# NXP PN7220 Compliant NFC Controller User Guide

**AN13971**
**PN7220 – Android porting guide**
**Rev. 1.0 — 18 September 2023**
**Application note**

Contents

## PN7220 Compliant NFC Controller

### Document information

| Information | Content |
|---|---|
| Keywords | PN7220, NCI, EMVCo, NFC Forum, Android, NFC |
| Abstract | This document describes how to port PN7220 middleware release to Android. |

**NXP Semiconductors**

**Revision history**

Revision history

| Rev | Date | Description |
|---|---|---|
| v.1.0 | 20230818 | Initial version |

## Introduction

This document provides guidelines for integrating a PN7220 NXP NCI-based NFC controller into an Android platform from a software perspective.

It first explains how to install the required kernel driver, and then describes step-by-step how to customize the AOSP sources to add support for the PN7220 NFC controller. Figure 1 shows the architecture of the entire Android NFC stack.

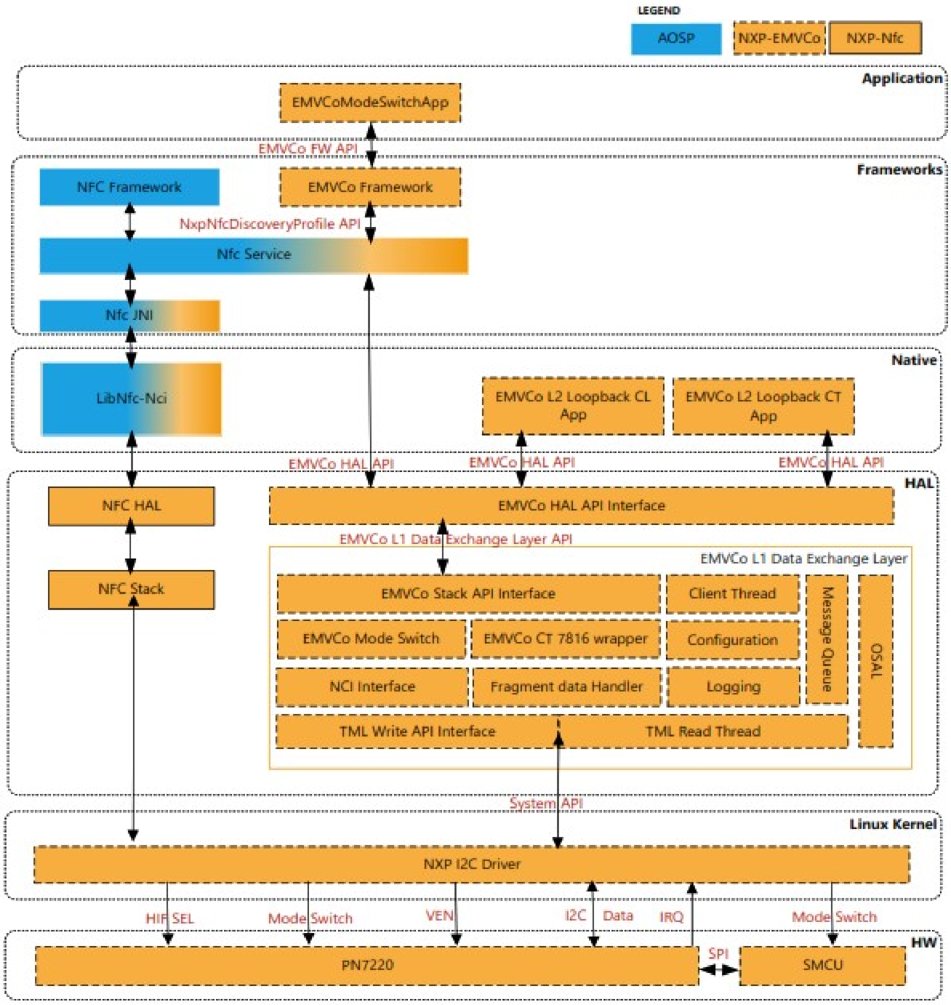


Figure 1. Android NFC stack

PN7220 is separated into single-host and dual-host scenarios. In general, the stack is the same for dual host, we add SMCU.

