Document Number: AQSUG
Rev. P9.0.0_1.0.0-beta, 11/2018

Android™ Quick Start Guide

Contents

1 Overview

This document guides you through the processes of downloading and running this release package. It only explains how to download and run the default release image with default configuration. For details on using the release package, see the *Android™ User's Guide* (AUG) included in this release package.

2 Hardware Requirements

The hardware requirements for using this release package are as follows:

Supported system-on-chips (SoCs):

- i.MX 8M Mini
- i.MX 8M Quad
- i.MX 8QuadMax
- i.MX 8QuadXPlus

Supported boards:

- EVK board and Platform
- MEK board and Platform

3 Working with the i.MX 8M Mini EVK Board

3.1 Board hardware

The figures below show the different components of the i.MX 8M Mini EVK board.

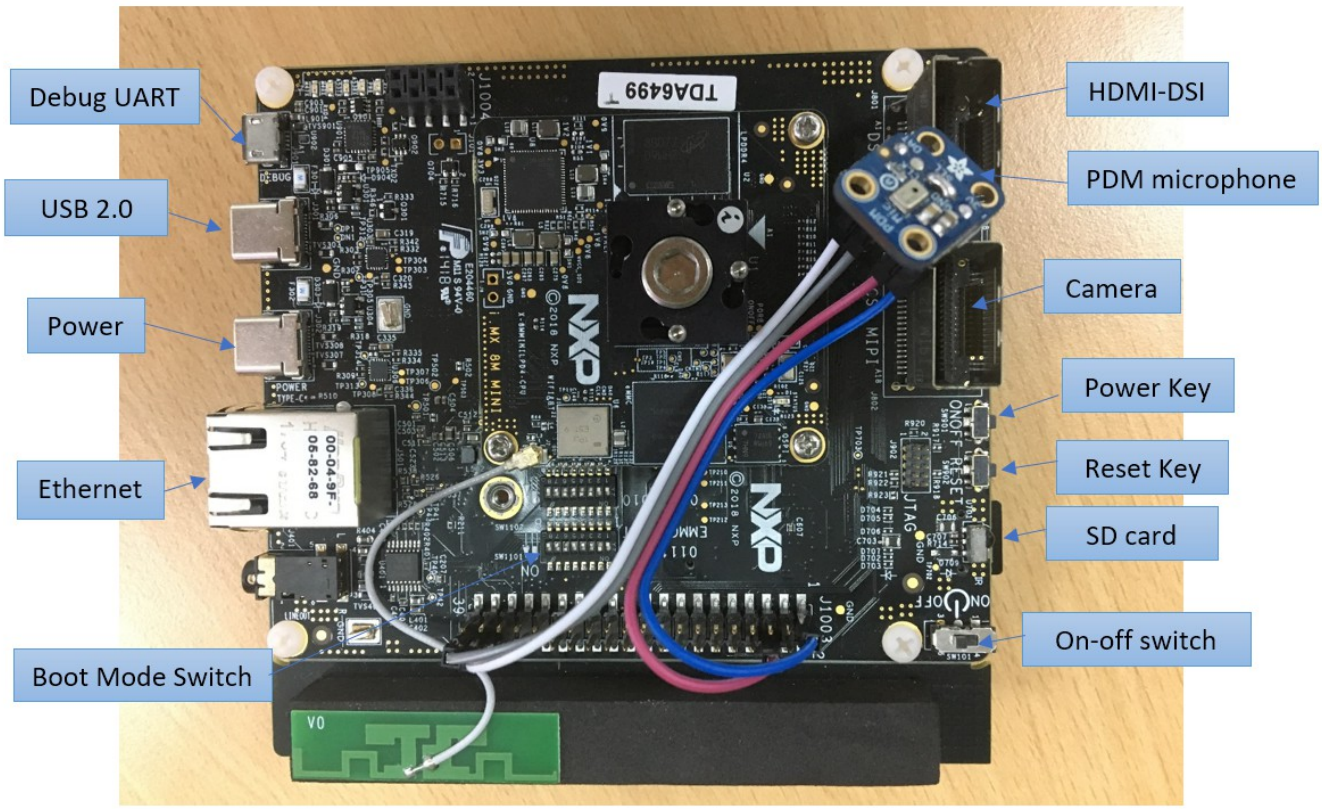

Figure 1. i.MX 8M Mini EVK board

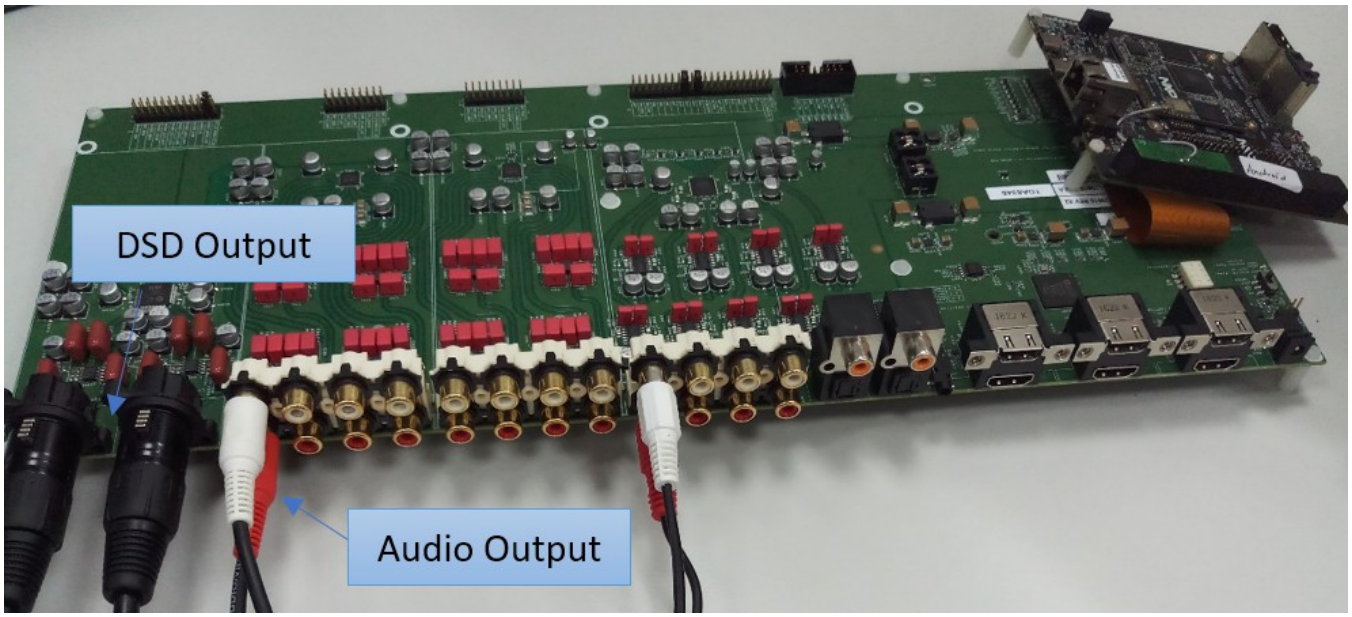

Figure 2. i.MX 8M Mini EVK with audio board

Figure 3. i.MX 8M Mini SAS cable with DSI-to-HDMI adapter

Figure 4. i.MX MIPI panel

Figure 5. i.MX MIPI panel

NOTE

- To test the MIPI-DSI to HDMI display, use the i.MX mini SAS cable to connect the DSI-to-HDMI adapter to the "HDMI DSI" port.
- To test the MIPI panel display, conect the i.MX MIPI panel to the "HDMI DSI" port.
- To test the camera, connect two i.MX MIPI cameras to the "MIPI Camera0" and "MIPI Camera1" ports at the same time.

3.2 Board images

The table below describes the location in the board partitions of the software images in android_p9.0.0_1.0.0-beta_image_8mmevk.tar.gz.

Table 1. Board images

Image name	Download target
/u-boot-imx8mm.imx	33 KB offset of MMC.
/imx8mm_m4_demo.img	5120 KB offset of MMC.
/partition-table.img	0 offset of MMC. If the actually size of the SD card is larger than 13 GB, use the default partition-table.img
/partition-table-7GB.img	0 offset of MMC. If the actually size of the SD card is larger than 7 GB, use this image as partition-table.img.
/partition-table-28GB.img	0 offset of MMC. If the actually size of the SD card is larger than 28 GB, use this image as partition-table.img.
/boot.img	boot_a and boot_b partitions
/vbmeta-imx8mm.img	vbmeta_a and vbmeta_b partitions to support MIPI-to-HDMI output.
/vbmeta-imx8mm-dsd.img	vbmeta_a and vbmeta_b partitions to support MIPI-to-HDMI output and Direct Stream Digital (DSD) playback.
/vbmeta-imx8mm-m4.img	vbmeta_a and vbmeta_b partitions to support MIPI-to-HDMI output and audio playback based on Cortex-M4 FreeRTOS.
/vbmeta-imx8mm-mipi-panel.img	vbmeta_a and vbmeta_b partitions to support MIPI panel output.
/system.img	system_a and system_b partitions.
/vendor.img	vendor_a and vendor_b partitions.
/dtbo-imx8mm.img	dtbo_a and dtbo_b paritions to support MIPI-to-HDMI output
/dtbo-imx8mm-dsd.img	dtbo_a and dtbo_b paritions to support MIPI-to-HDMI output and DSD playback
/dtbo-imx8mm-m4.img	dtbo_a and dtbo_b paritions to support MIPI-to-HDMI output and audio playback based Cortex-M4 FreeRTOS
/dtbo-imx8mm-mipi-panel.img	dtbo_a and dtbo_b paritions to support MIPI panel output

The table below describes the Universal Update Utility (UUU) scripts in android_p9.0.0_1.0.0-beta_image_8mmevk.tar.gz.They are used with the UUU binary file to download the images above into the board. For detailed information on how to download images with UUU, see Section 3.3 "Board images".

