MA35D1 OTA Update by Linux AB Partitions

Example Code Introduction for 64-bit NuMicro® Family

Document Information

Application	This example describes how to perform OTA updates on the rootfs A/B partitions in a Linux environment using the MA35D1 series development platform.
BSP Version	Linux-5.10.x
Hardware	NuMaker-HMI-MA35D1 V1.0

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1. Overview

OTA (Over-the-Air) is a technology that enables remote updates of firmware, software, or data on a device through network connections such as Wi-Fi or Ethernet. This document provides an example of updating the root filesystem (rootfs) using the **swupdate** application. For additional examples demonstrating how to update the Linux kernel and device tree using **swupdate**, please refer to the corresponding application note EC_MA35D1_OTA_Update_by_Linux_Stage_V1.00.

Since **swupdate** is an application that runs on the active root filesystem, directly using it to update the same root filesystem currently in use is risky. During the update process, it is difficult to determine whether certain files are being accessed by the running system. Updating files that are actively in use may result in simultaneous access conflicts between the update process and the operating system, potentially leading to system instability or unexpected behavior.

To ensure a safe and reliable root filesystem update process, it is both necessary and practical to partition the storage into two independent root filesystem areas. This approach allows the inactive root filesystem to be updated while the system continues running from the active one.

In this document, these two independent root filesystem areas are referred to as partition A (rootfs area A) and partition B (rootfs area B).

1.1 Environment Settings

For MA35D1 environment setup and image generation, please refer to Nuvoton MA35D1 Evaluation Board Quick Start.

The core concept of this example is to create two root filesystem partitions (rootfs A/B partitions) in the Linux storage layout. The **swupdate** is then used to invoke a shell script that updates the inactive root filesystem partition, which is not currently mounted by the running system. The script also automatically updates the U-Boot boot parameters.

After the update is completed, the system reboots and automatically mounts the newly updated root filesystem partition as the active rootfs, thereby completing the OTA root filesystem update process.

The storage layout containing the two rootfs partitions on the MA35D1 platform is illustrated in Figure 1-1. The A/B partition update flow is shown in Figure 1-2.