- The NXP I2C Driver is the kernel module that provides access to the hardware resources of the PN7220.
- The HAL module is the implementation of the specific HW abstraction layer of the controller NXP NFC.
- LibNfc-nci is a native library that provides NFC functionality.
- NFC JNI is a glue code between Java and Native classes.
- The NFC and EMVCo Framework is an application framework module that provides access to NFC and EMVCo functionality.

## Kernel driver

The NFC Android stack uses a nxpnfc kernel driver to communicate with the PN7220. It is available here.

**2.1 Driver details**
The nxpnfc kernel driver offers communication with the PN7220 over an I2C physical interface.
When loaded into the kernel, this driver exposes the interface to PN7220 through the device node named /dev/nxpnfc.

**2.2 Getting the source code**
Clone the PN7220 driver repository into the kernel directory, replacing the existing implementation:
$rm -rf drivers/nfc
$git clone "**https://github.com/NXPNFCLinux/nxpnfc.git**"-b PN7220-Driver drivers/

This ends up with the folder drivers/nfc containing the following files:

- README.md: repository information
- Make file: driver heading make file
- Kcon fig: driver configuration file
- License: driver licensing terms
- nfc subfolder containing:
  – commoc. c: generic driver implementation
  – common. h: generic driver interface definition
  – i2c_drv.c: i2c specific driver implementation
  – i2c_drv.h: i2c specific driver interface definition
  – Makefile: makefile that is included in the makefile of the driver
  – Kbuild => build file
  – Kconfig => driver configuration file

**2.3 Building the driver**
Including the driver into the kernel and making it load during the device boot is done thanks to the devicetree. After updating the device tree definition, the platform-related device tree must be rebuilt. NXP suggests using kernel version 5.10, since in this version complete validation is done.

1. Download the kernel
2. Get the driver source code.
3. Change the device tree definition (specific to the device that we are using).

4. Build the driver.

   a. Through the menuconfig procedure, include the target driver in the build.

After rebuilding the complete kernel, the driver will be included in the kernel image. We must make sure that all new kernel images are copied into the AOSP build.

## AOSP adaptation

NXP provides patches on top of the AOSP code. That means that the user can first get an AOSP code and apply patches from NXP. This section describes how to accomplish this. The current AOSP tag that we are using is [1].

### 3.1 AOSP build

1. We must get the AOSP source code. This we can do with:

   ```
   $ repo init -u https://android.googlesource.com/platform/manifest-b android-13.0.0_r3
   $ repo sync
   ```

   Note: The repo tool must be installed on the system. Follow the instructions [2].
2. When we have source code, we can enter the directory and build it:

   ```
   $cd Android_AROOT
   $source build/envsetup.sh
   $lunch select_target #target is DH we want to use for example: db845c-userdebug $make -j
   ```
3. When AOSP is successfully built, we must get NXP patches. This we can do with:

   ```
   $git clone “https://github.com/NXPNFCLinux/PN7220_Android13.git” vendor/nxp/
   ```
4. At this point, we have all needed to apply patches for PN7220 support. We can apply patches by running the install_NFC.sh script.

   ```
   $chmod +x /vendor/nxp/nfc/install_NFC.sh #sometimes we need to add executable rights to script
   $./vendor/nxp/nfc/install_NFC.sh
   ```

   **Note:** Check the output after running install_NFC.sh. If needed, we must make some changes by hand.
5. We can also add FW binaries:

   ```
   $git clone xxxxxxx
   $cp -r nfc-NXPNFCC_FW/InfraFW/pn7220/64-bit/libpn72xx_fw.so
   AROOT/vendor/nxp/pn7220/firmware/lib64/libpn72xx_fw.so
   $cp -r nfc-NXPNFCC_FW/InfraFW/pn7220/32-bit/libpn72xx_fw.so
   AROOT/vendor/nxp/pn7220/firmware/lib/libpn72xx_fw.so
   ```
6. Adding NFC to build

