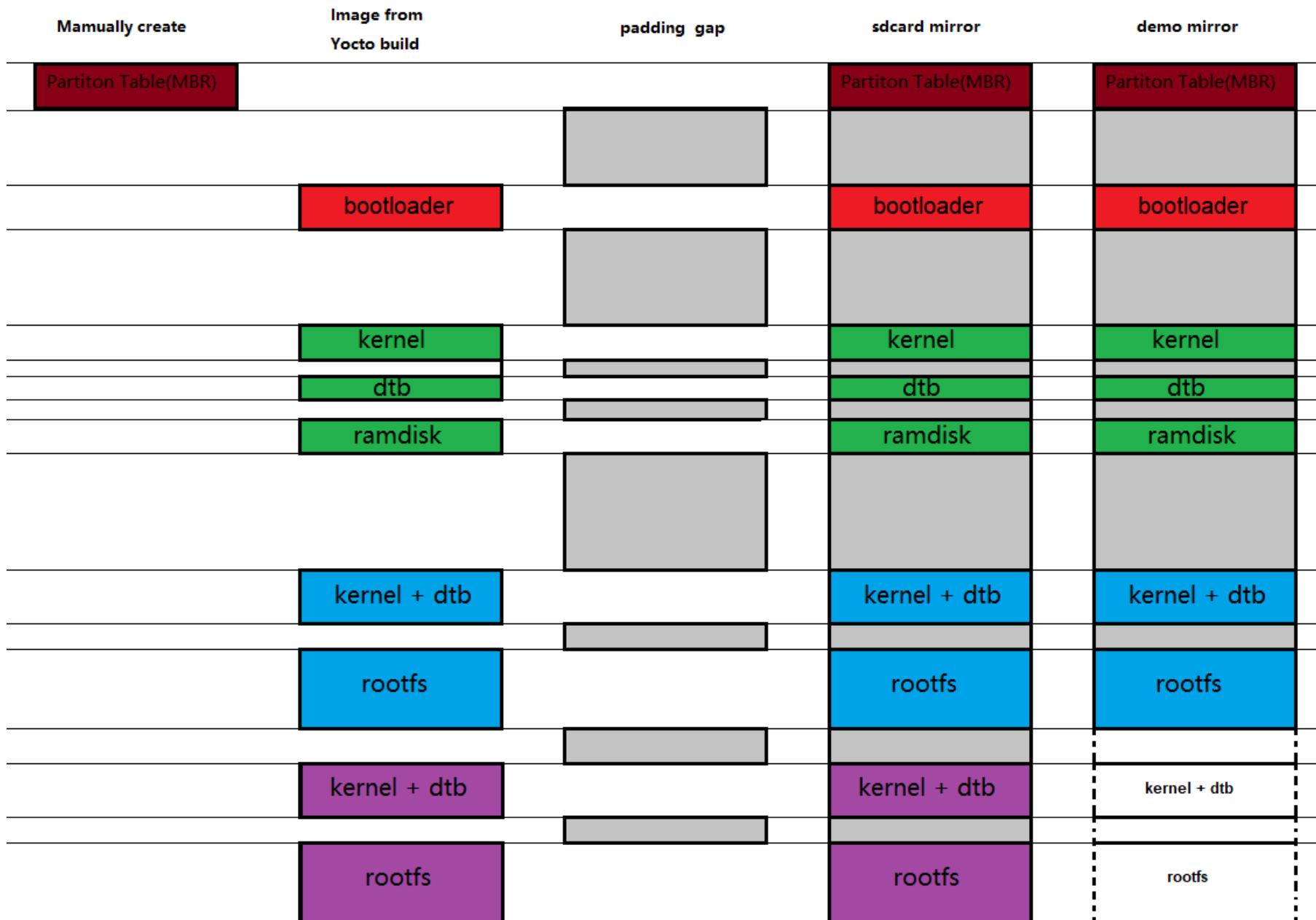


SDCARD Mirror Image

We can use a really SDcard or use kpartx to create sdcard mirror.

Here we introduce how to “cat” to create a SDCARD mirror.

Note: It is for MBR
GPT need to the secondary table at the tail.



Prepare Partition Table

```
truncate -s 7500M swu_dualslot_7.5G.pt
sudo parted -s swu_dualslot_7.5G.pt mklabel msdos
sudo parted swu_dualslot_7.5G.pt unit MiB mkpart primary fat32 100 220
sudo parted swu_dualslot_7.5G.pt unit MiB mkpart primary ext4 220 3220

sudo parted swu_dualslot_7.5G.pt unit MiB mkpart primary fat32 3220 3340
sudo parted swu_dualslot_7.5G.pt unit MiB mkpart primary ext4 3340 6340
sudo parted swu_dualslot_7.5G.pt unit MiB print
```

Sector size (logical/physical): 512B/512B

Partition Table: msdos

Disk Flags:

Number	Start	End	Size	Type	File system	Flags
1	100MiB	220MiB	120MiB	primary		lba
2	220MiB	3220MiB	3000MiB	primary		
3	3220MiB	3340MiB	120MiB	primary		lba
4	3340MiB	6340MiB	3000MiB	primary		

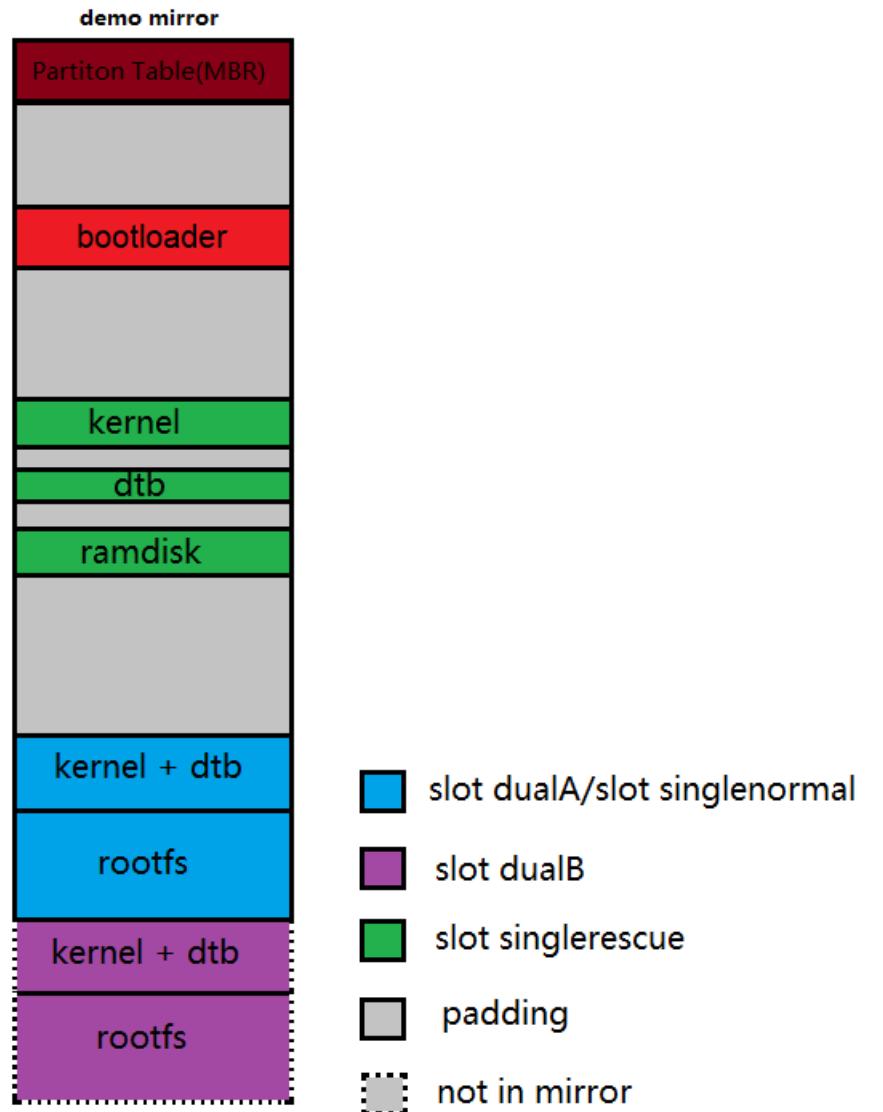
```
truncate -s 512 swu_dualslot_7.5G.pt
```