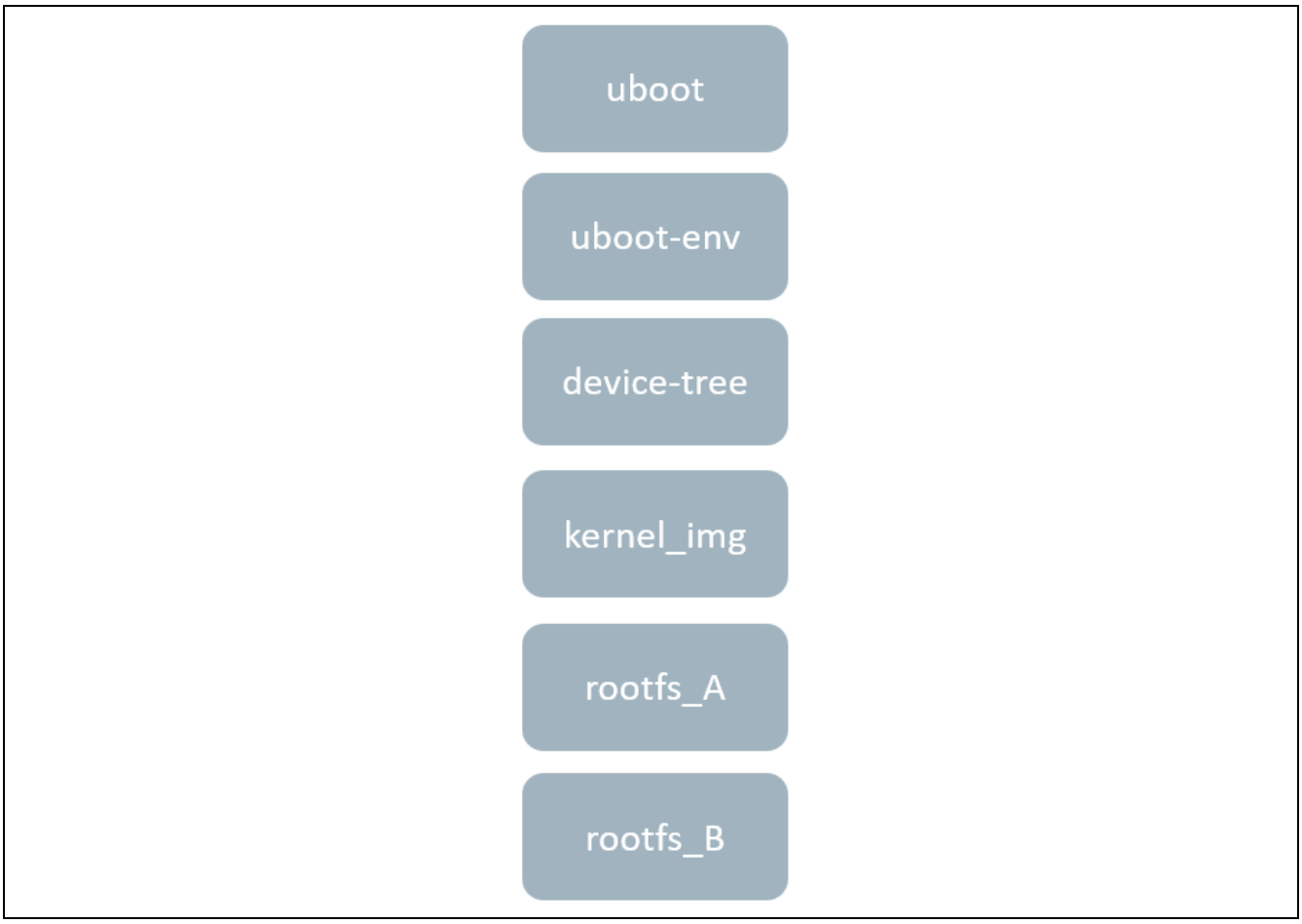


Figure 1-1 MA35D1 Kernel Partition Layout

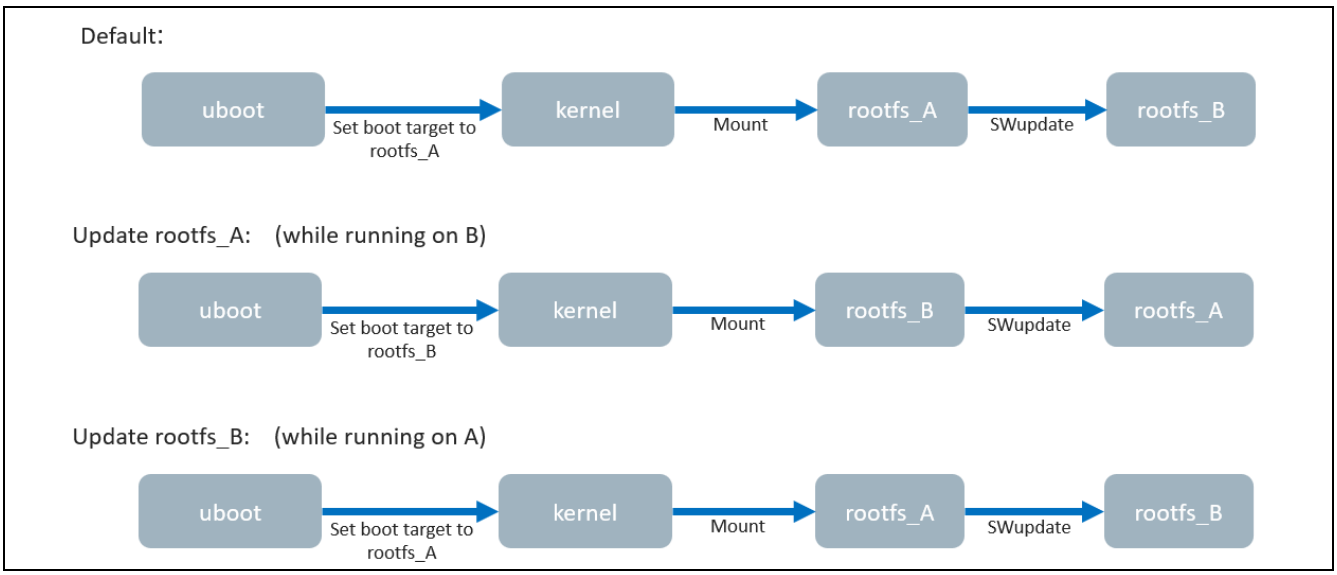


Figure 1-2 swupdate rootfs Update Flowchart

2. Code Description

2.1 eMMC/SD card Update Method

The following describes the rootfs A/B update process for eMMC/SD card:

1. Follow the description in the EC_MA35D1_OTA_Update_by_Linux_Stage_V1.00 to enable the **swupdate** configuration in Buildroot_2024.

   Modify */package/swupdate/swupdate.config* and add the following settings:

   CONFIG_SHELLSCRIPTHANDLER=y
   CONFIG_UBOOT_ENV=y

2. After completing the modifications, rebuild **swupdate**. The modified *swupdate.config* is shown in Figure 2-1.

```
$ make swupdate-rebuild
$ make
```



Figure 2-1 Modified swupdate.config

3. Modify */buildroot_2024/board/nuvoton/ma35xx/nuwriter/pack-sdcard.json* to add the eMMC/SD card space for the rootfs partition B. In this example, the partition B is configured with a size of 1 GB.

```
{
  "image": [
    …
   {
     "offset": "37748736",
     "file": "rootfs.ext4",
      "type": 0
    },
    {
      "offset": "1140850688",
      "file": "rootfs.ext4",
      "type": 0
    }
  ]
}
```

4. Add the following U-Boot parameters to uboot-env.txt located at */buildroot_2024/board/nuvoton/ma35xx/uboot-env.txt.*

   These parameters enable **swupdate** to apply the updated configuration during the rootfs update. After system reboot, the bootloader will automatically switch to the newly updated root filesystem to complete the boot process.

   This *uboot-env.txt* also includes an automatic rootfs A/B failover mechanism. During normal boot, once the kernel boots successfully, rootfs_active_count is reset to 3. If the system fails to boot successfully three consecutive times, it will automatically switch to the partition B.

   The same mechanism also applies to the partition B. After the kernel boots successfully, rootfs_active_count is reset to 3. If three consecutive boot attempts fail, the system will automatically switch back to the partition A.