Table 2. UUU scripts

UUU script name	Function
uuu-android-mx8mm-evk-emmc.lst	Used with the UUU binary file to download image files into eMMC. The m4_os partition is not flashed.
uuu-android-mx8mm-evk-sd.lst	Used with the UUU binary file to download image files into the SD card. The m4_os partition is not flashed.
uuu-android-mx8mm-evk-emmc-m4.lst	Used with the UUU binary file to download image files into eMMC. The m4_os partition is flashed.
uuu-android-mx8mm-evk-sd-m4.lst	Used with the UUU binary file to download image files into the SD card. The m4_os partition is flashed.

3.3 Flashing board images

The board image files can be flashed into the target board using UUU.

For the UUU binary file, download it from github: uuu release page on github. You can download the latest version.

- For Linux OS, download the file named "uuu".
- For Windows OS, download the file named "uuu.exe" and "libusb-1.0.dll", which need to be in the same directory.

  You can put these files in a path containing the system environment variable "PATH", and then directly call uuu in cmd or shell terminal.

For detailed information on UUU scripts, see Section 3.2 "Board images".

NOTE

UUU uses the fastboot tool to flash images. Make sure you have fastboot driver software installed on your computer.

Perform the following steps to flash the board images:

1. Download the UUU binary file from github as described above.
2. Change the first two bits of the board's sw1101 to 10 (1-2 bit) to enter serial download mode.
3. Power on the board. Connect the PC with the board using the USB cable on the board's USB 3.0 port.
4. Decompress release_package/android_p9.0.0_1.0.0-beta_image_8mmevk.tar.gz, which contains the image files and UUU scripts. Choose the correct UUU script file as shown in the following table.

Target device and boot storage	UUU script file
i.MX 8M Mini EVK eMMC	uuu-android-mx8mm-evk-emmc.lst
i.MX 8M Mini EVK SD	uuu-android-mx8mm-evk-sd.lst

To test MIPI-DSI to HDMI output and audio playback based on Cortex-M4 FreeRTOS, choose one of the UUU scripts shown in the following table.

Target device and boot storage	UUU script file
i.MX 8M Mini EVK eMMC	uuu-android-mx8mm-evk-emmc-m4.lst
i.MX 8M Mini EVK SD	uuu-android-mx8mm-evk-sd-m4.lst

NOTE

- If your SD card is 16 GB or the on-board eMMC is used as the boot device, to test the MIPI-DSI to HDMI output, you do not need to change the "partition-table.img" part of the UUU script.
- If your SD card is 32 GB, rename partition-table.img to partition-table-28GB.img in the corresponding UUU script.
- If your SD card is 8 GB, rename partition-table.img to partition-table-7GB.img in the corresponding UUU script.
- To test the MIPI-DSI to HDMI output and Direct Stream Digital (DSD) playback, rename boot-imx8mm.img and vbmeta-imx8mm.img to boot-imx8mm-dsd.img and vbmeta-imx8mm-dsd.img in the corresponding UUU script.
- To test the MIPI panel output, rename boot-imx8mm.img and vbmeta-imx8mm.img to boot-imx8mm-mipi-panel.img and vbmeta-imx8mm-mipi-panel.img in the corresponding UUU script.

5. Use UUU and the proper script file to flash image files.

Execute the following command to invoke the UUU binary file and UUU scripts to flash the image files.

- On a Linux system, open the shell terminal, and change the working directory to the directory that contains the UUU binary file. Execute the command below. `${uuu_script_path}` is the file path (including the name of the UUU script) of the UUU script that is used. It can be a relative path or an absolute path.

  ```
  > sudo ./uuu ${uuu_script_path}
  ```
- On a Windows system, open the cmd interface, and change the working directory to the directory that contains the UUU binary file and the DLL file. Execute the command below. `${uuu_script_path}` is the absolute file path (including the name of the UUU script) of the UUU script.

  ```
  > uuu.exe ${uuu_script_path}
  ```

6. Wait for the script file execution to complete. If there are no errors, you will get information on the command window as follows:

   ```
   C:\Users\user_01\tools\uuu>uuu.exe C:\Users\user_01\images\android_p9.0.0_1.0.0-
   beta_image_8mmevk\uuu-android-mx8mm-evk-emmc.lst
   uuu (Universal Update Utility) for nxp imx chips -- libuuu-1.1.30-g9f1b007

   Success 1    Failure 0

   2:2    21/21    [Done              ] FB: done
   ```

   As you can see, it is on the Windows system, and the absolute file path of the UUU script is used.
7. Power off the board.
8. Change the boot device as eMMC or SD card.
   - Change sw1101 to 01110010 and change sw1102 to 00101010 if you want to boot from eMMC.
   - Change sw1101 to 01000110 and change sw1102 to 00110100 if you want to boot from SD card.

3.4 Booting

After downloading the images, reboot the board using the power on/off switch.

3.4.1 Booting with MIPI-to-HDMI or MIPI panel display

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttymxc1,115200 earlycon=ec_imx6q,0x30890000,115200 init=/
init androidboot.console=ttymxc1 consoleblank=0 androidboot.hardware=freescale cma=800M
androidboot.primary_display=imx-drm firmware_class.path=/vendor/firmware
transparent_hugepage=never
U-Boot > saveenv
```

With the settings above, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

3.4.2 Booting with MIPI-DSI to HDMI display and audio playback based on Cortex-M4 FreeRTOS

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttymxc1,115200 earlycon=ec_imx6q,0x30890000,115200 init=/
init androidboot.console=ttymxc1 consoleblank=0 androidboot.hardware=freescale cma=800M
androidboot.primary_display=imx-drm firmware_class.path=/vendor/firmware
transparent_hugepage=never
U-Boot > setenv bootcmd "bootmcu && boota mmc0"   # for SD boot
U-Boot > setenv bootcmd "bootmcu && boota mmc1"   # for emmc boot
U-Boot > saveenv
```

NOTE

To use other boot images, do not add "bootmcu" to "bootcmd". The following command can recover bootcmd:

```
U-Boot > setenv bootcmd "boota mmc0"   # for SD boot
U-Boot > setenv bootcmd "boota mmc1"   # for emmc boot
U-Boot > saveenv
```

With the settings above, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

NOTE

Dtbo image supports DSD playback (boot-imx8mm-dsd.img) and uses Single HDMI Display. They share the same U-Boot environment variables.

3.5 Board reboot

After you have completed download and setup, reboot the board and wait for the Android platform to boot up.

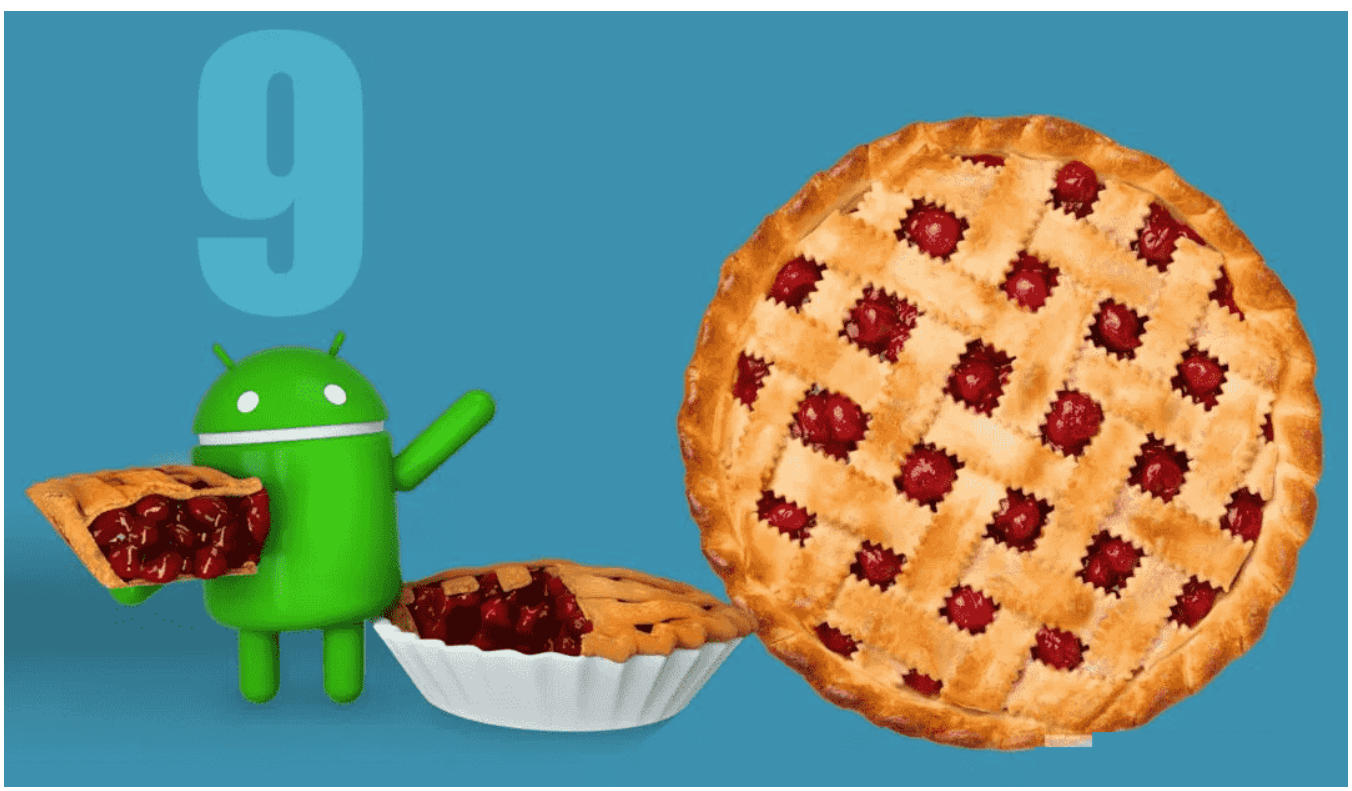

Figure 6. Android Pie image

4 Working with the i.MX 8M Quad EVK Board

4.1 Board hardware

The figures below show the different components of the i.MX 8M Quad EVK board.