Prepare Partition Table(cont.)

The layout is for this demo.

The slot dualA and dualB has no gap.

Please make sure the partition is enough to hold the image built from Yocto.



Assembling Demo Image - padding_file_create.sh

padding_file_create.sh is to create padding files.

usage:

./padding_file_create.sh <pad start> <pad filename> <pad end>

./padding_file_create.sh <pad filename> <pad end>

<pad start> and <pad end>

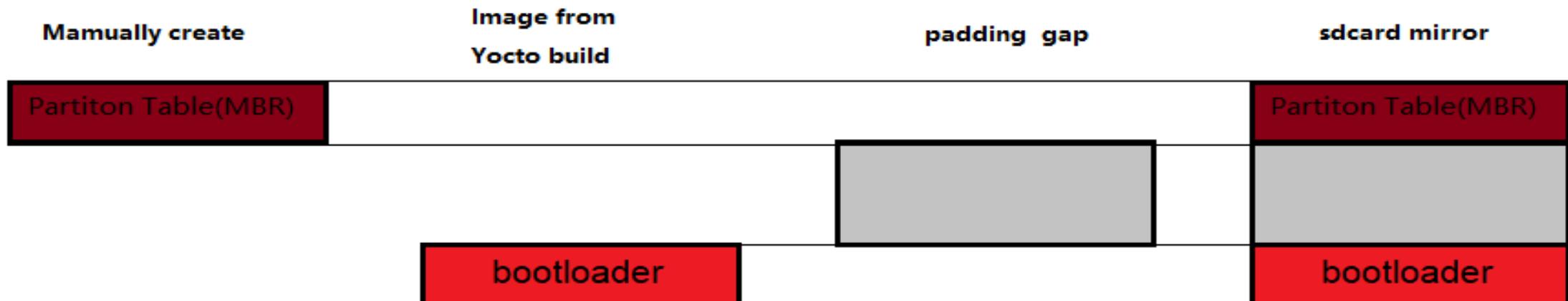
K = 1024, M = 1024*1024, G = 1024*1024*1024, and so on for T, P, E, Z, Y.

Example: ./padding_file_create.sh 0 swu_7.5G.pt 32K

swu_dualslot_7.5G.pt swu_dualslot_7.5G.pt _0_to_32K.pad

cat swu_dualslot_7.5G.pt swu_dualslot_7.5G.pt _0_to_32K.pad imx-boot-imx8mmevk-sd.bin-flash_evk > imx_8mm_boot.scard

Flash imx_8mm_boot.scard to sdcard. You can boot the board to the u-boot.



Assembling Demo Image - SWUpdate ramdisk + slota

```
image_assembling/imx8mmevk/
|-- common
|   '-- swu_dualslot_7.5G.pt
|-- readme.txt
|-- slota -> ../../images/slota_8mm_emmc
|-- slotb -> ../../images[slotb_8mm_emmc
`-- workspace
    |-- 00-swu_7.5G.pt -> ./common/swu_dualslot_7.5G.pt
    |-- 01-imx-boot -> ./slota/imx-boot-imx8mmevk-sd.bin-flash_evk
    |-- 02-Image -> ./slota/Image
    |-- 03-imx8mm-evk.dtb -> ./slota/imx8mm-evk.dtb
    |-- 04-swupdate-image -> ./slota/swupdate-image-imx8mmevk.cpio.gz.u-boot
    |-- 05-boot_pt -> ./common/slota_boot_pt_120M.mirror
    |-- 06-imx-image-multimedia -> ./slota/imx-image-multimedia-imx8mmevk.ext4
    |-- 12-Image -> ./slotb/Image
    |-- 13-imx8mm-evk.dtb -> ./slotb/imx8mm-evk.dtb
    |-- 15-boot_pt -> ./common/slota_boot_pt_120M.mirror
    |-- 16-imx-image-multimedia -> ./slotb/imx-image-multimedia-imx8mmevk.ext4
    '-- padding_file_create.sh
                                image_assembling_imx8mmevk.tgz
```

Assembling Demo Image - SWUpdate ramdisk + slota(cont.)

image_assembling/imx8mmevk/readme.txt

a. prepare boot partition mirror file

```
truncate -s 120M common/slot_a_boot_pt_120M.mirror  
truncate -s 120M common/slot_b_boot_pt_120M.mirror
```

b. create slota slotb symbol link

```
ln -s <yocto/deploy image directory a> slota  
ln -s <yocto/deploy image directory b> slotb
```

c. assemble image

```
cd workspace
```

```
mkfs.vfat 05-boot_pt
```

```
mkdir -i 05-boot_pt  
mcopy -i 05-boot_pt 03-imx8mm-evk.dtb ::imx8mm-evk.dtb  
mcopy -i 05-boot_pt 02-Image ::Image  
mkdir -i 05-boot_pt
```

```
./padding_file_create.sh 0k 00-swu_7.5G.pt 33K
```

```
./padding_file_create.sh 33K 01-imx-boot 8M
```

```
./padding_file_create.sh 8M 02-Image 38M
```

```
./padding_file_create.sh 38M 03-imx8mm-evk.dtb 42M
```

```
./padding_file_create.sh 42M 04-swupdate-image 100M
```