   The same *uboot-env.txt* can be used for eMMC/SD card, SPI-NAND, and raw NAND devices.

```
bootcmd=run check_boot_count; bootcmd=run set_mmc_part; run distro_bootcmd
bootdelay=3
baudrate=115200
   .
   .
kernelmem=256M
rootfs_active=a
mmc_part=1
mmc_block=mmcblk0p${mmc_part}
set_mmc_part=if test "${rootfs_active}" = "b"; then setenv mmc_part 2; else setenv
mmc_part 1; fi; setenv mmc_block mmcblk0p${mmc_part}
   .
   .
rootfs_active_count=3
check_boot_count=if test "${rootfs_active_count}" = "0"; then echo "Boot failed 3 times!
Rolling back..."; if test "${rootfs_active}" = "b"; then run switch_to_a; else run
switch_to_b; fi; setenv rootfs_active_count 3; saveenv; else if test
"${rootfs_active_count}" = "3"; then setenv rootfs_active_count 2; saveenv; elif test
"${rootfs_active_count}" = "2"; then setenv rootfs_active_count 1; saveenv; elif test
"${rootfs_active_count}" = "1"; then setenv rootfs_active_count 0; saveenv; fi; fi
   .
   .
```

5. Update */buildroot_2024/board/nuvoton/ma35xx/post-image.sh*. After the update, when programming via NuWriter, the partition B will be created by default, with the partition A sized at 244 MB and the partition B sized at 1 GB.

6. Add a *fw_env.config* file and place it in */output/target/etc/fw_env.config*. This configuration file is used by fw_printenv and fw_setenv under Linux, allowing them to correctly access the U-Boot environment.

   The following configuration indicates that the U-Boot environment is stored on */dev/mmcblk0*, starting at offset 0x40000, with an environment size of 0x10000.

```
/dev/mmcblk0 0x40000 0x10000
```

7. After completing the modifications, rebuild the kernel and program the image to the target board.

```
$ make arm-trusted-firmware-rebuild uboot-rebuild linux-rebuild
$ make
```

8. After the system successfully boots into the kernel, execute *boot_success.sh* to reset rootfs_active_count to 3. This prevents the A/B boot fallback mechanism from being triggered due to the boot retry count reaching zero.

```
MA35D1# ./boot_success.sh
```

9. Update *sw-description* and *generate_swu.sh* for the eMMC/SD card version.

```
{
  "software": {
    "version": "0.1.0",
    "description": ["Linux kernel and dtb and rootfs update for MA35 eMMC/SDcard"],
    "hardware-compatibility": ["1.0", "1.2", "1.3"],
    "scripts": [
      {
        "filename": "prepare_ab_update.sh",
        "type": "preinstall"
      },
      {
        "filename": "finish_ab_update.sh",
        "type": "postinstall"
      }
    ],
    "images": [
      {
        "version": "5.10",
        "name": "nand-kernel device tree",
        "filename": "Image.dtb",
        "device": "/dev/mmcblk0",
        "type": "raw",
        "offset": 2883584
      },
      {
        "version": "5.10",
        "name": "nand-kernel",
        "filename": "Image",
        "device": "/dev/mmcblk0",
        "type": "raw",
        "offset": 3145728
      },
      {
        "version": "5.10",
        "name": "rootfs_update",
        "filename": "rootfs.ext2",
        "device": "/tmp/update_rootfs",
        "type": "raw"
      }
    ]
  }
}
```

generate_swu.sh

```
# Assigns the value "sw-description Image" to the variable FILES, which contains a space-separated list of filenames to be included in the container
FILES="sw-description Image Image.dtb rootfs.ext2 prepare_ab_update.sh finish_ab_update.sh"
```

10. Place the successfully rebuilt image, *Image.dtb*, *rootfs.ext2*, *prepare_ab_update.sh*, *finish_ab_update.sh*, and *sw-description* into the same folder, and then run *generate_swu.sh*. After completion, the *pack_1.0.swu* image will be generated and shown in Figure 2-2.

```
ubuntu:~$ ./generate_swu.sh
```

```
nuvoton@ubuntu:~/                                                    /eMMC_SDcard$ ./generate_swu.s
sw-description
Image
Image.dtb
rootfs.ext2
prepare_ab_update.sh
finish_ab_update.sh
519199 blocks
nuvoton@ubuntu:~/                                                    /eMMC_SDcard$ ls -l
total 519228
-rwxrw-rw- 1 nuvoton nuvoton      1401 May 12 02:39 finish_ab_update.sh
-rwxrwxr-x 1 nuvoton nuvoton      1079 May 12 03:21 generate_swu.sh
-rw-r--r-- 1 nuvoton nuvoton  14123520 May 12 03:04 Image
-rwxr-xr-x 1 nuvoton nuvoton     43259 May 12 03:04 Image.dtb
-rw-rw-r-- 1 nuvoton nuvoton 265829888 May 12 05:08 pack_1.0.swu
-rwxrw-rw- 1 nuvoton nuvoton      1618 May 12 02:36 prepare_ab_update.sh
-rw-r--r-- 1 nuvoton nuvoton 251658240 May 12 03:04 rootfs.ext2
-rw-rw-r-- 1 nuvoton nuvoton       962 May 12 03:22 sw-description
-rw-rw-r-- 1 nuvoton nuvoton      1333 May 19  2023 sw-description 2021
nuvoton@ubuntu:~/                                                    /eMMC_SDcard$
```

Figure 2-2 Pack Image Generation

2.2 SPI-NAND Update Method

1. Follow the description in the EC_MA35D1_OTA_Update_by_Linux_Stage_V1.00 to enable the **swupdate** configuration in Buildroot_2024.