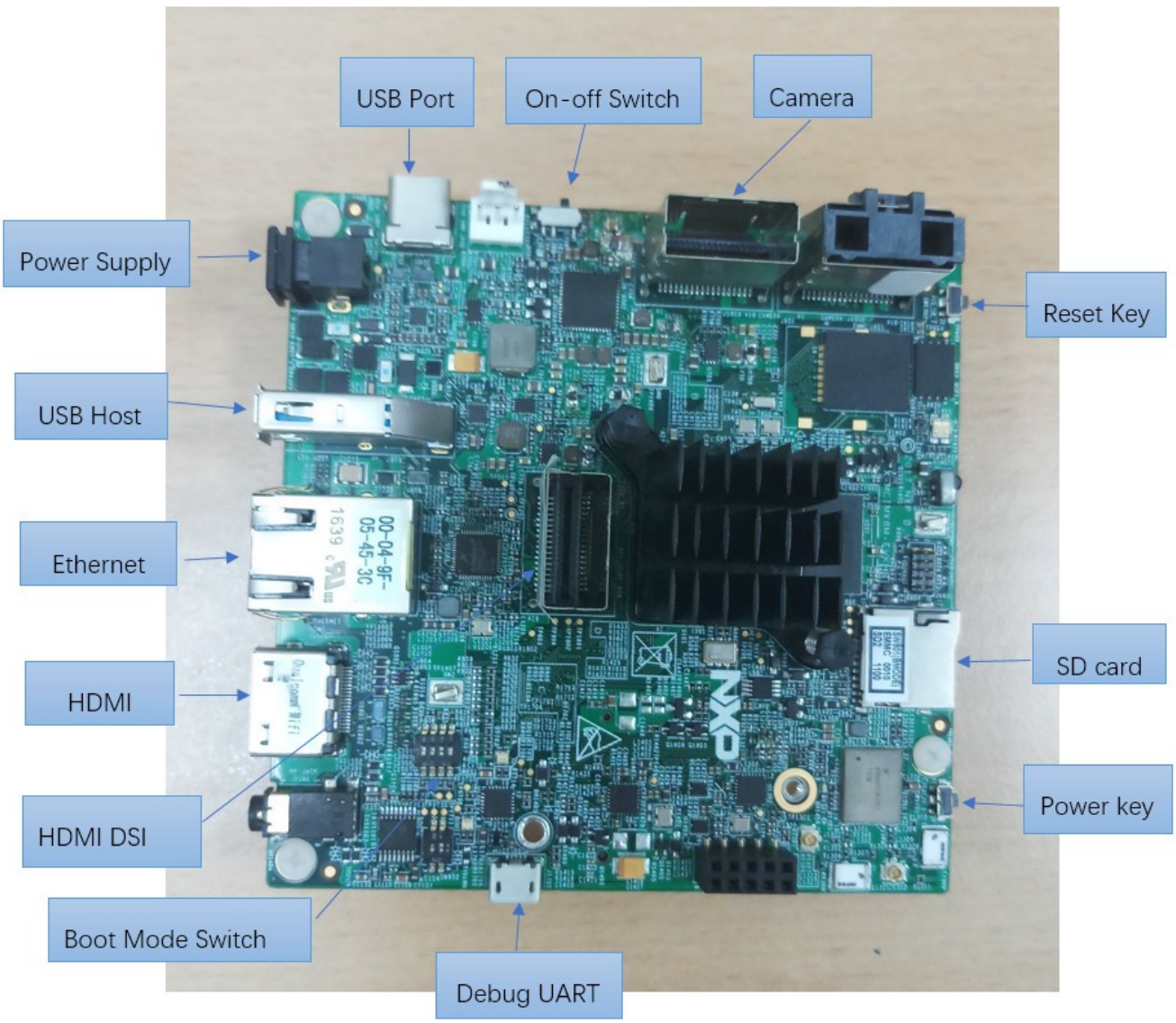

Figure 7. i.MX 8M Quad EVK board

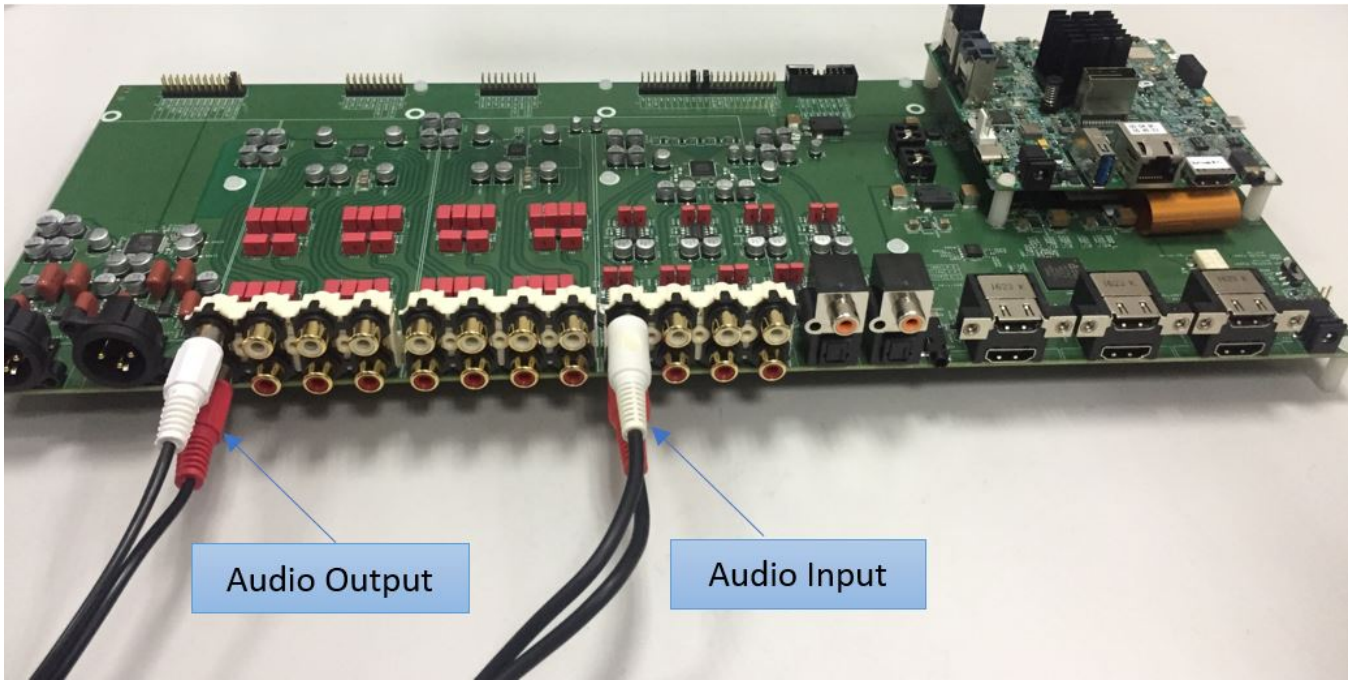

Figure 8. i.MX 8M Quad EVK with audio board

Figure 9. i.MX mini SAS cable with DSI-to-HDMI adapter

Figure 10. i.MX MIPI panel

Figure 11. i.MX MIPI panel

NOTE

- To test the MIPI-DSI to HDMI display, use the i.MX mini SAS cable to connect the DSI-to-HDMI adapter to the "HDMI DSI" port.
- To test the MIPI panel display, conect the i.MX MIPI panel to the "HDMI DSI" port.
- To test the camera, connect the i.MX CSI MIPI Camera to the "Camera" port.

4.2 Board images

The table below describes the location in the board partitions of the software images in android_p9.0.0_1.0.0-beta_image_8mqevk.tar.gz.