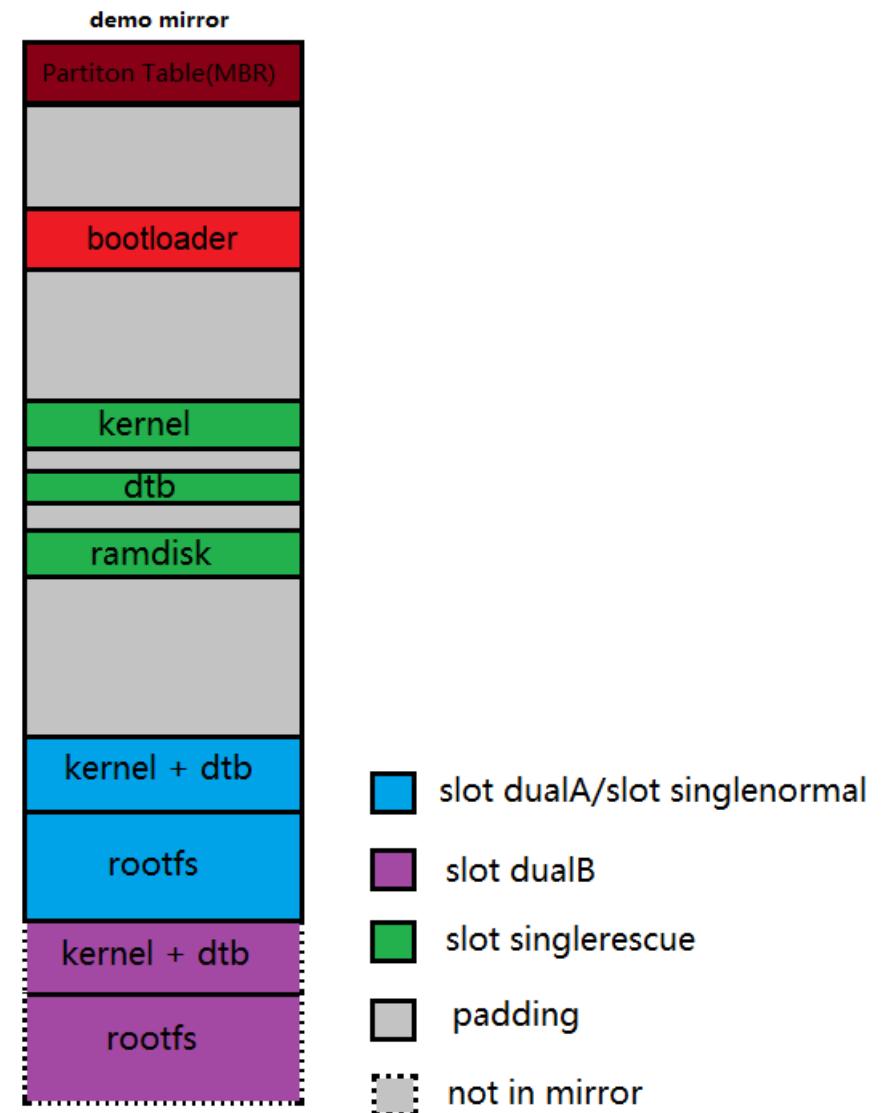
```
truncate -s 3000M 06-imx-image-multimedia
```

```
e2fsck -f 06-imx-image-multimedia
```

```
resize2fs 06-imx-image-multimedia
```

```
var_storage=emmc&& var_DATE=$(date "+%Y%m%d-%H%M%S") && cat $(ls 0*) > swu_slot_a_w_swu_rescue_imx8mm_${var_storage}_${var_DATE}.sdcard
```

```
uuu -b emmc_all imx-boot-imx8mmevk-sd.bin-flash_evk xxxxxxx.sdcards
```



Assembling Demo Image - SWUpdate ramdisk + slota + slotb

image_assembling/imx8mmevk/readme.txt

a. prepare boot partition mirror file

```
truncate -s 120M common/slotA_boot_pt_120M.mirror  
truncate -s 120M common/slotB_boot_pt_120M.mirror
```

b. create slota slotb symbol link

```
ln -s <yocto/deploy image directory a> slotA  
ln -s <yocto/deploy image directory b> slotB
```

c. assemble image

```
cd workspace
```

```
mkfs.vfat 05-boot_pt
```

```
mkdir -i 05-boot_pt  
mcopy -i 05-boot_pt 03-imx8mm-evk.dtb ::imx8mm-evk.dtb  
mcopy -i 05-boot_pt 02-Image ::Image  
mkdir -i 05-boot_pt
```

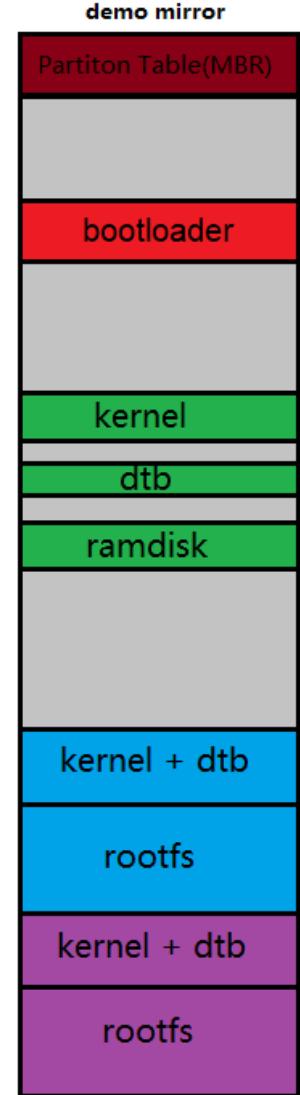
```
./padding_file_create.sh 0k 00-swu_7.5G.pt 33K  
./padding_file_create.sh 33K 01-imx-boot 8M  
./padding_file_create.sh 8M 02-Image 38M  
./padding_file_create.sh 38M 03-imx8mm-evk.dtb 42M  
./padding_file_create.sh 42M 04-swupdate-image 100M
```

```
truncate -s 3000M 06-imx-image-multimedia  
e2fsck -f 06-imx-image-multimedia  
resize2fs 06-imx-image-multimedia
```

```
var_storage=emmc&& var_DATE=$(date "+%Y%m%d-%H%M%S") && cat $(ls 0*) 15-boot_pt 16-imx-image-multimedia > swu_slotA_w_swu_rescue_imx8mm_${var_storage}_${var_DATE}.sdcard
```

```
uuu -b emmc_all imx-boot-imx8mmevk-sd.bin-flash_evk xxxxxxx.sdcards
```

- █ slot dualA/slot singlenormal
- █ slot dualB
- █ slot singlerescue
- █ padding
- █ not in mirror



```
mkfs.vfat 15-boot_pt
```

```
mkdir -i 15-boot_pt  
mcopy -i 15-boot_pt 13-imx8mm-evk.dtb ::imx8mm-evk.dtb  
mcopy -i 15-boot_pt 12-Image ::Image  
mkdir -i 15-boot_pt
```

```
truncate -s 3000M 16-imx-image-multimedia  
e2fsck -f 16-imx-image-multimedia  
resize2fs 16-imx-image-multimedia
```