2. Refer to the eMMC/SD card **swupdate** configuration and modify */package/swupdate/swupdate.config*.

3. After completing the modifications, please rebuild **swupdate**.

```
$ make swupdate-rebuild
$ make
```

4. Modify the U-Boot configuration (via uboot-menuconfig) to adjust the U-Boot environment storage size to 1 MB. To ensure proper alignment for A/B partition updates, set the rootfs size to 200 MB.

```
$ make uboot-menuconfig

   Command line interface  --->
         Filesystem commands  --->
   mtdparts=spi-nand0:3m(uboot),1m(uboot-env),256k(device-tree),24m(kernel),200m(rootfs)

$ make
```

5. Modify */buildroot_2024/output/build/linux-custom/arch/arm64/boot/dts/nuvoton/ma35d1.dtsi* to update spinand-uboot-env, setting its size to 0x100000 (1 MB). In addition, allocate SPI-NAND space for the partition B (spinand-rootfs-b). In this example, the partition B size is set to 200 MB.

```
flash: flash@0 {
    compatible = "spi-nand";
    reg = <0 0x0 0x2000000>;
    spi-max-frequency = <50000000>;
    #address-cells = <1>;
    #size-cells = <1>;

    uboot@0 {
      label = "spinand-uboot";
      reg = <0x0 0x300000>;
    };
    uboot-env@300000 {
      label = "spinand-uboot-env";
      reg = <0x300000 0x100000>;
    };
    device-tree@400000 {
      label = "spinand-device-tree";
      reg = <0x400000 0x40000>;
    };
    kernel@440000 {
      label = "spinand-kernel";
      reg = <0x440000 0x1800000>;
    };
    rootfs@1c00000 {
      label = "spinand-rootfs";
      reg = <0x1c00000 0xC800000>;
    };
    rootfs-b@E400000 {
      label = "spinand-rootfs-b";
      reg = <0xE400000 0xC800000>;
    };

  };
```

6. Add the following U-Boot parameters to the *uboot-env.txt* file. These parameters allow **swupdate** to automatically apply the updated settings during the rootfs update process. On the next reboot, the system will boot from the newly updated root filesystem. For the MA35D1-SOM platform, both the NAND driver and SPI-NAND driver will be enabled during boot. In this example, spinand_ubiblock is set to 9, and spinand_uboot_para_blk is set to 6.

   If the system boots without the NAND driver enabled and only the SPI-NAND driver is used, spinand_ubiblock should be set to 4 and spinand_uboot_para_blk should be set to 1. The file is located at */buildroot_2024/board/nuvoton/ma35xx/uboot-env.txt*.

   This *uboot-env.txt* also includes an automatic rootfs A/B failover mechanism. During normal boot, once the kernel boots successfully, rootfs_active_count is reset to 3. If the system fails to boot successfully three consecutive times, it will automatically switch to the partition B.

The same mechanism also applies to the partition B. After the kernel boots successfully, rootfs_active_count is reset to 3. If three consecutive boot attempts fail, the system will automatically switch back to the partition A.

The same *uboot-env.txt* can be used for eMMC/SD card, SPI-NAND, and raw NAND devices.

```
bootcmd=run check_boot_count; run distro_bootcmd
bootdelay=3
baudrate=115200

   .
   .
rootfs_active=a
spinand_ubiblock=9
nand_ubiblock=4
spinand_uboot_para_blk=6
switch_to_a=setenv rootfs_active "a"; setenv nand_ubiblock 4; setenv spinand_ubiblock 9
switch_to_b=setenv rootfs_active "b"; setenv nand_ubiblock 5; setenv spinand_ubiblock 10
rootfs_active_count=3
check_boot_count=if test "${rootfs_active_count}" = "0"; then echo "Boot failed 3 times!
Rolling back..."; if test "${rootfs_active}" = "b"; then run switch_to_a; else run
switch_to_b; fi; setenv rootfs_active_count 3; saveenv; else if test
"${rootfs_active_count}" = "3"; then setenv rootfs_active_count 2; saveenv; elif test
"${rootfs_active_count}" = "2"; then setenv rootfs_active_count 1; saveenv; elif test
"${rootfs_active_count}" = "1"; then setenv rootfs_active_count 0; saveenv; fi; fi   .
   .
   .
```

7. Add a *fw_env.config* file and place it in */output/target/etc/fw_env.config*. This configuration file is used by fw_printenv and fw_setenv under Linux, allowing them to correctly access the U-Boot environment.

   The following configuration indicates that the U-Boot environment is stored in */dev/ubi1_0*, starting at offset 0x00000, with an environment size of 0x10000 and an erase size of 0x1F000.

```
/dev/ubi1_0    0x00000  0x10000  0x1F000
```

8. After completing the modifications, rebuild the kernel and program the image to the target board.

```
$ make arm-trusted-firmware-rebuild uboot-rebuild linux-rebuild
$ make
```

9. After the system successfully boots into the kernel, execute *boot_success.sh* to reset rootfs_active_count to 3. This prevents the A/B boot fallback mechanism from being triggered due to the boot retry count reaching zero.

```
MA35D1# ./boot_success.sh
```

10. Update *sw-description* and *generate_swu.sh* for the SPI-NAND version.

```
{
  "software": {
```

```
    "version": "0.1.0",
    "description": ["Linux kernel and dtb and rootfs update for MA35 SPI-NAND"],
    "hardware-compatibility": ["1.0", "1.2", "1.3"],
    "scripts": [
      {
        "filename": "prepare_ab_update.sh",
        "type": "preinstall"
      },
      {
        "filename": "finish_ab_update.sh",
        "type": "postinstall"
      }
    ],
    "images": [
      {
        "version": "5.10",
        "name": "spinand-kernel device tree",
        "filename": "Image.dtb",
        "device": "/dev/mtdblock2",
        "type": "raw"
      },
      {
        "version": "5.10",
        "name": "spinand-kernel",
        "filename": "Image",
        "device": "/dev/mtdblock3",
        "type": "raw"
      },
      {
        "version": "5.10",
        "name": "rootfs_update",
        "filename": "rootfs.ubi",
        "device": "/tmp/update_rootfs",
        "type": "raw"
      }
    ]
  }
}
```

generate_swu.sh

```
# Assigns the value "sw-description Image" to the variable FILES, which contains a space-separated list of filenames to be included in the container
FILES="sw-description Image Image.dtb rootfs.ubi prepare_ab_update.sh finish_ab_update.sh"
```