   In the device.mk makefile (for example, device/brand/platform/device.mk), include specific makefiles:

   ```
   $(call inherit-product, vendor/nxp/nfc/device-nfc.mk)
   ```

   In the BoardConfig.mk makefile (for example, device/brand/platform/BoardConfig.mk), include a specific makefile:

   ```
   -include vendor/nxp/nfc/BoardConfigNfc.mk
   ```
7. Adding DTA application

   ```
   $git clone https://github.com/NXPNFCProject/NXPAndroidDTA.git $git checkout NFC_DTA_v13.02_OpnSrc
   $patch -p1 AROOT_system_nfc-dta.patch
   $ cp -r nfc-dta /system/nfc-dta
   $<AROOT>/system/nfc-dta/$ mm -j
   ```

8. Now we can build AOSP again with all the changes we made:

```
$cd framework/base
$mm
$cd ../..
$cd vendor/nxp/frameworks
$mm #after this one, we should see com.nxp.emvco.jar inside out/target/product/xxxx/system/framwework/
$cd ../../..
$cd hardware/nxp/nfc
$mm
$cd ../../..
$make -j
```

Now, we are able to flash our device host with the Android image that includes NFC features.

### 3.2 Android NFC Apps and Lib on targets

In this subsection, we describe where specific compiled files are pushed. If there is any change, we can replace only that one file. Table 1 shows all locations.

Table 1. Compiled files with device target

| Project location | Compiled Files | Location in target device |
|---|---|---|
| "$ANDROID_ROOT"/packages/apps/Nfc | lib/NfcNci.apk<br>oat/libnfc_nci_jni.so | /system/app/NfcNci/<br>/system/lib64/ |
| "$ANDROID_ROOT"/system/nfc | libnfc_nci.so | /system/lib64/ |
| "$ANDROID_ROOT"/hardware/nxp/nfc | nfc_nci_nxp_pn72xx.so<br>android.hardware.nfc_72xx@1.2-service android.hardware.nfc_72xx@1.2-service.rc<br>android.hardware.nfc@1.0.so<br>android.hardware.nfc@1.1.so<br>android.hardware.nfc@1.2.so | /vendor/lib64<br>/vendor/bin/hw/<br>/vendor/etc/init<br>system/lib64/<br>system/lib64/<br>system/lib64/ |
| "$ANDROID_ROOT"/hardware/nxp/nfc | vendor.nxp.nxpnfc@2.0.so | /system/lib64 |
| "$ANDROID_ROOT"/vendor/nxp/frameworks | com.nxp.emvco.jar | /system/framework<br>/vendor/framework |
| "$ANDROID_ROOT"/hardware/nxp/emvco | emvco_poller.so<br>android.hardware.emvco-service<br>android.hardware.emvco-service.rc<br>android.hardware.emvco-V1-ndk.so<br>android.hardware.emvco-V2-ndk.so | /vendor/lib64<br>/vendor/bin/hw/<br>/vendor/etc/init<br>system/lib64/<br>system/lib64/ |

### 3.3 Patch mapping

Every patch must be applied to a specific location. Table 2 shows the patch name and the location where we must apply it and a block name, which shows us where in the NFC stack (Figure 1) is located.