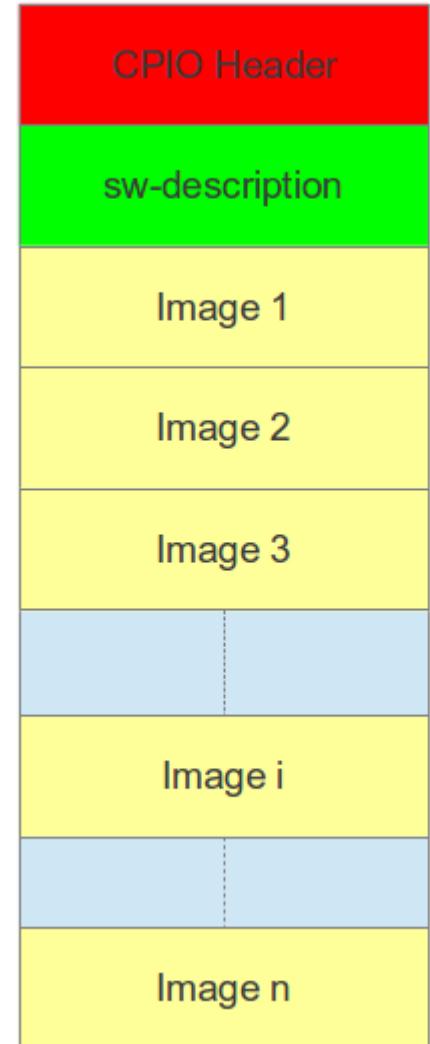
Update Image Build



SWUpdate Image

The main concept is that the manufacturer delivers a single big image. All single images are packed together (cpio was chosen for its simplicity and because can be streamed) together with an additional file (sw-description), that contains meta information about each single image.

SWUpdate is thought to be able to stream the received image directly into the target, without any temporary copy.



SWUpdate: syntax and tags with the default parser

For sw-description

Please refer to the <https://sbabic.github.io/swupdate/sw-description.html> for complete guide.
In this presentation, just show several typical cases.

```
software =  
{  
  
    version = "1.0";  
    description = "bootloader update for test Project";  
    hardware-compatibility: [ "1.0", "1.2", "1.3"];  
    imx8mmevk: {  
  
        images: (  
            {  
                version ="5.4.70";  
                install-if-different = true;  
                name = "bootloader";  
                filename = "imx-boot-imx8mmevk-sd.bin-flash_evk";  
                sha256 = "faa3249e6940376093110b24b697ff4598587d48226b6732d9046aedcf6930ed";  
                device = "/dev/mmcblk2boot0";  
            }  
        );  
  
    }  
}
```

Update images from verified source

For signed image, please refer to https://sbabic.github.io/swupdate/signed_images.html

```
software =
{
    version = "1.0";
    description = "Firmware update for test Project";
    hardware-compatibility: ["1.0", "1.2", "1.3"];
    imx8mmevk: {

        images: (
            {
                filename = "imx-boot-imx8mmevk-sd.bin-flash_evk";
                sha256 = "07fb81ac2ab0d7db375a5efe3e6057d1318acac33ccd4a64fe21abd9a4526ffa";
                device = "/dev/mmcblk2boot0";
                offset = "33K";
            }
        );
    };
}
```

Update images from verified source(cont.)

Generate key :

Private key: openssl genrsa -aes256 -out priv.pem

Public key: openssl rsa -in priv.pem -out public.pem -outform PEM -pubout

public.pem is used on target board renamed as /etc/swu_public.pem

Board specific settings

<https://sbabic.github.io/swupdate/sw-description.html#board-specific-settings>

the hardware information file /etc/hwrevision contains hardware information for upgrade check

In this demo:

/etc/hwrevision
imx8mmevk 1.0

eMMC:

/etc/fw_env.config
/dev/mmcblk2 0x400000 0x2000
/dev/mmcblk2 0x402000 0x2000

SDcard:

/etc/fw_env.config
/dev/mmcblk1 0x400000 0x2000
/dev/mmcblk1 0x402000 0x2000

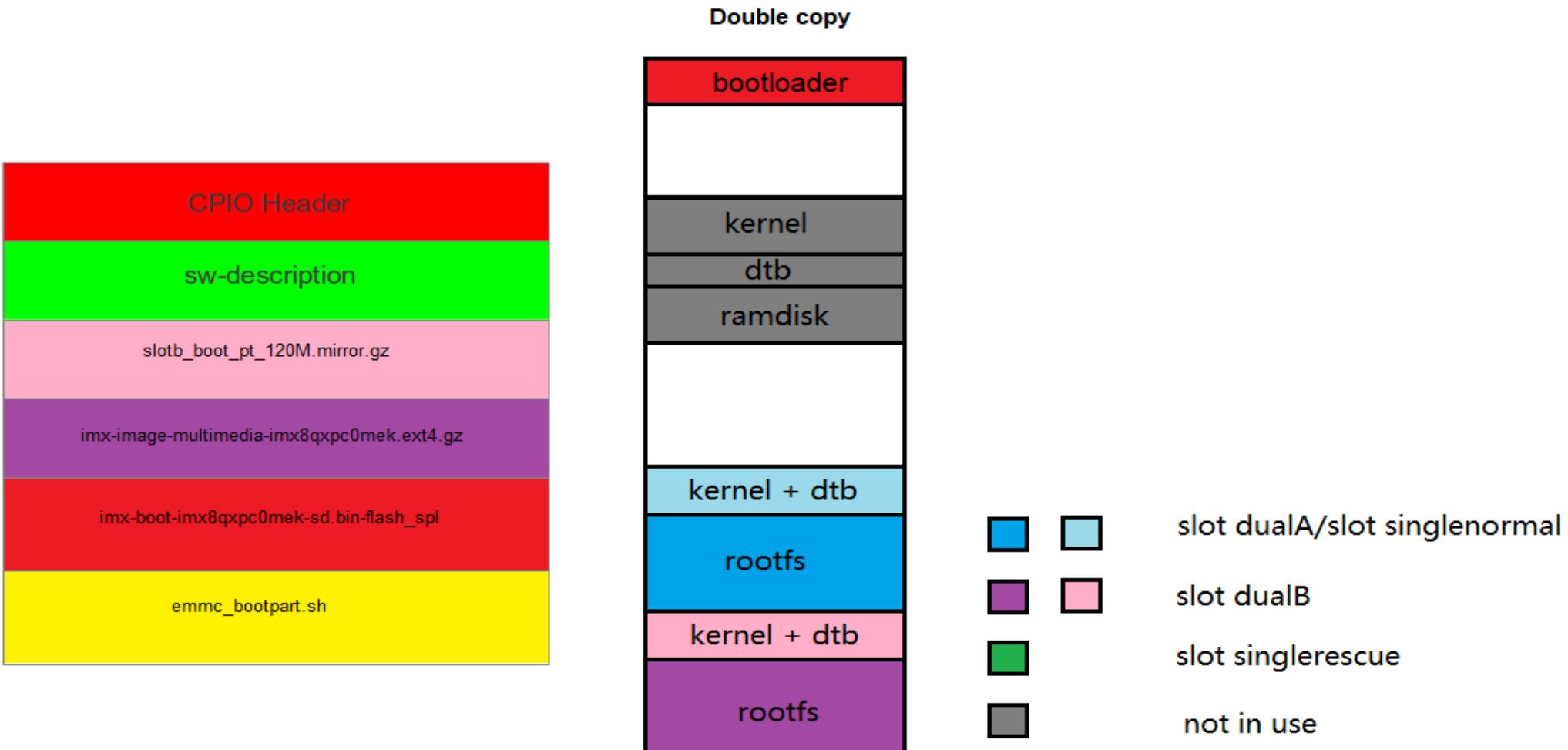
Example of upgrade image create directory

```
swu-image/imx8mmevk/LF_v5.10.9_1.0.0/doublecopy_slotb_bootloader_bootpart_rootfs_emmc/  
|-- emmc_bootpart.sh --- swu post script  
|-- Image           --- linux kernel image  
|-- i.mx8mm_evk_1.0_slotB_LF_v5.10.9_1.0.0-doublecopy-emmc-full_20210713-113324.swu --- swu image  
|-- imx8mm-evk.dtb   --- linux dtb  
|-- imx-boot-imx8mmevk-sd.bin-flash_evk --- bootloader  
|-- imx-image-multimedia-imx8mmevk.ext4 --- rootfs ext4 mirror  
|-- imx-image-multimedia-imx8mmevk.ext4.gz  
|-- priv.pem          --- private key  
|-- readme.txt        --- readme  
|-- slotb_boot_pt_120M.mirror      --- boot partition fat mirror  
|-- slotb_boot_pt_120M.mirror.gz  
|-- sw-description      --- description (meta) of the images  
|-- sw-description.sig  
`-- swu_signed_image_build.sh    --- upgrade image build script
```