11. Place the successfully rebuilt image, *Image.dtb*, *rootfs.ubi*, *prepare_ab_update.sh*, *finish_ab_update.sh*, and sw-description in the same directory, then execute *generate_swu.sh*. After execution, the *pack_1.0.swu* file will be generated and shown in Figure 2-3.

```
ubuntu:~$ ./generate_swu.sh
```

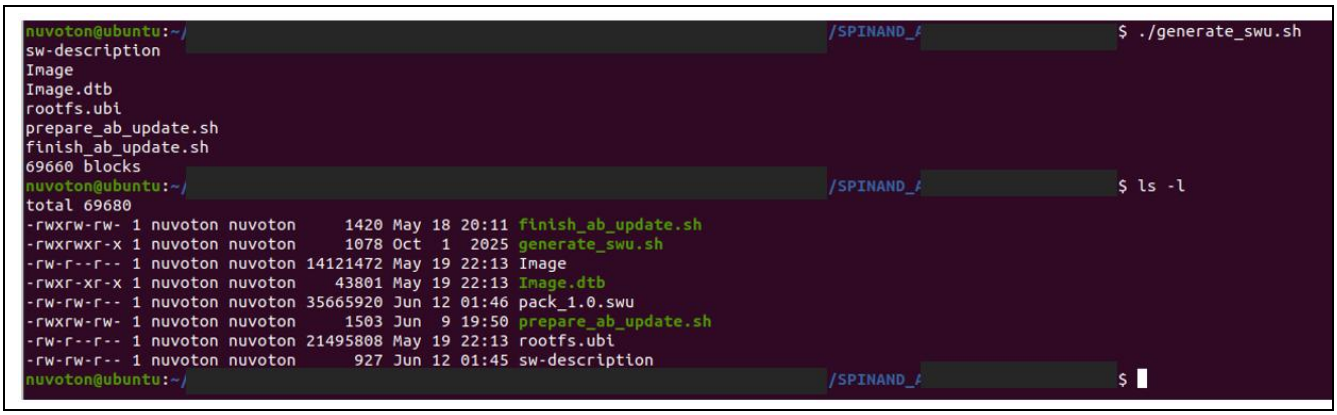


Figure 2-3 Pack Image Generation

2.3 NAND Update Method

1. Follow the description in the EC_MA35D1_OTA_Update_by_Linux_Stage_V1.00 to enable the **swupdate** configuration in Buildroot_2024.

2. Refer to the eMMC/SD card **swupdate** configuration and modify */package/swupdate/swupdate.config*.

3. After completing the modifications, rebuild **swupdate**.

```
$ make swupdate-rebuild
$ make
```

4. Modify */buildroot_2024/output/build/linux-custom/arch/arm64/boot/dts/nuvoton/ma35d1.dtsi* to allocate NAND storage space for the rootfs partition B (nand-rootfs-b). In this example, the partition B size is set to 200 MB.

```
  partitions {
      compatible = "fixed-partitions";
      #address-cells = <1>;
      #size-cells = <1>;

      uboot@0 {
        label = "nand-uboot";
        reg = <0x0000000 0x300000>;
        read-only;
      };
      uboot-env@300000 {
        label = "nand-uboot-env";
        reg = <0x300000 0xC0000>;
      };
      device-tree@3c0000 {
        label = "nand-device-tree";
        reg = <0x3C0000 0x40000>;
      };
      kernel@400000 {
        label = "nand-kernel";
        reg = <0x400000 0x1800000>;
      };
      rootfs@1c00000 {
        label = "nand-rootfs";
        reg = <0x1C00000 0xC800000>;
      };
      rootfs-b@E400000 {
```

```
        label = "nand-rootfs-b";
        reg = <0xE400000 0xC800000>;
      };

    };
```

5. Add the following U-Boot parameters to the *uboot-env.txt* file to allow **swupdate** to automatically apply the updated settings during the rootfs update process. On the next reboot, the system will boot from the newly updated root filesystem. In this example, nand_ubiblock is initially set to 4 and rootfs_active is set to a. The file is located at */buildroot_2024/board/nuvoton/ma35xx/uboot-env.txt*.

   This *uboot-env.txt* also includes an automatic rootfs A/B failover mechanism. During normal boot, once the kernel boots successfully, rootfs_active_count is reset to 3. If the system fails to boot successfully three consecutive times, it will automatically switch to the partition B.

   The same mechanism also applies to the partition B. After the kernel boots successfully, rootfs_active_count is reset to 3. If three consecutive boot attempts fail, the system will automatically switch back to the partition A.

   The same *uboot-env.txt* can be used for eMMC/SD card, SPI-NAND, and raw NAND devices.

```
bootcmd=run check_boot_count; run distro_bootcmd
bootdelay=3
baudrate=115200

   .
   .
rootfs_active=a
nand_ubiblock=4
switch_to_a=setenv rootfs_active "a"; setenv nand_ubiblock 4; setenv spinand_ubiblock 9
switch_to_b=setenv rootfs_active "b"; setenv nand_ubiblock 5; setenv spinand_ubiblock 10
rootfs_active_count=3
   .
   .
   .

check_boot_count=if test "${rootfs_active_count}" = "0"; then echo "Boot failed 3 times!
Rolling back..."; if test "${rootfs_active}" = "b"; then run switch_to_a; else run
switch_to_b; fi; setenv rootfs_active_count 3; saveenv; else if test
"${rootfs_active_count}" = "3"; then setenv rootfs_active_count 2; saveenv; elif test
"${rootfs_active_count}" = "2"; then setenv rootfs_active_count 1; saveenv; elif test
"${rootfs_active_count}" = "1"; then setenv rootfs_active_count 0; saveenv; fi; fi
   .
   .
```

6. Add a *fw_env.config* file and place it in */output/target/etc/fw_env.config*. This configuration file is used by fw_printenv and fw_setenv under Linux, allowing them to correctly access the U-Boot environment.