Table 2. Patch location in NFC Stack

| Block name | Patch name | Location to apply |
|---|---|---|
| NFC HAL and EMVCo HAL | AROOT_hardware_interfaces.patch | hardware/interfaces/ |
| NFC Stack | AROOT_hardware_nxp_nfc.patch | hardware/nxp/nfc/ |
| EMVCo L1 Data Exchange Layer = EMVCo Stack | AROOT_hardware_nxp_emvco.patch | hardware/nxp/emvco/ |
| LibNfc-Nci | AROOT_system_nfc.patch | system/nfc/ |
| NFC JNI | AROOT_packages_apps_Nfc.patch | packages/apps/nfc/ |
| NFC Service | AROOT_packages_apps_Nfc.patch | packages/apps/nfc/ |
| NFC Framework | AROOT_frameworks_base.patch | frameworks/base/ |
| EMVCo Framework | AROOT_vendor_nxp_frameworks.patch | vendor/nxp/frameworks/ |

### 3.4 Flashing images

Images can be found in /out/target/product/{selected_DH}. To flash system images, we must run the following commands (tested on the Dragonboard 845c).

```
$ adb reboot bootloader
$ fastboot flash boot boot_uefi.img
$ fastboot flash vendor_boot vendor_boot.img
$ fastboot flash super super.img
$ fastboot flash userdata userdata.img
$ fastboot format:ext4 metadata $fastboot reboot
```

After the images are flashed, we must perform some MW clean-up by running the following commands (tested on the Dragonboard 845c).

```
$ adb wait-for-device
$ adb root
$ adb wait-for-device
$ adb remount
$ adb shell rm -rf vendor/etc/init/android.hardware.nfc@1.1-service.rc
$ adb shell rm -rf vendor/etc/init/android.hardware.nfc@1.2-service.rc
$ adb push Test_APK/EMVCoAidlHalComplianceTest/EMVCoAidlHalComplianceTestsystem/etc
$ adb shell chmod 0777 /system/etc/EMVCoAidlHalComplianceTest
$ adb push Test_APK/EMVCoAidlHalDesfireTest/EMVCoAidlHalDesfireTest system/etc
$ adb shell chmod 0777 /system/etc/EMVCoAidlHalDesfireTest
$ adb push Test_APK/EMVCoModeSwitchApp/EMVCoModeSwitchApp.apk
system/app/EMVCoModeSwitchApp/EMVCoModeSwitchApp.apk
$ adb shell sync
$ adb reboot
$ adb wait-for-device
```

### 3.5 Config files

In PN7220, we have four different configuration files.

1. libemvco-nxp.conf
2. libnfc-nci.conf
3. libnfc-nxp.conf
4. libnfc-nxp-eeprom.conf

**Note:** Pay attention that the configuration files provided in the example relate to the NFC controller demo board. These files must be adopted according to the targeted integration.
All four files must be pushed to the specific location.

**Table 3. Locations of configuration files**

| **Name of configuration file** | **Location in device** |
|---|---|
| libemvco-nxp.conf | vendor/etc |
| libnfc-nci.conf | vendor/etc |
| libnfc-nxp.conf | system/etc |
| libnfc-nxp-eeprom.conf | vendor/etc |

libnfc-nxp-eeprom.conf

Table 4. libnfc-nxp-eeprom.conf explanation

| **Name** | **Explanation** | **Default value** |
|---|---|---|
| NXP_SYS_CLK_SRC_SEL | System clock source selection configuration | 0x01 |
| NXP_SYS_CLK_FREQ_SEL | System clock frequency selection configuration | 0x08 |
| NXP_ENABLE_DISABLE_STANBY | Option to enable or disable Standby mode | 0x00 |
| NXP_ENABLE_DISABLE_LPCD | Option to enable or disable LPCD. | 0x00 |

Note: If there is no clock configured, either PLL or Xtal, then the MW stack retries in a loop to get the clock and initialize successfully. libnfc-nci.conf

**Table 5. libnfc-nci.conf explanation**

| **Name** | **Explanation** | **Default value** |
|---|---|---|
| APPL_TRACE_LEVEL | Log levels for libnfc-nci | 0xFF |
| PROTOCOL_TRACE_LEVEL | Log levels for libnfc-nci | 0xFFFFFFFF |
| NFC_DEBUG_ENABLED | NFC debug enable setting | 0x01 |
| NFA_STORAGE | Set the target directory for NFC file storage | /data/vendor/nfc |
| HOST_LISTEN_TECH_MASK | Configure host listen feature | 0x07 |
| NCI_HAL_MODULE | NCI HAL Module name | nfc_nci.pn54x |
| POLLING_TECH_MASK | Configuration of the polling technologies | 0x0F |