Table 3. Board images

Image name	Download target
/u-boot-imx8mq.imx	33 KB offset of SD card
/partition-table.img	0 offset of MMC. If the actually size of the SD card is larger than 13 GB, use the default partition-table.img
/partition-table-7GB.img	0 offset of MMC. If the actually size of the SD card is larger than 7 GB, use this image as partition-table.img
/partition-table-28GB.img	0 offset of MMC. If the actually size of the SD card is larger than 28 GB, use this image as partition-table.img
/boot.img	boot_a and boot_b paritions
/vbmeta-imx8mq-dsd.img	vbmeta_a and vbmeta_b partitions to support i.MX 8M Quad B4 board HDMI output and DSD playback
/vbmeta-imx8mq-mipi.img	vbmeta_a and vbmeta_b partitions to support i.MX 8M Quad B4 board MIPI-to-HDMI output
/vbmeta-imx8mq-dual.img	vbmeta_a and vbmeta_b partitions to support i.MX 8M Quad B4 board HDMI and MIPI-to-HDMI dual output
/vbmeta-imx8mq-mipi-panel.img	vbmeta_a and vbmeta_b partitions to support i.MX 8M Quad B4 board MIPI panel output
/vbmeta-imx8mq-b3.img	vbmeta_a and vbmeta_b partitions to support i.MX 8M Quad B3 board HDMI output
/vbmeta-imx8mq-mipi-b3.img	vbmeta_a and vbmeta_b partitions to support i.MX 8M Quad B3 board MIPI-to-HDMI output
/vbmeta-imx8mq-mipi-panel-b3.img	vbmeta_a and vbmeta_b partitions to support i.MX 8MQuad B3 board MIPI panel output
/system.img	system_a and system_b partitions
/vendor.img	vendor_a and vendor_b partitions
/dtbo-imx8mq.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B4 board HDMI output
/dtbo-imx8mq-dsd.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B4 board HDMI output and DSD playback
/dtbo-imx8mq-mipi.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B4 board MIPI-to-HDMI output
/dtbo-imx8mq-dual.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B4 board HDMI and MIPI-to-HDMI dual output
/dtbo-imx8mq-mipi-panel.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B4 board MIPI panel output
/dtbo-imx8mq-b3.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B3 board HDMI output
/dtbo-imx8mq-mipi-b3.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B3 board MIPI-to-HDMI output
/dtbo-imx8mq-mipi-panel-b3.img	dtbo_a and dtbo_b partitions to support i.MX 8M Quad B3 board MIPI panel output

The table below describes UUU scripts in android_p9.0.0_1.0.0-beta_image_8mqevk.tar.gz. They are used with the UUU binary file to download the images above into the board. For detailed information on how to download images with UUU, see Section 4.3 "Flashing board images".

Table 4. UUU scripts

UUU script name	Function
uuu-android-mx8mq-evk-emmc.lst	Used with the UUU binary file to download image files into eMMC.
uuu-android-mx8mq-evk-sd.lst	Used with the UUU binary file to download image files into the SD card.

4.3 Flashing board images

The board image files can be flashed into the target board using UUU.

For the UUU binary file, download it from github: uuu release page on github. You can download the latest version.

- For Linux OS, download the file named "uuu".
- For Windows OS, download the file named "uuu.exe" and "libusb-1.0.dll", which need to be in the same directory.

  You can put these files in a path containing the system environment variable "PATH", and then directly call uuu in cmd or shell terminal.

For detailed information on UUU scripts, see Section 4.2 "Board images".

NOTE

UUU uses the fastboot tool to flash images. Make sure you have fastboot driver software installed on your computer.

Perform the following steps to flash the board images:

1. Download the UUU binary file from github as described above.
2. Change the board's sw802 (boot mode) to 01 (1-2 bit) to enter serial download mode.
3. Power on the board. Connect the PC with the board using the USB cable on the board USB 3.0 port.

NOTE

- There are two USB ports on the i.MX 8M Quad EVK board: USB-to-UART and USB 3.0. The USB-to-UART port is known as debug UART, and the USB 3.0 port is known as USB in the hardware image above. The debug UART can be used to watch the log of the hardware boot processing.
- The SD card must be plugged in after the board is powered on.

4. Decompress release_package/android_p9.0.0_1.0.0-beta_image_8mmevk.tar.gz, which contains the image files and UUU scripts. Choose the correct UUU script file as shown in the following table.

Target device and boot storage	UUU script file
i.MX 8MQuad EVK SD	uuu-android-mx8mq-evk-sd.lst
i.MX 8MQuad EVK eMMC	uuu-android-mx8mq-evk-emmc.lst

NOTE

- If the SD card is 16 GB or the on-board eMMC is used as the boot device, you do not need to change the "partition-table.img" part of the UUU script.
- If the SD card is 32 GB, change partition-table.img to partition-table-28GB.img in the corresponding UUU script.
- If the SD card is 8 GB, change partition-table.img to partition-table-7GB.img in the corresponding UUU script.

- To test the feature on the i.MX 8M Quad B4 board:
  - To test the HDMI output, you do not need to change the "vbmeta-imx8mq.img and dtbo-imx8mq.img" part of the UUU script.
  - To test the MIPI-to-HDMI output, change vbmeta-imx8mq.img and dtbo-imx8mq.img to vbmeta-imx8mq-mipi.img and dtbo-imx8mq-mipi.img in the corresponding UUU script.
  - To test the MIPI panel output, change vbmeta-imx8mq.img and dtbo-imx8mq.img to vbmeta-imx8mq-mipi-panel.img and dtbo-imx8mq-mipi-panel.img in the corresponding UUU script.
  - To test the HDMI and MIPI-to-HDMI dual output, change vbmeta-imx8mq.img and dtbo-imx8mq.img to vbmeta-imx8mq-dual.img and dtbo-imx8mq-dual.img in the corresponding UUU script.
  - To test the HDMI output and DSD playback, change vbmeta-imx8mq.img and dtbo-imx8mq.img to vbmeta-imx8mq-dsd.img and dtbo-imx8mq-dsd.img in the corresponding UUU script.
- To test the feature on the i.MX 8M Quad B3 board:
  - To test the HDMI output, change vbmeta-imx8mq.img and dtbo-imx8mq.img to vbmeta-imx8mq-b3.img and dtbo-imx8mq-b3.img in the corresponding UUU script.
  - To test the MIPI-to-HDMI output, change vbmeta-imx8mq.img and dtbo-imx8mq.img to vbmeta-imx8mq-mipi-b3.img and dtbo-imx8mq-mipi-b3.img in the corresponding UUU script.
  - To test the MIPI panel output, change vbmeta-imx8mq.img and dtbo-imx8mq.img to vbmeta-imx8mq-mipi-panel-b3.img and dtbo-imx8mq-mipi-panel-b3.img in the corresponding UUU script.

5. Use UUU and the proper script file to flash image files.

   Execute the following command to invoke the UUU binary file and UUU scripts to flash the image files.

   - On a Linux system, open the shell terminal, and change the working directory to the directory that contains the UUU binary file. Execute the command below. `${uuu_script_path}` is the file path (including the name of the UUU script) of the UUU script that is used. It can be a relative path or an absolute path.

     ```
     > sudo ./uuu ${uuu_script_path}
     ```
   - On a Windows system, open the cmd interface, and change the working directory to the directory that contains the UUU binary file and the DLL file. Execute the command below. `${uuu_script_path}` is the absolute file path (including the name of the UUU scripts) of the UUU script.

     ```
     > uuu.exe ${uuu_script_path}
     ```
6. Wait for the script file execution to complete. If there are no errors, the command line interface displays the following information:

   ```
       PS C:\Users\user_01\tools\uuu> uuu.exe  C:\Users\user_01\images
   \android_p9.0.0_1.0.0-beta_image_8mqevk\uuu-android-mx8mq-evk-sd.lst
       uuu (Universal Update Utility) for nxp imx chips -- libuuu-1.1.30-g9f1b007

       Succuess 1    Failure 0

       2:2   19/19   [Done                          ] FB: done
   ```
7. Power off the board.
8. Change the boot device as eMMC or SD card. Change the board's sw802 (boot mode) to 10 (1-2 bit) to exit serial download mode.
   - Change SW801 to switch the board back to 1100 (SD boot mode).
   - Change SW801 to switch the board back to 0010 (eMMC boot mode).

Problems may be encountered when using UUU:

- If the partition table image file being used needs a larger storage capacity than what is really on the board, for example, the default partition table image file needs more than 13 GB storage capacity, and a 8 GB target SD card device is plugged after power-on, the command line interface prompts as follows:

```
2:2    5/19   [write backup GPT image fail ] FB[-t 600000]: flash gpt partition-
table.img
```

- If the data speed of the target device is too slow, the command line interface prompts as follows when flashing system.img. In this situation, you may use an SD card with high data speed, or just modify the UUU script file, changing the number after "-t" to a lager value. Currently, it is 200000, as shown in the following prompt:

```
2:2    9/19   [Bulk read failure           ] FB[-t 200000]: flash system_a system.img
```

4.4 Booting

After downloading the images, boot the board by connecting it to the power supply.