The following configuration indicates that the U-Boot environment is stored in */dev/mtd1*, starting at offset 0x00000, with an environment size of 0x10000 and an erase size of 0x20000.

```
/dev/mtd1 0x0 0x10000 0x20000
```

7. After completing the modifications, rebuild the kernel and program the image to the target board.

```
$ make arm-trusted-firmware-rebuild uboot-rebuild linux-rebuild
$ make
```

8. After the system successfully boots into the kernel, execute *boot_success.sh* to reset rootfs_active_count to 3. This prevents the A/B boot fallback mechanism from being triggered due to the boot retry count reaching zero.

```
MA35D1# ./boot_success.sh
```

9. Update *sw-description* and *generate_swu.sh* for the NAND version.

```
{
  "software": {
    "version": "0.1.0",
    "description": ["Linux kernel and dtb and rootfs update for MA35 NAND"],
    "hardware-compatibility": ["1.0", "1.2", "1.3"],
    "scripts": [
      {
        "filename": "prepare_ab_update.sh",
        "type": "preinstall"
      },
      {
        "filename": "finish_ab_update.sh",
        "type": "postinstall"
      }
    ],
    "images": [
      {
        "version": "5.10",
        "name": "nand-kernel device tree",
        "filename": "Image.dtb",
        "device": "/dev/mtdblock2",
        "type": "raw"
      },
      {
        "version": "5.10",
        "name": "nand-kernel",
        "filename": "Image",
        "device": "/dev/mtdblock3",
        "type": "raw"
      },
      {
        "version": "5.10",
        "name": "rootfs_update",
        "filename": "rootfs.ubi",
```

```
        "device": "/tmp/update_rootfs",
        "type": "raw"
      }
    ]
  }
}
```

generate_swu.sh

```
# Assigns the value "sw-description Image" to the variable FILES, which contains a space-separated list of filenames to be included in the container
FILES="sw-description Image Image.dtb rootfs.ubi prepare_ab_update.sh finish_ab_update.sh"
```

10. Place the successfully rebuilt image, *Image.dtb*, *rootfs.ubi*, *prepare_ab_update.sh*, *finish_ab_update.sh*, and sw-description in the same directory, then execute *generate_swu.sh*. After execution, the *pack_1.0.swu* file will be generated.

```
ubuntu:~$ ./generate_swu.sh
```

3. Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - Linux-5.10.x
  - Swupdate-2023.12

3.2 Hardware Requirements

- Circuit components
  - NuMaker-HMI-MA35D1 V1.0, as shown in Figure 3-1.