**Table 5. libnfc-nci.conf explanation…continued**

| Name | Explanation | Default value |
|---|---|---|
| P2P_LISTEN_TECH_MASK | P2P is not supported in PN7220 | 0xC5 |
| PRESERVE_STORAGE | Verify the content of all nonvolatile stores. | 0x01 |
| AID_MATCHING_MODE | Provides different ways to match the AID | 0x03 |
| NFA_MAX_EE_SUPPORTED | Maximum EE supported number | 0x01 |
| OFFHOST_AID_ROUTE_PWR_STATE | Set the OffHost AID supported state | 0x3B |

**libnfc-nxp.conf**

**Table 6. libnfc-nxp.conf explanation**

| Name | Explanation | Default value |
|---|---|---|
| NXPLOG_EXTNS_LOGLEVEL | Configuration for extns logging level | 0x03 |
| NXPLOG_NCIHAL_LOGLEVEL | Configuration for enabling logging of HAL | 0x03 |
| NXPLOG_NCIX_LOGLEVEL | Configuration for enabling logging of NCI TX packets | 0x03 |
| NXPLOG_NCIR_LOGLEVEL | Configuration for enabling logging of NCI RX packets | 0x03 |
| NXPLOG_FWDNLD_LOGLEVEL | Configuration for enabling logging of FW download functionality | 0x03 |
| NXPLOG_TML_LOGLEVEL | Configuration for enabling logging of TM | 0x03 |
| NXP_NFC_DEV_NODE | NFC Device Node name | idev/rixpnfc” |
| MIFARE_READER_ENABLE | Extension for NFC reader for MIFARE en able | Ox01 |
| NXP_FW_TYPE | Firmware file type | Ox01 |
| NXP_I2C_FRAGMENTATION_ ENABLED | Configure 12C fragmentation | 0x00 |
| NFA_PROPRIETARY_CFG | Set Vendor proprietary configuration | {05, FF, FF, 06, 81, 80, 70, FF, FF} |
| NXP_EXT_TVDD_CFG | Set TVDD configuration mode | 0x02 |
| NXP_EXT TVDD_CFG_1 | Configure TVDD settings according to TV DD mode selected | Check config file |
| NXP_EXT_TVDD_CFG_2 | Configure TVDD settings according to TV DD mode selected | Check config file |
| NXP_CORE_CONF | Configure standardized parts of the NFC controller | { 20, 02, 07, 02, 21, 01, 01, 18, 01, 02 } |
| NXP_CORE_CONF_EXTN | Configure proprietary parts of the NFC co ntroller | {00, 00, 00, 00} |
| NXP_SET_CONFIG_ALWAYS | Always send CORE_CONF and CORE_CONF_EXTN (not recommended enabling it.) | Ox00 |
| NXP_RF_CONF_BLK_1 | RF settings | Check config file |
| ISO_DEP_MAX_TRANSCEIVE | Define maximum ISO-DEP extended AP DU length | OxFEFF |
| PRESENCE_CHECK_ALGORITHM | Set the algorithm used for the T4T presen ce check  procedure | 2 |
| NXP_FLASH_CONFIG | Flashing Options Configurations | 0x02 |

