

Identium CX1500N SDK

Integration Guide for Android

Version 2.9 · 7 October 2026



This guide explains how to integrate the Identium CX1500N UHF RFID SDK into an Android application, how to build it from source, and how to verify its support for Android's 16 KB memory page size.

The SDK is delivered as a single self-contained Android library: `IdentiumCX1500N.aar`. It contains everything required — native libraries and third-party JARs are bundled inside it.

1. What is in the package

Path	Description
sdk/IdentiumCX1500N.aar	The SDK. This single file is all an integrator needs.
docs/	Full API and protocol reference (Chinese), plus an English developer guide covering RF link configuration and parameter persistence.
ModuleAPI/	SDK library module (Java + C). Builds the AAR.
uhfDemo/	Demo application, integrated the same way a customer would integrate it.
tools/	Build and maintenance scripts.

2. Android 16 KB page size — current status

Google Play requires applications targeting Android 15 or newer to support 16 KB memory pages. This is a memory page size, not a file size — nothing needs to be made smaller. Native libraries must be laid out so the system can map them on a device whose page size is 16 KB.

Status of this release, verified by loading each library on a real Android 15 device with a 16 KB page size and inspecting the memory permissions applied:

Native library	64-bit (arm64-v8a)	Notes
libModuleAPI.so	Ready	Built with the correct 16 KB linker setting.
libpower.so	Ready	Built with the correct 16 KB linker setting.
libSerialPortHc.so	Not ready	Still uses a 4 KB layout. See below.

One library is still outstanding

libSerialPortHc.so is a prebuilt third-party library that provides serial port access and handheld power-rail control. On a 16 KB device it loads without an error but its code is mapped non-executable, so the application fails the first time it calls into it.

This library must be rebuilt from source with `-Wl, -z, max-page-size=16384`. It cannot be corrected by editing the file: the fix changes the memory layout, which only the linker can do. Identium has raised this with the module manufacturer.

Impact: applications built on this release will run correctly on CX1500N handheld hardware, which uses 4 KB pages. The limitation affects submission to Google Play and any future 16 KB device.

32-bit (armeabi-v7a) libraries are exempt — devices with 16 KB pages are 64-bit only.

3. Requirements

Item	Version
Android	5.0 (API 21) minimum; API 22 or newer recommended
Android Gradle Plugin	8.11.0
Android NDK	27.0.12077973 (only needed to build the SDK from source)
compileSdk	34
JDK	17
ABIs	arm64-v8a, armeabi-v7a

The UHF module is real hardware

The module is reached over a serial port and GPIO lines. The SDK only works on CX1500N handheld hardware — it cannot run on an emulator, and it will not work on a standard phone or tablet.

4. Adding the SDK to your application

Copy `IdentiumCX1500N.aar` into your module's `libs/` folder, then declare it:

```
android {
    defaultConfig {
        minSdk 22
        ndk { abiFilters 'arm64-v8a', 'armeabi-v7a' }
    }
}

dependencies {
    implementation files('libs/IdentiumCX1500N.aar') // the only SDK dependency
    implementation 'androidx.appcompat:appcompat:1.7.0'
    implementation 'com.google.android.material:material:1.12.0'
}
```

Add the permissions your application actually needs. Storage permissions are only required if you export scan data to external storage:

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
    android:maxSdkVersion="32" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
    android:maxSdkVersion="29" />
```

5. Basic usage

Everything goes through the `UHFReader` singleton. Connect once, register a listener, then start and stop inventory as needed.

```
// Connect to the module
UHFReaderResult<Boolean> r = UHFReader.getInstance().connect();
if (r.getResultCode() == UHFReaderResult.ResultCode.CODE_SUCCESS) {

    // Tags arrive on a background thread - marshal to the UI thread before
    // touching any view.
    UHFReader.getInstance().setOnInventoryDataListener(tags -> {
        runOnUiThread() -> showTags(tags));
    });

    UHFReader.getInstance().startInventory();
}

// ... later
UHFReader.getInstance().stopInventory();
UHFReader.getInstance().disconnect();
```

Inventory callbacks run on a background thread

The listener is invoked from the SDK's own reader thread, not the main thread. Do not update views directly from it, and keep the callback short — a slow callback reduces read throughput.

Protocol parameters — session, target, Q and the RF link profile — are **module** settings, not per-command arguments. They take effect immediately and the module keeps them until it loses power, at which point it reverts to the values stored in its flash. Re-apply them after every power cycle. The English developer guide in docs/ explains the RF link profiles in detail.

6. Building from source

Point the build at your Android SDK, then build. The demo build refreshes the AAR automatically.

```
# Point the build at your Android SDK (either method works)
cp local.properties.example local.properties # then set sdk.dir=...
export ANDROID_HOME=/path/to/android/sdk      # or just use the environment

# Produce the distributable SDK -> sdk/IdentiumCX1500N.aar
./gradlew :ModuleAPI:makeSdkAar

# Build the demo application (refreshes the AAR first)
./gradlew :uhfDemo:assembleDebug
```

The demo APK is written to uhfDemo/build/outputs/apk/debug/ as IdentiumCX1500N-Demo_2.9.apk.

7. Verifying 16 KB support yourself

If you rebuild any native library, or receive an updated one, verify it before shipping. Checking the declared alignment alone is **not** sufficient — a library can report 16 KB alignment and still fail to load. Three conditions must all hold:

- The declared alignment of every loadable segment is at least 16 KB.
- Each segment's file offset is consistent with its address in memory — otherwise the library is mapped out of position and fails to load.
- No two segments share a 16 KB page. If they do, code and data get one set of memory permissions and the library's code can end up non-executable. This is the case that loads successfully and crashes later.

Identium can supply a verification tool that checks all three. To confirm on a real device, create an emulator from an Android 15 or newer system image whose name includes *16 KB Page Size*, and check:

```
adb shell getconf PAGE_SIZE          # must print 16384
```

8. Troubleshooting

Symptom	Likely cause
Connect fails on a handheld	The module is powered down, or another application already holds the serial port. Disconnect cleanly before leaving your activity.
Crash when the application starts reading, on a 16 KB device	The libSerialPortHc.so limitation in section 2. Expected until the rebuilt library arrives.
Nothing works on an emulator	Expected. The SDK needs the handheld's serial port and GPIO lines.
Settings are lost after the module is power cycled	Protocol parameters live in module RAM. Re-apply them on reconnect.
Google Play rejects the upload for 16 KB support	See section 2. A rebuilt libSerialPortHc.so is required.

9. Support

Identium Tech Solutions Pvt. Ltd. — identiumone.com

When reporting an issue, please include the handheld model, the Android version, the SDK version above, and a logcat capture covering connect and the failing operation.