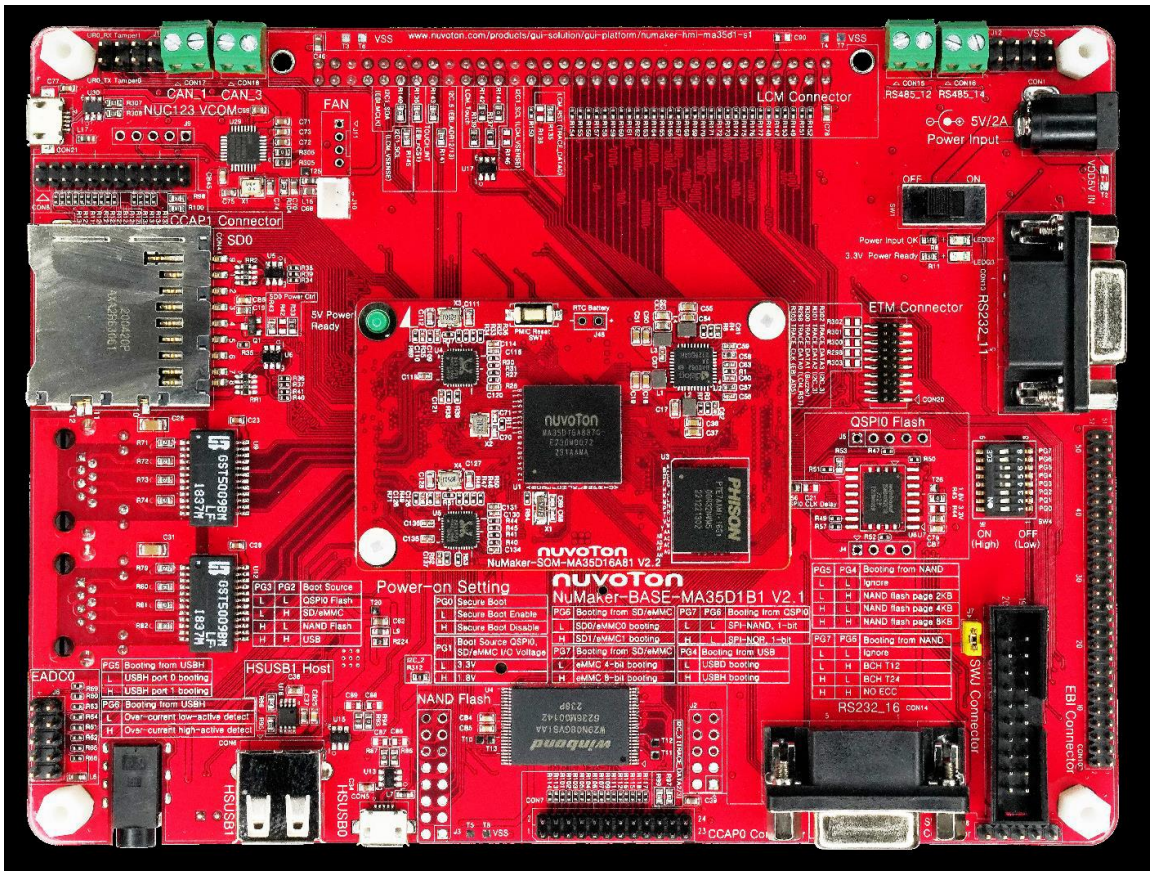


Figure 3-1 NuMaker-HMI-MA35D1 Board

4. Directory Information

The directory structure is shown below.

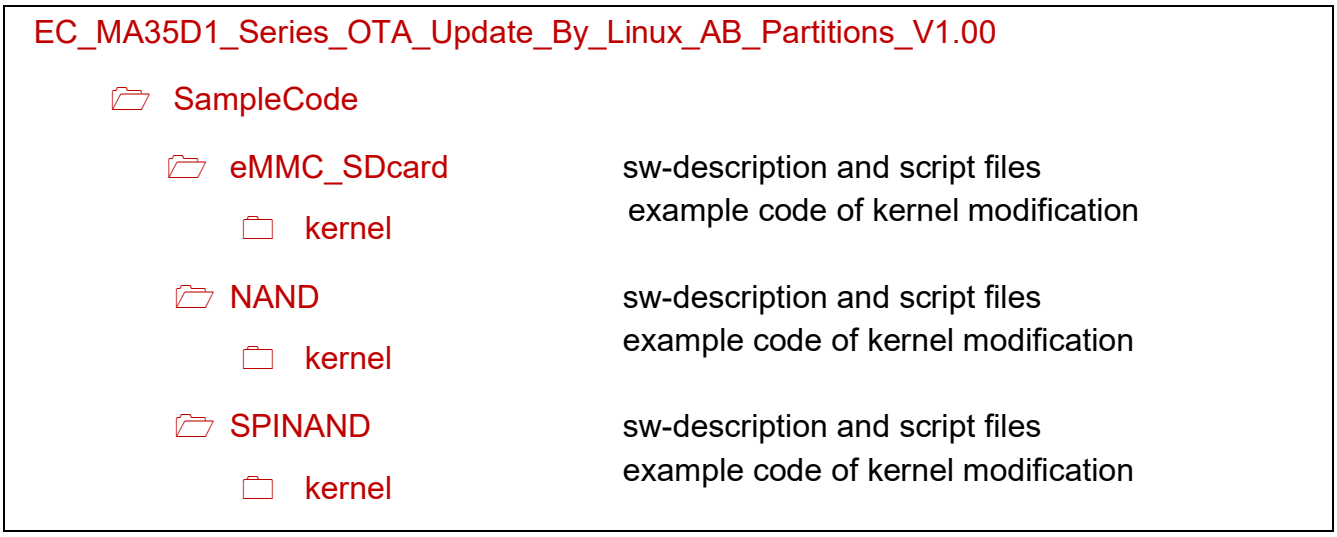


Figure 4-1 Directory Structure

5. Example Code Execution

1. On the MA35D1 device, the system can be configured to obtain an IP address from a router via DHCP (Dynamic Host Configuration Protocol). In this example, the assigned IP address is 192.168.0.175.

```
MA35D1# ifconfig eth0 up
MA35D1# udhcpc -i eth0
udhcpc: started, v1.33.1
udhcpc: sending discover
udhcpc: sending discover
udhcpc: sending select for 192.168.0.175
udhcpc: lease of 192.168.0.175 obtained, lease time 7200
deleting routers
adding dns 192.168.0.1
```

2. Ensure that the PC and the MA35D1 device are connected to the same network, then enter 192.168.0.175:8080.

3. "Click here, or drag and drop a software update image file to this area."

4. Refer to Chapter 2 and select the appropriate update method. Then, run *generate_swu.sh* to generate the *pack_1.0.swu* file. Drag and upload the packaged *pack_1.0.swu* file to complete the automatic A/B partition update. The eMMC/SD card update process is shown in Figure 5-1, and the SPI-NAND/NAND update process is shown in Figure 5-2.

5. After the partition is automatically updated, you can check the boot log to verify whether the update is successful, as shown in Figure 5-3.