libemvco-nxp.conf

**Table 7. libemvco-nxp.conf explanation**

| Name | Explanation | Default value |
|---|---|---|
| NXP LOG EXTNS LOGLEVEL | Configuration for extns logging level | 0x03 |
| NXP LOG NCIHAL LOGLEVEL | Configuration for enabling logging of HAL | 0x03 |
| NXP LOG NCIX LOGLEVEL | Configuration for enabling logging of NCI TX packets | 0x03 |
| NXP LOG NCIR LOGLEVEL | Configuration for enabling logging of NCI RX packets | 0x03 |
| NXP LOG TML LOGLEVEL | Configuration for enabling logging of TML | 0x03 |
| NXP_EMVCO_DEBUG_ENABLED | Enable debugging | 0x03 |
| NXP EMVCO DEV NODE | EMVCo Device Node name | “/dev/nxpnfc” |
| NXP PCD SETTINGS | Configuration to set polling delay between 2 phases | (20, 02, 07, 01, A0, 64, 03, EC, 13, 06) |
| NXP SET CONFIG | Option to set config command for debugging purpose | Check config file |
| NXP GET CONFIG | Option to get config command for debugging purpose | Check config file |

### 3.6 DTA APPLICATION

To allow NFC Forum certification testing, a device test application is provided. It is composed of several components in the different Android layers, which must be built and included in the Android image.
To push the DTA application, we must follow the next steps:

1. Copy all DTA files to one location

```
$cp -rf “out/target/product/hikey960/system/lib64/libosal.so” /DTA-PN7220
$cp -rf “out/target/product/hikey960/system/lib64/libmwif.so” /DTA-PN7220
$cp -rf “out/target/product/hikey960/system/lib64/libdta.so” /DTA-PN7220
$cp -rf “out/target/product/hikey960/system/lib64/libdta_jni.so” /DTA-PN7220
$cp -rf “out/target/product/hikey960/system/app/NxpDTA/NxpDTA.apk” /DTAPN7220
```

2. Push the binaries to the device as bellow

```
adb shell mkdir /system/app/NxpDTA/
adb push libosal.so /system/lib64/
adb push libdta.so /system/lib64/
adb push libdta_jni.so /system/lib64/
adb push libmwif.so /system/lib64/
adb push NxpDTA.apk /system/app/NxpDTA/
```

After flashing the target, the DTA application should then be present in the list of installed applications. Check UG for a detailed description of how to use the application.

## i.MX 8M Nano porting

As an example, we show what porting to the i.MX 8M platform looks like. To get more information, check [3].

### 4.1 Hardware

At the moment, NXP does not provide the adapter board. Check Table 8 to see how to connect boards with wires.

**Table 8. PN7220 to i.MX 8M Nano connections**

| PIN | PN7220 | i.MX 8M NANO |
|---|---|---|
| VEN | J27 – 7 | J003 – 40 |
| IRQ | J27 – 6 | J003 – 37 |
| SDA | J27 – 3 | J003 – 3 |
| SCL | J27 – 2 | J003 – 5 |
| MODE_SWITCH | J43 – 32 | J003 – 38 |
| GND | J27 – 1 | J003 – 39 |

### 4.2 Software

The steps described in this section explain how we can port PN7200 to the i.MX 8M Nano platform. Same steps with a bit of modification, it can be used to port to any other DH that is running Android OS.
Note: In this porting example, we are using 13.0.0_1.0.0_Android_Source.
We can reuse patches related to AOSP code. What must be changed is:

1. Device tree (in i.MX 8M Nano, this is AROOT_vendor_nxp-opensource_imx_kernel.patch)
2. Device-specific patch (in i.MX 8M Nano, this is AROOT_device_nxp.patch)

In AROOT_vendor_nxp-opensource_imx_kernel.patch, we can see how the driver is included and how the device tree is built. This is specific to every device host since we must take care of pin configuration, and this is different between boards. We also must take care of menu configuration.
In AROOT_device_nxp.patch, we are including nfc into the build. In general, we are making sure, that all the services are included correctly, etc. When porting to a specific device host, take this patch as a reference and include all the things inside.
One additional thing we did in porting is located in the device-nfc.mk file:
We need to comment out following lines:

```
# BOARD_SEPOLICY_DIRS += vendor/$(NXP_VENDOR_DIR)/nfc/sepolicy \
# vendor/$(NXP_VENDOR_DIR)/nfc/sepolicy/nfc
```