4.4.1 Booting with single display: HDMI display

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttymxc0,115200 earlycon=imxuart,0x30860000,115200 init=/
init androidboot.gui_resolution=1080p androidboot.console=ttymxc0 consoleblank=0
androidboot.hardware=freescale cma=1280M androidboot.primary_display=imx-drm
firmware_class.path=/vendor/firmware androidboot.fbTileSupport=enable
U-Boot > saveenv
```

With above settings, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

4.4.2 Booting with single display: MIPI-to-HDMI display

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttymxc0,115200 earlycon=imxuart,0x30860000,115200 init=/
init androidboot.lcd_density=160 androidboot.console=ttymxc0 consoleblank=0
androidboot.hardware=freescale cma=1280M androidboot.primary_display=mxsfb-drm
firmware_class.path=/vendor/firmware
U-Boot > saveenv
```

With the settings above, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

4.4.3 Booting with dual displays: HDMI and MIPI-to-HDMI displays

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttymxc0,115200 earlycon=imxuart,0x30860000,115200 init=/
init androidboot.gui_resolution=1080p androidboot.console=ttymxc0 consoleblank=0
androidboot.hardware=freescale cma=1280M androidboot.primary_display=imx-drm
firmware_class.path=/vendor/firmware
U-Boot > saveenv
```

With above settings, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

4.4.4 Booting with single display: MIPI panel

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttymxc0,115200 earlycon=imxuart,0x30860000,115200 init=/
init androidboot.console=ttymxc0 consoleblank=0 androidboot.hardware=freescale cma=1280M
androidboot.primary_display=imx-drm firmware_class.path=/vendor/firmware
U-Boot > saveenv
```

With the settings above, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

4.5 Board reboot

After you have completed download and setup, reboot the board and wait for the Android platform to boot up.

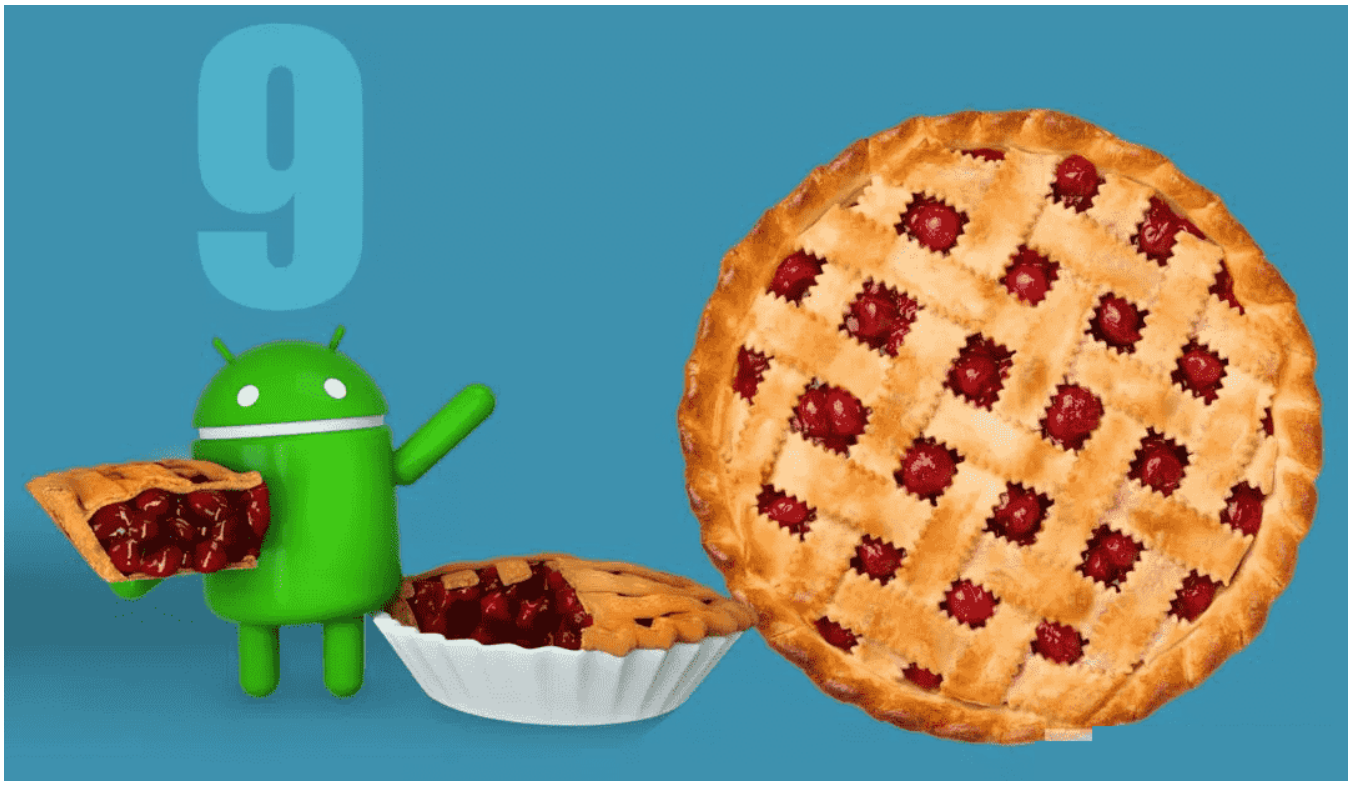

Figure 12. Android Pie image

5 Working with the i.MX 8QuadMax MEK Board

5.1 Board hardware

The figures below show the different components of the i.MX 8QuadMax MEK board.

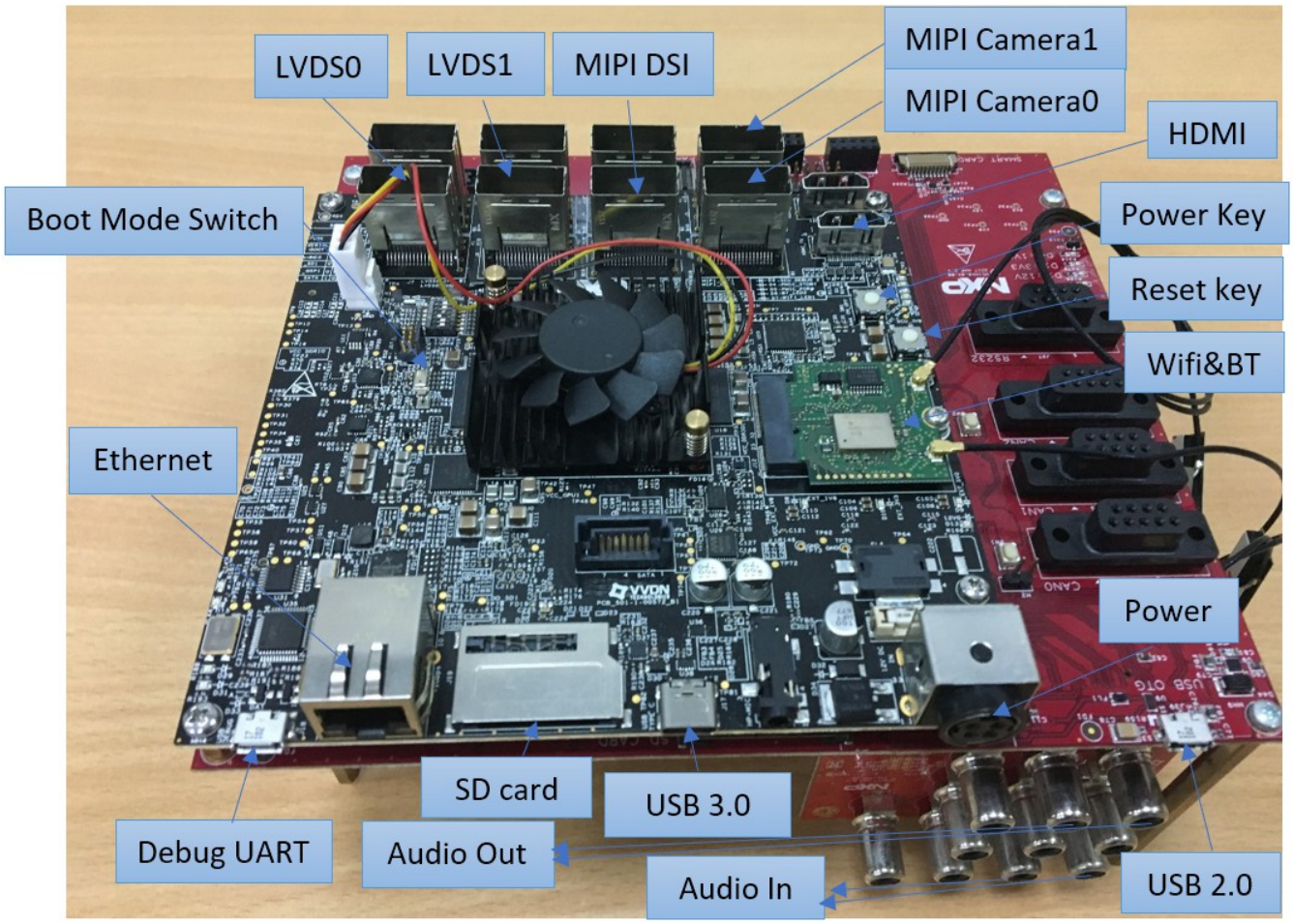

Figure 13. i.MX 8QuadMax MEK board

Figure 14. i.MX mini SAS cable with DSI-to-HDMI adapter

Figure 15. i.MX mini SAS cable with LVDS-to-HDMI adapter

Figure 16. i.MX MIPI camera

NOTE

- To test the MIPI-DSI to HDMI display, use the i.MX mini SAS cable to connect the DSI-to-HDMI adapter to the "HDMI DSI" port.
- To test the LVDS-to-HDMI display, use the i.MX mini SAS cable to connect the LVDS-to-HDMI adapter to the "LVDS0/LVDS1" port.
- To test the camera, connect two i.MX MIPI cameras to the "MIPI Camera0" and "MIPI Camera1" ports.