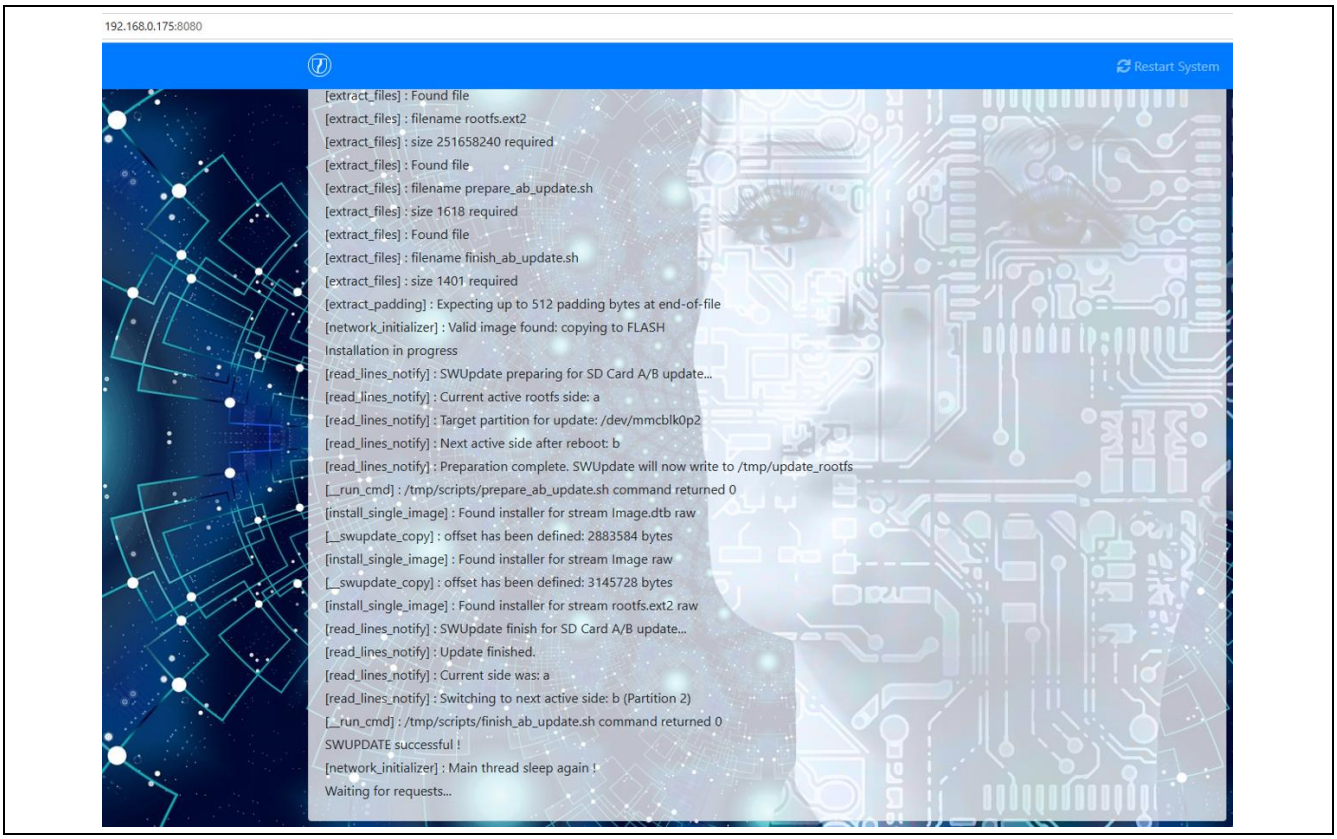


Figure 5-1 eMMC/SD Card rootfs Update to Partition B by swupdate



Figure 5-2 SPI-NAND/NAND rootfs Update to Partition B by swupdate

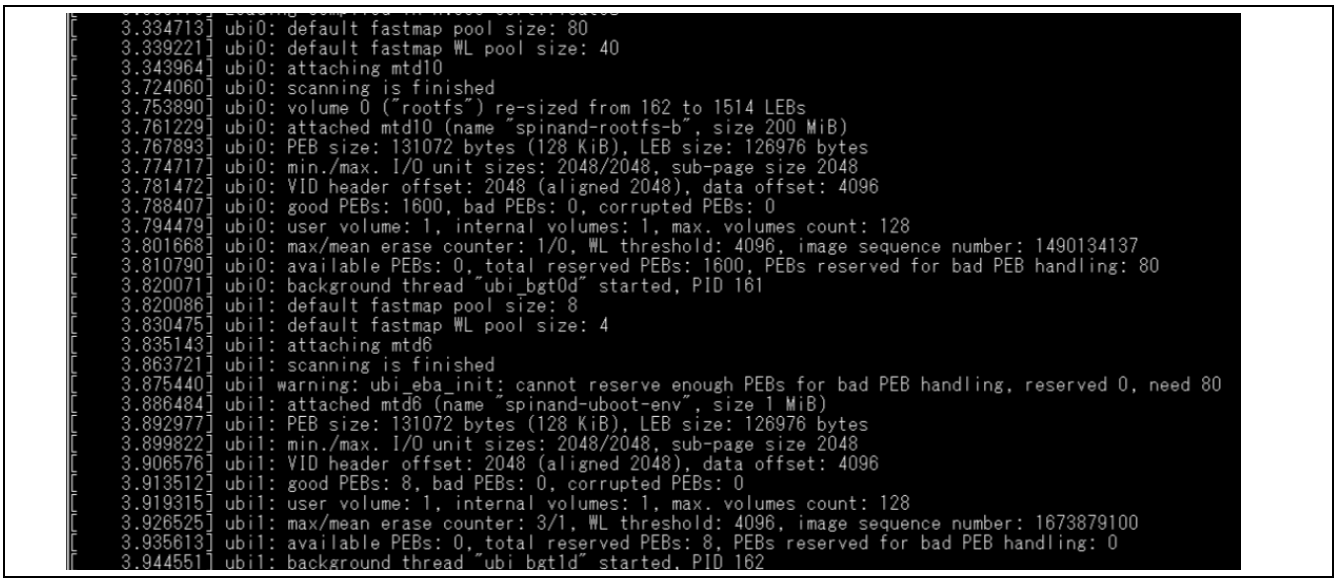


Figure 5-3 Successful Log for SPI-NAND Partition B Update by swupdate

6. Revision History

Date	Revision	Description
2026.06.12	1.00	Initial version.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.