The reason for this is that we are including sepolicy in the device-specific BoardConfig.mk file. Steps to build images:

```
> Get AOSP code for i.MX8M Nano
> Build AOSP
> Get NXP patches ([5])
> Apply all patches with install_nfc.sh
> cd framework/base
> mm
> cd ../..
> cd vendor/nxp/frameworks
> mm #after this one, we should see com.nxp.emvco.jar inside out/target/product/ imx8mn/system/framwework/
> cd ../../..
> cd hardware/nxp/nfc
> mm
> cd ../../..
> make
> Download images and use uuu tool to flash i.MX8M Nano
```

## Abbreviations

Table 9. Abbreviations

| Acronym | Description |
|---|---|
| APDU | application protocol data unit |
| AOSP | Android open source project |
| DH | device host |
| HAL | hardware abstraction layer |
| FW | firmware |
| I2C | inter-integrated circuit |
| LPCD | lower powerd card detection |
| NCI | NFC controller interface |
| NFC | near-field communication |
| MW | middleware |
| PLL | phase-locked loop |
| P2P | peer to peer |
| RF | radio frequency |
| SDA | serial data |
| SMCU | secure microcontroller |
| SW | software |

## References

[1] AOSP r3 tag: **https://android.googlesource.com/platform/manifest-b** android-13.0.0_r3
[2] Source control tools: **https://source.android.com/docs/setup/download**
[3] i.MX: **https://www.nxp.com/design/software/embedded-software/i-mx-software/android-os-for-i-mxapplications-processors:IMXANDROID**
[4] PN7220 kernel driver: **https://github.com/NXPNFCLinux/nxpnfc/tree/PN7220-Driver**
[5] PN7220 MW: **https://github.com/NXPNFCLinux/PN7220_Android13**

## Note about the source code in the document

disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## Legal information

### 8.1 Definitions

**Draft —** A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

### 8.2 Disclaimers

**Limited warranty and liability —** Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including – without limitation lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

**Right to make changes** — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

**Suitability for use** — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

**Applications** — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

**Terms and conditions of commercial sale** — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at **http://www.nxp.com/profile/terms**, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

**Export control** — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

**Suitability for use in non-automotive qualified products** — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

**Evaluation products** — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of noninfringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not

based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

**Translations** — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

**Security** — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at **PSIRT@nxp.com**) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

**NXP B.V. –** NXP B.V. is not an operating company and it does not distribute or sell products.

### 8.3 Licenses

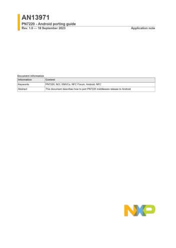
Purchase of NXP ICs with NFC technology — Purchase of an NXP Semiconductors IC that complies with one of the Near Field Communication (NFC) standards ISO/IEC 18092 and ISO/IEC 21481 does not convey an implied license under any patent right infringed by implementation of any of those standards. Purchase of NXP Semiconductors IC does not include a license to any NXP patent (or other IP right) covering combinations of those products with other products, whether hardware or software.

### 8.4 Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.
NXP — wordmark and logo are trademarks of NXP B.V.
EdgeVerse — is a trademark of NXP B.V.
i.MX — is a trademark of NXP B.V.
I2C-bus — logo is a trademark of NXP B.V.
Oracle and Java — are registered trademarks of Oracle and/or its affiliates.

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

## Documents / Resources

| | |
|---|---|
| <br>AN13971<br>PN7220 - Android porting guide<br>Rev. 1.0 — 18 September 2023 Application note<br>Document information<br>Information: Content<br>Keywords: PN7220, NCI, EMVCo, NFC Forum, Android, NFC<br>Abstract: This document describes how to port PN7220 middleware release to Android. | NXP PN7220 Compliant NFC Controller [pdf] User Guide<br>PN7220 Compliant NFC Controller, PN7220, Compliant NFC Controller, NFC Controller, Controller |

## References

- User Manual