5.2 Board images

The table below describes the location in the board partitions of the software images in android_p9.0.0_1.0.0-beta_image_8qmek.tar.gz.

Table 5. Board images

Image name	Download target
/u-boot-imx8mq.imx	0 KB offset of eMMC and 32 KB offset of SD card.
/uuu-u-boot-imx8qm.imx	Bootloader used by UUU for the i.MX 8QuadMax MEK board. It is not flashed to MMC.
/partition-table.img	Program to the first 17 KB, and then back up to the last 17 KB of the boot storage. GPT table image for 16 GB boot storage.
/partition-table-7GB.img	Program to the first 17 KB, and then back up to last 17 KB of the boot storage. GPT table image for 8 GB boot storage.
/partition-table-28GB.img	Program to first 17 KB, and then back up to last 17 KB of the boot storage. GPT table image for 32 GB boot storage.
/boot.img	boot_a and boot_b paritions
/vbmeta-imx8qm.img	vbmeta_a and vbmeta_b partitions to support LVDS-to-HDMI/MIPI-to-HDMI display.
/vbmeta-imx8qm-hdmi.img	vbmeta_a and vbmeta_b partitions to support physical HDMI display.
/system.img	system_a and system_b partitions
/vendor.img	vendor_a and vendor_b partitions
/dtbo-imx8qm.img	dtbo_a and dtbo_b partitions to support LVDS-to-HDMI/ MIPI-to-HDMI display.
/dtbo-imx8qm-hdmi.img	dtbo_a and dtbo_b partitions to support physical HDMI display.

The table below describes UUU scripts in android_p9.0.0_1.0.0-beta_image_8qmek.tar.gz. They are used with the UUU binary file to download the images above into the board. For detailed information on how to download images with UUU, see Section 5.3 "Flashing board images".

Table 6. UUU scripts

UUU script name	Function
uuu-android-mx8qm-mek-emmc.lst	Used with the UUU binary file to download image files into eMMC.
uuu-android-mx8qm-mek-sd.lst	Used with the UUU binary file to download image files into the SD card.

5.3 Flashing board images

The board image files can be flashed into the target board using UUU.

For the UUU binary file, download it from github: uuu release page on github. You can download the latest version.

- For Linux OS, download the file named "uuu".
- For Windows OS, download the file named "uuu.exe" and "libusb-1.0.dll", which need to be in the same directory.

  You can put these files in a path containing the system environment variable "PATH", and then directly call uuu in cmd or shell terminal.

For detailed information on UUU scripts, see Section 5.2 "Board images".

NOTE

UUU uses the fastboot tool to flash images. Make sure you have fastboot driver software installed on your computer.

Perform the following steps to flash the board images:

1. Download the UUU binary file from github as described above.
2. Change the board's SW2 (boot mode) to 001000 (1-6 bit) to enter serial download mode.
3. Power on the board. Connect the PC with the board using the USB cable on the board USB 3.0 type-C port.

NOTE

- There are three USB ports on the i.MX 8QuadMax MEK board: USB-to-UART, USB 2.0, and USB 3.0.
- The USB-to-UART port is known as debug UART, and can be used to watch the log of the hardware boot processing.
- USB 2.0 is USB Host and USB 3.0 is USB OTG.

4. Decompress release_package/android_p9.0.0_1.0.0-beta_image_8qmek.tar.gz, which contains the image files and UUU scripts. Choose the correct UUU script file as shown in the following table.

Target device and boot storage	UUU script file
i.MX 8QuadMax MEK SD	uuu-android-mx8qm-mek-sd.lst
i.MX 8QuadMax MEK eMMC	uuu-android-mx8qm-mek-emmc.lst

NOTE

- If your SD card is 16 GB or the on-board eMMC is used as the boot device, you do not need to change the "partition-table.img" part of the UUU script.
- If your SD card is 32 GB, change partition-table.img to partition-table-28GB.img in the corresponding UUU script.
- If your SD card is 8 GB, change partition-table.img to partition-table-7GB.img in the corresponding UUU script.

- To test the LVDS-to-HDMI/MIPI-to-HDMI display, you do not need to change the "dtbo-imx8qm.img and vbmeta-imx8qm.img" part of the UUU script.
- To test the physical HDMI display, change dtbo-imx8qm.img and vbmeta-imx8qm.img to dtbo-imx8qm-hdmi.img and vbmeta-imx8qm-hdmi.img in the corresponding UUU script.

5. Use UUU and the proper script file to flash image files.

   Execute the following command to invoke the UUU binary file and UUU scripts to flash the image files.

   - On a Linux system, open the shell terminal, and change the working directory to the directory that contains the UUU binary file. Execute the command below. `${uuu_script_path}` is the file path (including the name of the UUU script) of the UUU script that is used. It can be a relative path or an absolute path.

     ```
     > sudo ./uuu ${uuu_script_path}
     ```
   - On a Windows system, open the cmd interface, and change the working directory to the directory that contains the UUU binary file and the DLL file. Execute the command below. `${uuu_script_path}` is the absolute file path (including the name of the UUU scripts) of the UUU script.

     ```
     > uuu.exe ${uuu_script_path}
     ```
6. Wait for the script file execution to complete. If there are no errors, you will get information on the command window as follows:

   ```
       PS C:\Users\user_01\tools\uuu> uuu.exe  C:\Users\user_01\images
   \android_p9.0.0_1.0.0-beta_image_8qmek\uuu-android-mx8qm-mek-sd.lst
       uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.1.41-0-g668becf

       Succuess 1    Failure 0

              1/ 0   [                              ]
       2:2   19/19   [Done                           ] FB: done
   ```

   As you can see, it is on the Windows system, and the absolute file path of the UUU script is used. The target device is the SD card.
7. Power off the board.
8. Change the boot device as eMMC or SD card.
   - Change SW2 to switch the board back to 000100 (1-6 bit) to enther eMMC boot mode.
   - Change SW2 to sithch the board back to 001100 (1-6 bit) to enther SD boot mode.

Problems may be encountered when using UUU:

- If the partition table image file being used needs a larger storage capacity than what is really on the board, for example, the default partition table image file needs more than 13 GB storage capacity, and a 8 GB target SD card device is plugged after power-on, the command line interface prompts as follows:

  ```
  2:2    5/19   [write backup GPT image fail ] FB: flash gpt partition-table.img
  ```
- If the data speed of the target device is too slow, the command line interface prompts as follows when flashing system.img. In this situation, you may use an SD card with high data speed, or just modify the UUU script file, changing the number after "-t" to a lager value. Currently, it is 100000, as shown in the following prompt:

  ```
  2:2    14/19   [Bulk read failure           ] FB[-t 100000]: flash system_a system.img
  ```

5.4 Booting

After downloading the images, boot the board by connecting it to the power supply.

5.4.1 Booting with LVDS-to-HDMI/MIPI-to-HDMI display

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttyLP0,115200 earlycon=lpuart32,0x5a060000,115200,115200
init=/init androidboot.console=ttyLP0 consoleblank=0 androidboot.hardware=freescale
androidboot.fbTileSupport=enable cma=800M@0x960M-0xe00M androidboot.primary_display=imx-drm
firmware_class.path=/vendor/firmware
U-Boot > saveenv
```

With above settings, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

5.4.2 Booting with physical HDMI display

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttyLP0,115200 earlycon=lpuart32,0x5a060000,115200,115200
init=/init androidboot.console=ttyLP0 consoleblank=0 androidboot.hardware=freescale
androidboot.fbTileSupport=enable cma=800M@0x960M-0xe00M androidboot.primary_display=imx-drm
androidboot.displaymode=1080p firmware_class.path=/vendor/firmware
U-Boot > saveenv
```

With the settings above, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

5.5 Board reboot

After you have completed download and setup, reboot the board and wait for the Android platform to boot up.