Example
Double Copy Update
slotb bootloader, boot
partition, rootfs On
eMMC



Double Copy update slotb bootloader, boot partition, rootfs on eMMC



Double Copy update slotb bootloader, boot partition, rootfs on eMMC(cont.)

The directory for creating upgrade image

imx8mmevk_doublecopy_slotb_full_LF_v5.10.9_1.0.0.tgz

```
swu-image/imx8mmevk/LF_v5.10.9_1.0.0/doublecopy_slotb_bootloader_bootpart_rootfs_emmc/  
|-- emmc_bootpart.sh  
|-- priv.pem  
|-- readme.txt  
|-- sw-description  
`-- swu_signed_image_build.sh
```

Double Copy update slotb bootloader, boot partition, rootfs on eMMC(cont.)

swu-image/imx8mmevk/LF_v5.10.9_1.0.0/doublecopy_slotb_bootloader_bootpart_rootfs_emmc/readme.txt

a. truncate -s 120M slotb_boot_pt_120M.mirror --- only run for first time.
b. cp <yocto/deploy image directory>/{Image,imx8mm-evk.dtb,imx-boot-imx8mmevk-sd.bin-flash_evk,imx-image-multimedia-imx8mmevk.ext4} .
c. run following command
mkfs.vfat slotb_boot_pt_120M.mirror

mkdir -i slotb_boot_pt_120M.mirror

mcopy -i slotb_boot_pt_120M.mirror imx8mm-evk.dtb ::imx8mm-evk.dtb
mcopy -i slotb_boot_pt_120M.mirror Image ::Image

mkdir -i slotb_boot_pt_120M.mirror

gzip -9k slotb_boot_pt_120M.mirror

truncate -s 3000M imx-image-multimedia-imx8mmevk.ext4
e2fsck -f imx-image-multimedia-imx8mmevk.ext4
resize2fs imx-image-multimedia-imx8mmevk.ext4

gzip -9k imx-image-multimedia-imx8mmevk.ext4
d. get sha256sum and modify the sw-description
sha256sum slotb_boot_pt_120M.mirror.gz imx-image-multimedia-imx8mmevk.ext4.gz imx-boot-imx8mmevk-sd.bin-flash_evk emmc_bootpart.sh
e. run ./swu_signed_image_build.sh
pass phrase is test

Double Copy update slotb bootloader, boot partition, rootfs on eMMC(cont.)

sw-description

```
software =  
{  
    version = "1.0";  
    description = "Firmware update for test Project";  
    hardware-compatibility: [ "1.0", "1.2", "1.3"];  
    imx8mmevk: {  
  
        images: (  
            {  
                filename = "slotb_boot_pt_120M.mirror.gz";  
                sha256 = "5082d2289a4ea86c18d647dc5add355b75422cceed9a866c30f6f4b38618376f";  
                compressed = "zlib";  
                device = "/dev/mmcblk2p3";  
            },  
            {  
                filename = "imx-image-multimedia-imx8mmevk.ext4.gz";  
                sha256 = "90b1383c5c3f0a6b3623b8a081d6c58604b6c56ce3f43cd55906dc1afe9ef5e6";  
                compressed = "zlib";  
                device = "/dev/mmcblk2p4";  
            },  
            {  
                filename = "imx-boot-imx8mmevk-sd.bin-flash_evk";  
                sha256 = "07fb81ac2ab0d7db375a5efe3e6057d1318acac33ccd4a64fe21abd9a4526ffa";  
                device = "/dev/mmcblk2boot0";  
                offset = "33K";  
            }  
        );  
        scripts: (  
            {  
                filename = "emmc_bootpart.sh";  
                sha256 =  
                    "76ecf14df4f6830fdb77b41d78092c6d440b8889124b6b38c45ba381367992e8";  
                type = "postinstall";  
            }  
        );  
        bootenv:(  
            {  
                name = "upgrade_available";  
                value = "1";  
            },  
            {  
                name = "bootslot";  
                value = "dualB";  
            }  
        );  
    }  
};
```

Double Copy update slotb bootloader, boot partition, rootfs on eMMC(cont.)

emmc_bootpart.sh

```
echo "enable emmc(/dev/mmcblk2) boot partition 1"  
mmc bootpart enable 1 1 /dev/mmcblk2
```

Double Copy update slotb bootloader, boot partition, rootfs on eMMC(cont.)

swu_signed_image_build.sh

```
#!/bin/bash

MODE="RSA-PKCS-1.5"
PRODUCT_NAME="i.mx8mm_evk"
CONTAINER_VER="1.0"
SLOT="B"
BSP_VER="LF_v5.10.9_1.0.0"
EXTRA_INFO="-doublecopy-emmc-full"
IMAGES=
    slotb_boot_pt_120M.mirror.gz
    imx-image-multimedia-imx8mmevk.ext4.gz
    imx-boot-imx8mmevk-sd.bin-flash_evk
    emmc_bootpart.sh
"
FILES="sw-description sw-description.sig $IMAGES"
build_DATE="$(date "+%Y%m%d-%H%M%S")"

for IMG in ${IMAGES}; do test ! -e "${IMG}" && echo "${IMG} not exists!!!" && exit -1; done

#if you use RSA
if [ x"$MODE" == "xRSA-PKCS-1.5" ]; then
    openssl dgst -sha256 -sign priv.pem sw-description > sw-description.sig
elif [ x"$MODE" == "xRSA-PSS" ]; then
    openssl dgst -sha256 -sign priv.pem -sigopt rsa_padding_mode:pss -sigopt rsa_pss_saltlen:-2 sw-description > sw-description.sig
else
    openssl cms -sign -in sw-description -out sw-description.sig -signer mycert.cert.pem -inkey mycert.key.pem -outform DER -nosmimecap -binary
fi

94 for i in $FILES; do
    echo $i;done | cpio -ov -H crc > ${PRODUCT_NAME}_${CONTAINER_VER}_slot${SLOT}_${BSP_VER}${EXTRA_INFO}_${build_DATE}.swu
```