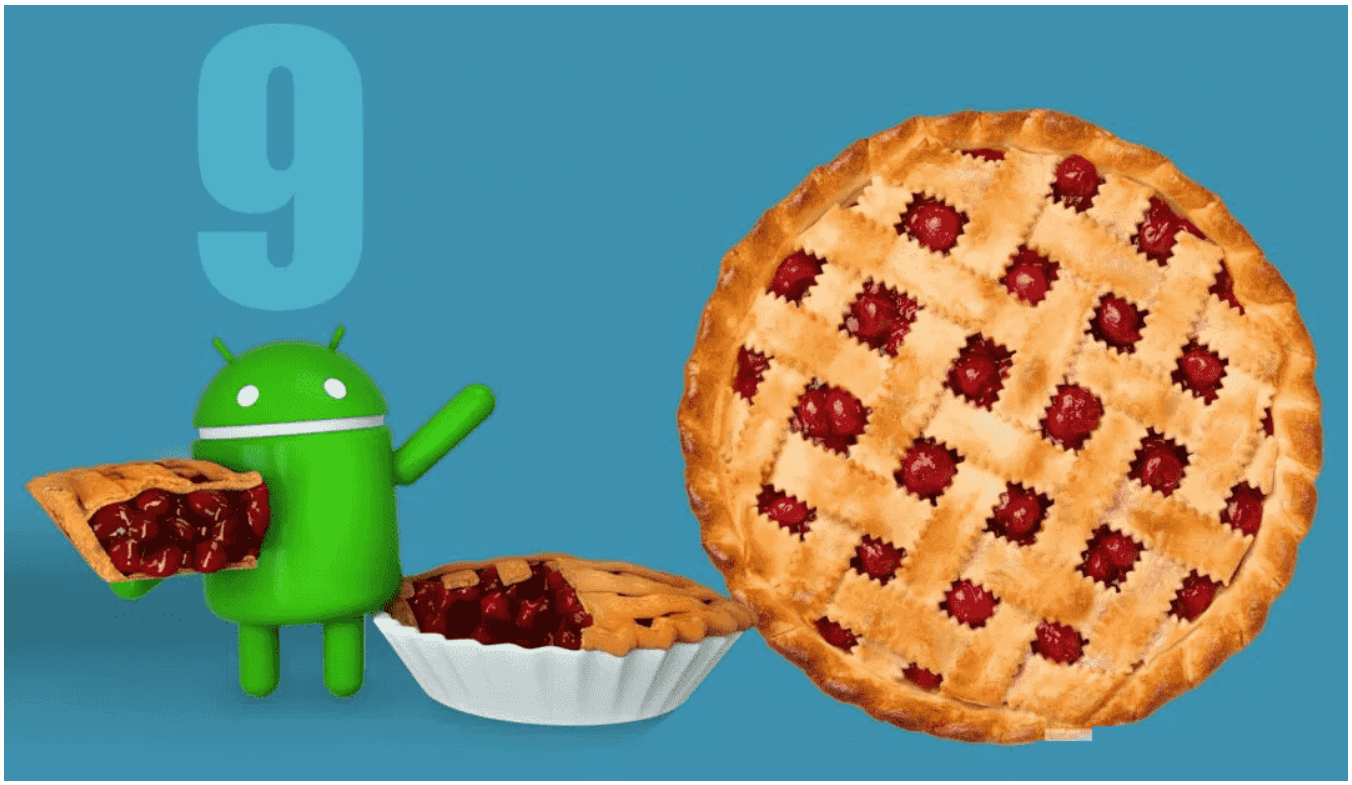

Figure 17. Android Pie image

6 Working with the i.MX 8QuadXPlus MEK Board

6.1 Board hardware

The figures below show the different components of the i.MX 8QuadXPlus MEK board.

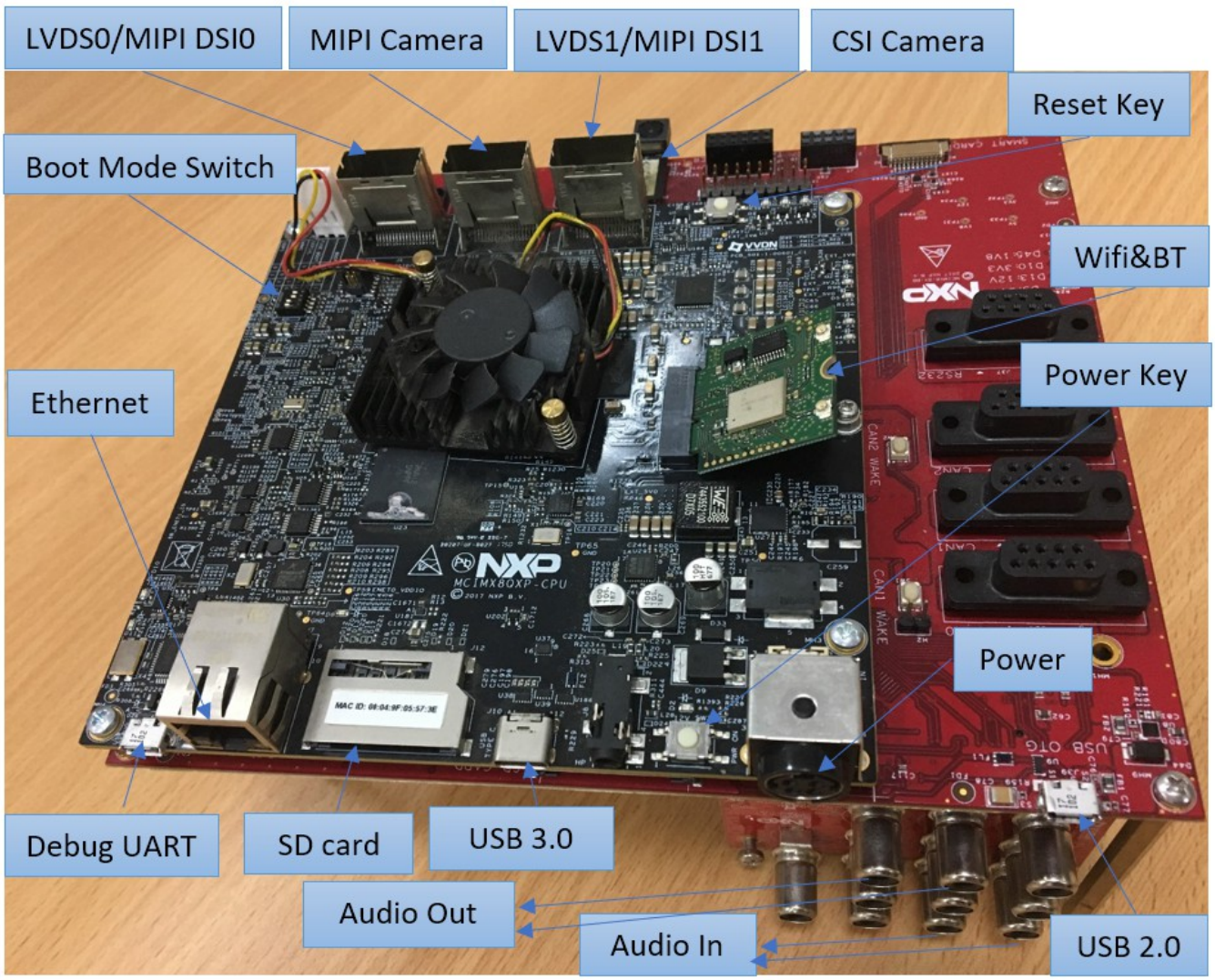

Figure 18. i.MX 8QuadXPlus MEK board

Figure 19. i.MX mini SAS cable with DSI-to-HDMI adapter

Figure 20. i.MX mini SAS cable with LVDS-to-HDMI adapter

Figure 21. i.MX MIPI camera

NOTE

- To test the MIPI-DSI to HDMI display, use the i.MX mini SAS cable to connect the DSI to HDMI adapter to the "MIPI DSI" port.
- To test the LVDS-to-HDMI display, use the i.MX mini SAS cable to connect the LVDS-to-HDMI adapter to the "LVDS0/LVDS1" port.
- To test the camera, connect the i.MX MIPI camera to the "MIPI Camera" port and connect the OV5640 camera to the "CSI Camera" port at the same time.

6.2 Board images

The table below describes the location in the board partitions of the software images in android_p9.0.0_1.0.0-beta_image_8qmek.tar.gz.

Table 7. Board images

Image name	Download target
/u-boot-imx8qxp.imx	32 KB offset of MMC.
/uuu-u-boot-imx8qxp.imx	Bootloader used by UUU for the i.MX 8QuadMax MEK board. It is not flashed to MMC.
/partition-table.img	Program to the first 17 KB, and then back up to the last 17 KB of the boot storage. GPT table image for 16 GB boot storage.
/partition-table-7GB.img	Program to the first 17 KB, and then back up to last 17 KB of the boot storage. GPT table image for 8 GB boot storage.
/partition-table-28GB.img	Program to first 17 KB, and then back up to last 17 KB of the boot storage. GPT table image for 32 GB boot storage.
/boot.img	boot_a and boot_b paritions
/vbmeta-imx8qxp.img	vbmeta_a and vbmeta_b partitions to support single LVDS-to-HDMI/MIPI-to-HDMI or dual LVDS-to-HDMI display with dual-camera support.
/vbmeta-imx8qxp-ov5640mipi.img	vbmeta_a and vbmeta_b partitions to support single LVDS-to-HDMI/MIPI-to-HDMI or dual LVDS-to-HDMI displays with single MIPI camera support.
/system.img	system_a and system_b partitions
/vendor.img	vendor_a and vendor_b partitions
/dtbo-imx8qxp.img	dtbo_a and dtbo_b partitions to support single LVDS-to-HDMI/MIPI-to-HDMI or dual LVDS-to-HDMI displays with dual-camera support.
/dtbo-imx8qxp-ov5640mipi.img	dtbo_a and dtbo_b partitions to support single LVDS-to-HDMI/MIPI-to-HDMI or dual LVDS-to-HDMI displays with single MIPI camera support.

The table below describes UUU scripts in android_p9.0.0_1.0.0-beta_image_8qmek.tar.gz. They are used with the UUU binary file to download the images above into the board. For detailed information on how to download images with UUU, see Section 6.3 "Flashing board images".

Table 8. UUU scripts

UUU script name	Function
uuu-android-mx8qxp-mek-emmc.lst	Used with the UUU binary file to download image files into eMMC.
uuu-android-mx8qxp-mek-sd.lst	Used with the UUU binary file to download image files into the SD card.

6.3 Flashing board images

The board image files can be flashed into the target board using UUU.

For the UUU binary file, download it from github: uuu release page on github. You can download the latest version.

- For Linux OS, download the file named "uuu".
- For Windows OS, download the file named "uuu.exe" and "libusb-1.0.dll", which need to be in the same directory.