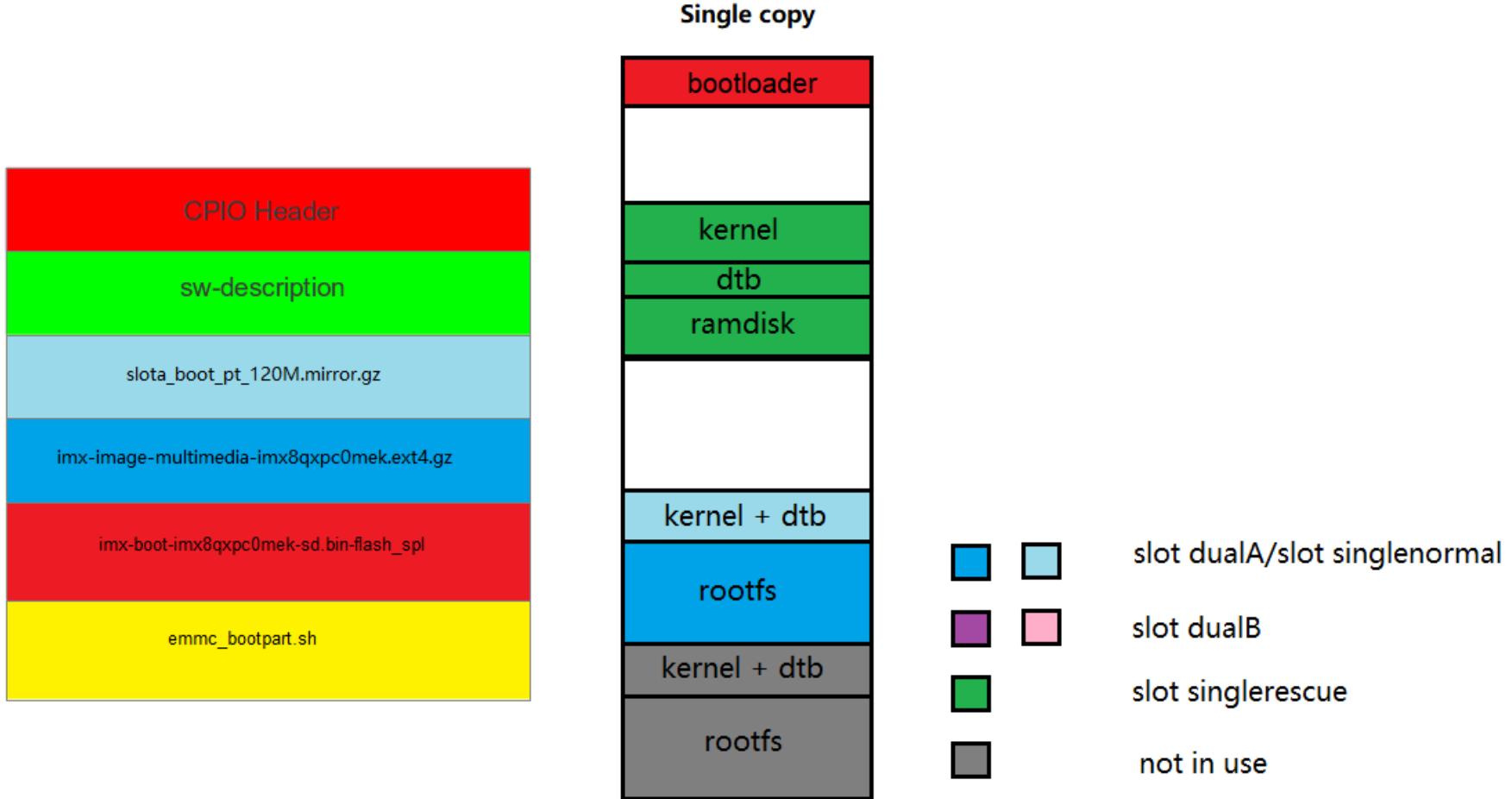
Example

Single Copy Update

bootloader, boot partition, rootfs On eMMC



Single Copy update slots bootloader, boot partition, rootfs on eMMC



Single Copy update slots bootloader, boot partition, rootfs on eMMC(cont.)

The directory for creating upgrade image

imx8mmevk_singlecopy_slots_full_LF_v5.10.9_1.0.0.tgz

```
swu-image/imx8mmevk/LF_v5.10.9_1.0.0/singlecopy_slots_bootloader_bootpart_rootfs_emmc/  
|-- emmc_bootpart.sh  
|-- priv.pem  
|-- readme.txt  
|-- sw-description  
`-- swu_signed_image_build.sh
```

Single Copy update slots bootloader, boot partition, rootfs on eMMC(cont.)

swu-image/imx8mmevk/LF_v5.10.9_1.0.0/singlecopy_slots_bootloader_bootpart_rootfs_emmc/readme.txt

a. truncate -s 120M slots_boot_pt_120M.mirror --- only run for first time.

b. cp <yocto/deploy image directory>/{Image,imx8mm-evk.dtb,imx-boot-imx8mmevk-sd.bin-flash_evk,imx-image-multimedia-imx8mmevk.ext4} .

c. run following command

```
mkfs.vfat slots_boot_pt_120M.mirror
```

```
mkdir -i slots_boot_pt_120M.mirror
```

```
mcopy -i slots_boot_pt_120M.mirror imx8mm-evk.dtb ::imx8mm-evk.dtb
```

```
mcopy -i slots_boot_pt_120M.mirror Image ::Image
```

```
mkdir -i slots_boot_pt_120M.mirror
```

```
gzip -9k slots_boot_pt_120M.mirror
```

```
truncate -s 3000M imx-image-multimedia-imx8mmevk.ext4
```

```
e2fsck -f imx-image-multimedia-imx8mmevk.ext4
```

```
resize2fs imx-image-multimedia-imx8mmevk.ext4
```

```
gzip -9k imx-image-multimedia-imx8mmevk.ext4
```

d. get sha256sum and modify the sw-description

```
sha256sum slots_boot_pt_120M.mirror.gz imx-image-multimedia-imx8mmevk.ext4.gz imx-boot-imx8mmevk-sd.bin-flash_evk emmc_bootpart.sh
```

e. run ./swu_signed_image_build.sh

pass phrase is test



Single Copy update slots bootloader, boot partition, rootfs on eMMC(cont.)