  You can put these files in a path containing the system environment variable "PATH", and then directly call uuu in cmd or shell terminal.

For detailed information on UUU scripts, see Section 6.2 "Board images".

NOTE

UUU uses the fastboot tool to flash images. Make sure you have fastboot driver software installed on your computer.

Perform the following steps to flash the board images:

1. Download the UUU binary file from github as described above.
2. Change the board's SW2 (boot mode) to 1000 (1-4 bit) to enter serial download mode.
3. Power on the board. Connect the PC with the board using the USB cable on the board USB 3.0 type-C port.

NOTE

- There are three USB ports on the i.MX 8QuadXPlus MEK board: USB-to-UART, USB 2.0, and USB 3.0.
- The USB-to-UART port is known as debug UART, and can be used to watch the log of the hardware boot processing.
- USB 2.0 is USB Host and USB 3.0 is USB OTG.

4. Decompress release_package/android_p9.0.0_1.0.0-beta_image_8qmek.tar.gz, which contains the image files and UUU scripts. Choose the correct UUU script file as shown in the following table.

Target device and boot storage	UUU script file
i.MX 8QuadXPlus MEK SD	uuu-android-mx8qxp-mek-sd.lst
i.MX 8QuadXPlus MEK eMMC	uuu-android-mx8qxp-mek-emmc.lst

NOTE

- If your SD card is 16 GB or the on-board eMMC is used as the boot device, you do not need to change the "partition-table.img" part of the UUU script.

- If your SD card is 32 GB, change partition-table.img to partition-table-28GB.img in the corresponding UUU script.
- If your SD card is 8 GB, change partition-table.img to partition-table-7GB.img in the corresponding UUU script.
- To test the single LVDS-to-HDMI/MIPI-to-HDMI or dual LVDS-to-HDMI display with dual-camera support, you do not need to change the "dtbo-imx8qxp.img and vbmeta-imx8qxp.img" part of the UUU script.
- To test the single LVDS-to-HDMI/MIPI-to-HDMI or dual LVDS-to-HDMI display with single MIPI camera support, change dtbo-imx8qxp.img and vbmeta-imx8qxp.img to dtbo-imx8qxp-ov5640mipi.img and vbmeta-imx8qxp-ov5640mipi.img.

5. Use UUU and the proper script file to flash image files.

   Execute the following command to invoke the UUU binary file and UUU scripts to flash the image files.

   - On a Linux system, open the shell terminal. Execute the command below. `${uuu_script_path}` is the file path (including the name of the UUU script) of the UUU script that is used. It can be a relative path or an absolute path.

     ```
     > sudo uuu ${uuu_script_path}
     ```
   - On a Windows system, open the cmd interface, and change the working directory to the directory that contains the UUU binary file and the DLL file. Execute the command below. `${uuu_script_path}` is the absolute file path (including the name of the UUU scripts) of the UUU script.

     ```
     > uuu.exe ${uuu_script_path}
     ```
6. Wait for the script file execution to complete. If there are no errors, the command line interface displays the following information:

   ```
       PS C:\Users\user_01\tools\uuu> uuu.exe  C:\Users\user_01\images
   \android_p9.0.0_1.0.0-beta_image_8qmek\uuu-android-mx8qxp-mek-sd.lst
       uuu (Universal Update Utility) for nxp imx chips -- libuuu-1.1.30-g9f1b007

       Succuess 1    Failure 0

       2:2   19/19   [Done                          ] FB: done
   ```

   As you can see, it is on the Windows system, and the absolute file path of the UUU script is used. The target device is the SD card.

7. Power off the board.
8. Change the boot device as eMMC or SD card. Change the board's sw802 (boot mode) to 10 (1-2 bit) to exit serial download mode.
   - Change SW2 to switch the board back to 0100 (1-4 bit) to enter eMMC boot mode
   - Change SW2 to switch the board back to 1100 (1-4 bit) to enter SD boot mode.

Problems may be encountered when using UUU:

- If the partition table image file being used needs a larger storage capacity than what is really on the board, for example, the default partition table image file needs more than 13 GB storage capacity, and a 8 GB target SD card device is plugged after power-on, the command line interface prompts as follows:

  ```
  2:2    5/19   [write backup GPT image fail ] FB: flash gpt partition-table.img
  ```
- If the data speed of the target device is too slow, the command line interface prompts as follows when flashing system.img. In this situation, you may use an SD card with high data speed, or just modify the UUU script file, changing the number after "-t" to a lager value. Currently, it is 100000, as shown in the following prompt:

  ```
  2:2    13/19   [Bulk read failure           ] FB[-t 100000]: flash system_a system.img
  ```

6.4 Booting

After downloading the images, boot the board by connecting it to the power supply.

6.4.1 Booting with single LVDS-to-HDMI/MIPI-to-HDMI or dual LVDS-to-HDMI displays

In the U-Boot prompt, set the U-Boot environment variables as follows:

```
U-Boot > setenv bootargs console=ttyLP0,115200 earlycon=lpuart32,0x5a060000,115200,115200
init=/init androidboot.console=ttyLP0 consoleblank=0 androidboot.hardware=freescale
androidboot.fbTileSupport=enable cma=800M@0x960M-0xe00M androidboot.primary_display=imx-drm
firmware_class.path=/vendor/firmware
U-Boot > saveenv
```

With above settings, the Android platform does not start the shell console. To disable selinux, append "androidboot.selinux=permissive" to the U-Boot's bootargs. Boot environment variables are as follows:

```
U-Boot > setenv append_bootargs androidboot.selinux=permissive
U-Boot > saveenv
```

6.5 Board reboot

After you have completed download and setup, reboot the board and wait for the Android platform to boot up.

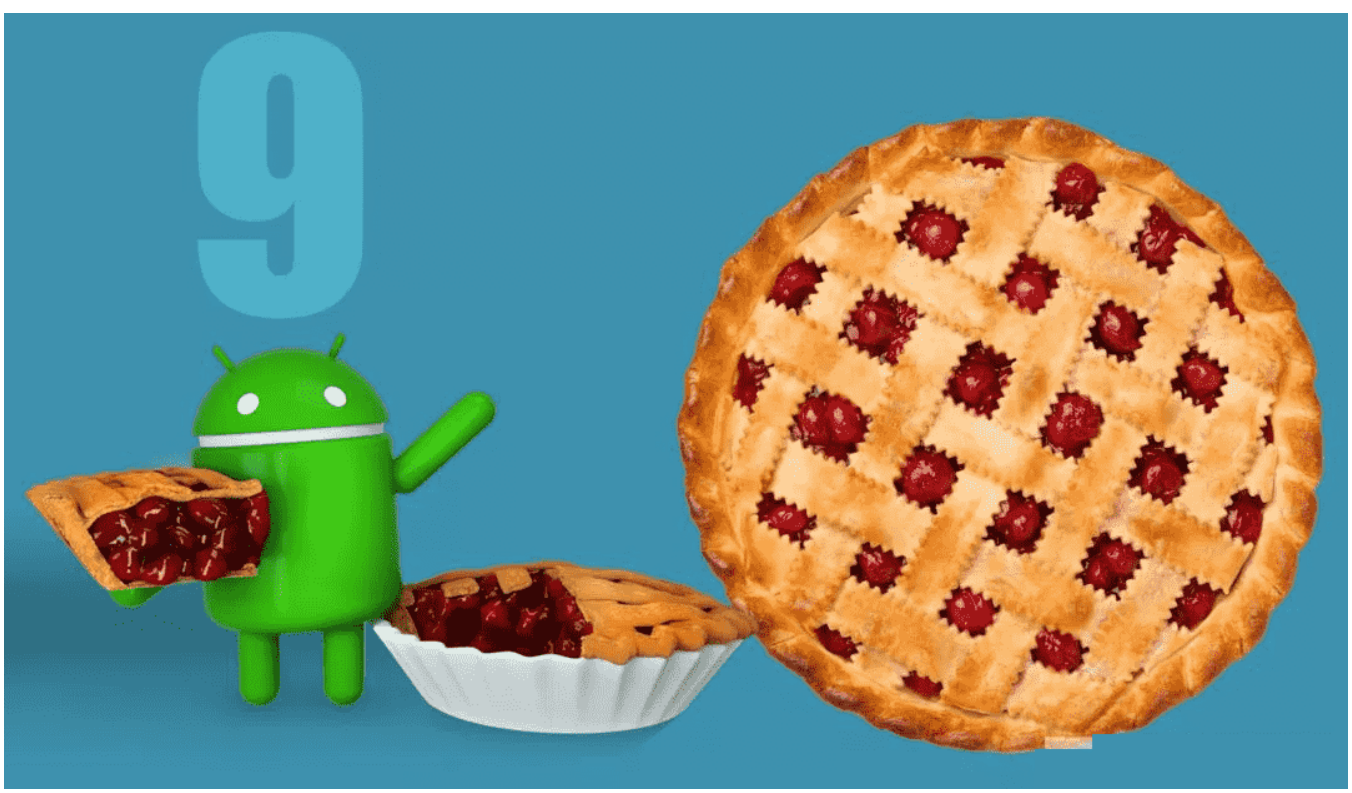

Figure 22. Android Pie image

7 Revision History

Table 9. Revision history

Revision number	Date	Substantive changes
P9.0.0_1.0.0-beta	11/2018	Initial release

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

Document Number AQSUG
Revision P9.0.0_1.0.0-beta, 11/2018