sw-description

```
software =  
{  
    version = "1.0";  
    description = "Firmware update for test Project";  
    hardware-compatibility: [ "1.0", "1.2", "1.3"];  
    imx8mmevk: {  
  
        images: (  
            {  
                filename = "slotb_boot_pt_120M.mirror.gz";  
                sha256 = "5082d2289a4ea86c18d647dc5add355b75422cceed9a866c30f6f4b38618376f";  
                compressed = "zlib";  
                device = "/dev/mmcblk2p1";  
            },  
            {  
                filename = "imx-image-multimedia-imx8mmevk.ext4.gz";  
                sha256 = "90b1383c5c3f0a6b3623b8a081d6c58604b6c56ce3f43cd55906dc1afe9ef5e6";  
                compressed = "zlib";  
                device = "/dev/mmcblk2p2";  
            },  
            {  
                filename = "imx-boot-imx8mmevk-sd.bin-flash_evk";  
                sha256 = "07fb81ac2ab0d7db375a5efe3e6057d1318acac33ccd4a64fe21abd9a4526ffa";  
                device = "/dev/mmcblk2boot0";  
                offset = "33K";  
            }  
        );  
        scripts: (  
            {  
                filename = "emmc_bootpart.sh";  
                sha256 =  
                    "76ecf14df4f6830fdb77b41d78092c6d440b8889124b6b38c45ba381367992e8";  
                type = "postinstall";  
            }  
        );  
        bootenv:(  
            {  
                name = "upgrade_available";  
                value = "1";  
            },  
            {  
                name = "bootslot";  
                value = "dualB";  
            }  
        );  
    }  
};
```

Single Copy update bootloader, boot partition, rootfs on eMMC(cont.)

Single copy upgrade starts from getting into upgrade mode (slot singlerescue).
After upgrade done, boot to the normal run.

In Linux:

```
fw_setenv bootslot singlerescue  
fw_setenv upgrade_available 1  
reboot
```

You will see u-boot switch to ram disk boot.
Once boot done, the SWUpdate Mongoose is running.
You can use the web browser to do upgrade.

Example

Single Copy Update

kernel, dtb On eMMC



Example Single Copy update kernel, dtb on eMMC

The directory for creating upgrade image

imx8mmevk_singlecopy_slots_kernel_dtb.tgz

swu-image/imx8mmevk/LF_v5.10.9_1.0.0/singlecopy_slots_kernel_dtb

|-- priv.pem

|-- readme.txt

|-- sw-description

`-- swu_signed_image_build.sh

Example Single Copy update kernel, dtb on eMMC(cont.)

`swu-image/imx8mmevk/LF_v5.10.9_1.0.0/singlecopy_slots_kernel_dtb/readme.txt`

- a. cp <yocto/deploy image directory>/{Image,imx8mm-evk.dtb} .
- b. get sha256sum and modify the sw-description
`sha256sum Image imx8mm-evk.dtb`
- c. run `./swu_signed_image_build.sh`
pass phrase is test

Example Single Copy update kernel, dtb on eMMC(cont.)

sw-description

```
software =  
{  
    version = "1.0";  
    description = "Firmware update for test Project";  
    hardware-compatibility: [ "1.0", "1.2", "1.3"];  
    imx8mmevk: {  
        files: (  
            {  
                filename = "Image";  
                path = "/Image";  
                sha256 =  
                    "a89248214c066934d2fce25d234a859edd1245ef3c  
                    bf71d35484bbe60fb9f4f";  
                device = "/dev/mmcblk2p1";  
                filesystem = "vfat";  
            },  
            {  
                filename = "imx8mm-evk.dtb";  
                path = "/imx8mm-evk.dtb";  
                sha256 =  
                    "32b4f8d07c4bd1b58b72ff0168e6402e9bad095f91336d52  
                    31b13dde90775617";  
                device = "/dev/mmcblk2p1";  
                filesystem = "vfat";  
            },  
            {  
                filename = "imx8mm-evk.dtbo";  
                path = "/imx8mm-evk.dtbo";  
                sha256 =  
                    "32b4f8d07c4bd1b58b72ff0168e6402e9bad095f91336d52  
                    31b13dde90775617";  
                device = "/dev/mmcblk2p1";  
                filesystem = "vfat";  
            }  
        );  
    };  
};
```

Example Single Copy update kernel, dtb on eMMC(cont.)

swu_signed_image_build.sh

```
#!/bin/bash

MODE="RSA-PKCS-1.5"
PRODUCT_NAME="i.mx8mmevk"
CONTAINER_VER="1.0"
SLOT="S"
BSP_VER="LF_v5.10.9_1.0.0"
EXTRA_INFO="-singlecopy-emmc-kernel-dtb"
IMAGES="Image
    imx8mm-evk.dtb
"
FILES="sw-description sw-description.sig $IMAGES"
build_DATE="$(date "+%Y%m%d-%H%M%S")"

for IMG in ${IMAGES}; do test ! -e "${IMG}" && echo "${IMG} not exists!!!" && exit -1; done

#if you use RSA
if [ x"$MODE" == "xRSA-PKCS-1.5" ]; then
    openssl dgst -sha256 -sign priv.pem sw-description > sw-description.sig
elif [ x"$MODE" == "xRSA-PSS" ]; then
    openssl dgst -sha256 -sign priv.pem -sigopt rsa_padding_mode:pss -sigopt rsa_pss_saltlen:-2 sw-description > sw-description.sig
else
    openssl cms -sign -in sw-description -out sw-description.sig -signer mycert.cert.pem -inkey mycert.key.pem -outform DER -nosmimecap -binary
fi

for i in $FILES;do
    echo $i;done | cpio -ov -H crc > ${PRODUCT_NAME}_${CONTAINER_VER}_slot${SLOT}_${BSP_VER}${EXTRA_INFO}_${build_DATE}.swu
```



Hawkbit



Hawkbit Server Build

<https://www.eclipse.org/hawkbit/>

<https://www.eclipse.org/hawkbit/gettingstarted/>

Please follow the instruction to build the hawkbit server or run docker version

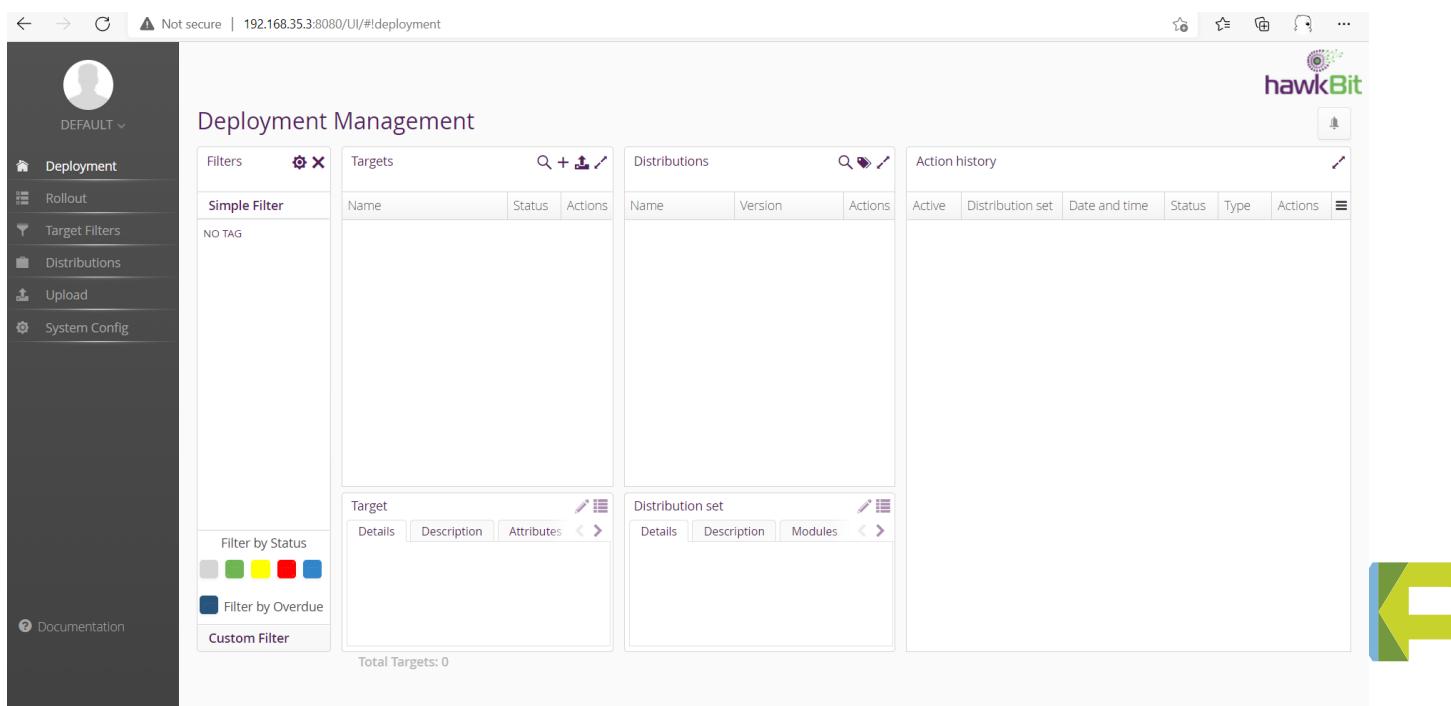
Hawkbit Server Run

Assume you are in the hawkbit directory.

```
java -jar ./hawkbit-runtime/hawkbit-update-server/target/hawkbit-update-server-0.3.0-SNAPSHOT.jar \
--hawkbit.dmf.rabbitmq.enabled=false \
--hawkbit.server.ddi.security.authentication.anonymous.enabled=true
```

Once the hawkbit server is running, you can login and manage the server at <http://ip:8080>

The username & password: admin



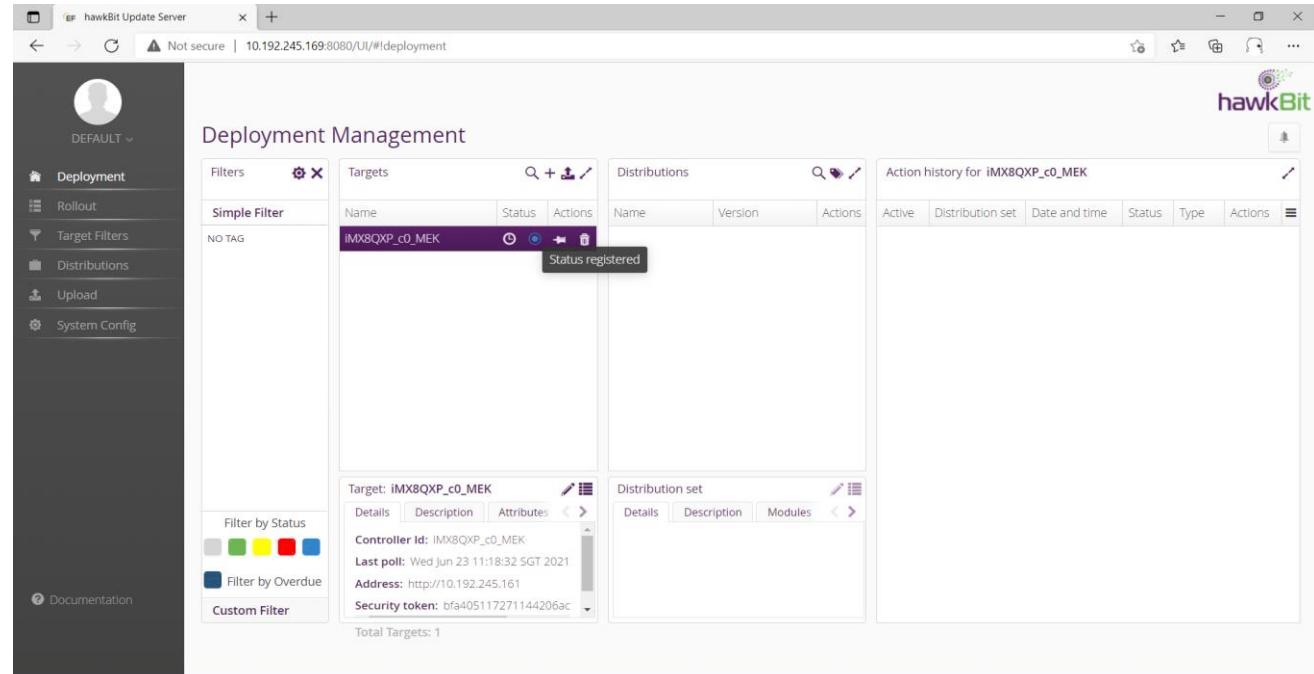
The screenshot shows the Hawkbit Deployment Management interface. The top navigation bar includes a user icon, a dropdown menu, and links for 'Deployment', 'Rollout', 'Target Filters', 'Distributions', 'Upload', and 'System Config'. On the left, a sidebar provides access to 'Documentation' and other system settings. The main content area is titled 'Deployment Management' and contains several sections: 'Targets' (empty), 'Distributions' (empty), 'Action history' (empty), 'Target' (empty), and 'Distribution set' (empty). A legend at the bottom indicates target status colors: grey, green, yellow, red, blue, and dark blue. The URL in the browser is 'http://192.168.35.3:8080/UI/#/deployment'.

Suricatta daemon mode Run

Suricatta daemon mode and Mongoose daemon mode can run at together at the same time.
But for not making this demo too complex. Mongoose daemon mode is default auto start up.

Suricatta daemon mode is manually startup.

```
systemctl stop swupdate  
systemctl stop swupdate-sysrestart  
systemctl start swupdate-sysrestart  
systemctl stop swupdate-progress  
systemctl start swupdate-progress
```



```
swupdate -l 5 -u '-t default -u http://10.192.245.169:8080 -i iMX8MM_EVK' -p "swupdate -u '-t default -u http://10.192.245.169:8080 -i iMX8MM_EVK -c 2"
```

Note: Hawkbit update is scheduled upgrade, you may need to wait for upgrade done.





SECURE CONNECTIONS
FOR A SMARTER WORLD