PTX130R AIS Product Support Library

Introduction

This document describes the usage of the Product Support Library for PTX130R application using Android AOSP and running it on the Qualcomm® Robotics (RB3) Dragonboard-845c board.

Contents

1. Requirements

This document applies to:

- AIS SDK for PTX1xxR family

2. Software

In this section, all software related topics are analyzed in detail and relevant instructions are given to final user.

2.1 Software Documentation

To have a proper understanding of the provided Product Support Library tool, reference the following documents:

- PTX130R AIS SDK for Android Integration User Manual
- PTX1xxR NFC IoT-Reader OS Stack Integration Manual

2.2 Software Tools

- CMake
- Android NDK Package
- ADB and Fastboot
- Config Tool

2.3 Software Setup

This section describes the step-by-step guidelines for deploying and debugging the downloaded SDK content to the hardware boards.

2.3.1 Building the Project

The Android NDK toolset is used to build the software and can be installed by following the instructions in the Android NDK Guide. All the other components are just usual tools like Cmake and ADB.

The Native Development Kit (NDK) is a set of tools that allows you to use C and C++ code with Android and provides platform libraries that are used to manage native activities and access physical device components, such as sensors and touch input. This approach is taken so that Renesas' C or C++ libraries are used.

- **export ABI=arm64-v8a** – Indicates the Android ABI to be used.
- **export NDK=~/Android/Sdk/ndk/<ndk_version>** – The NDK toolchain location.
- **export MINSDKVERSION=android-34** – This is dependent on the Android SDK version and on the device that will run this application.

Use this command to configure the CMake environment:

```
cmake . -DCMAKE_TOOLCHAIN_FILE=$NDK/build/cmake/android.toolchain.cmake \
-DANDROID_ABI=$ABI -DANDROID_PLATFORM=android-$MINSDKVERSION
```

To start the build use the following command to start the build:

```
`cmake --build . --target badger_cli --config Debug`
```

The configuration can be set to either **Release** or **Debug**.

2.3.2 Platform-Specific Implementation

The system architecture of IoT-Reader(OS) SDK requires two system components:

- **Host Controller** – Represented in this case by Qualcomm® Robotics (RB3) Dragonboard-845c board.
- **NFC Chip** – Represented by PTX130R-EB-ST board

To ensure a flexible solution, the user is responsible to implement the interfacing between the two system components.

For the software architecture, this layer is represented by the Platform component.

This component is dependent on the platform/MCU application processor being used. It also depends on the physical hardware interface being used (SPI or I2C) between the application processor platform and the Renesas chip. The Platform component includes the following sub-modules:

- **Interface** – Implements the driver for the physical interface used for communication with the PTX chip.
- **GPIO** – Implements the driver for the GPIO used for IRQ. This submodule is needed when SPI or I^2C are used as the physical interface.

2.3.2.1 HAL Interface

For communicating with the PTX130R chip, the HAL interface needs to be properly configured.

There are two options for communication:

- SPI
- I^2C

For SPI setup, the hardware connection needs to be setup depending on the platform used. In case of the reference board, the **/dev/spidev2.0** device is used along with the following configurations:

- IRQ GPIO pin
- SPI speed

For I^2C, the device in use is **/dev/i2c-1** along with the following configurations:

- device i2c address
- IRQ GPIO pin
- I^2C speed

These configurations will differ depending on the platform used for running the application.

A detailed description of every API can be found in the PTX1xxR NFC IoT-Reader API for OS Stack Integration (SDK v7.2.0) User Manual

2.4 Software Capabilities

The AIS PSL application is used to gather measurement data for:

- Signal detector
- Q-measurement
- RSSI measurement
- LPCD I/Q Limit
- RX gain measurement
- Temperature calibration

All interactions with the application are performed through a command line interface where the user can select the required option.

```
db845c:/data/test # ./badger_cli -h
Successful initialization of the stack.

Print Revision Information...

C-Stack Revision.......: 0x2728
Local Modifications....: 0x0000
NSC-Code Revision......: 9245
NSC-Toolchain Revision.: 8362
Chip-ID................: 0x00
Product-ID.............: Unknown

Print Revision Information...OK
RT logging...ENABLED.
Option --help requested
Usage: ./badger_cli [OPTIONS]
Options:
  -i, --interface <INTERFACE>   Specify communication interface (e.g., SPI, I2C, UART)
  -d, --device <DEVICE>         Specify device in use
  -c, --config <FILE>           Specify configuration file
  -s, --signal-detector         Enable signal detection (no argument)
  -t, --type <TYPE>             Specify type for signal detector
  -q, --q-measurement           Enable Q-measurement (no argument)
  -f, --frequency-step-size <SIZE> Set frequency step size
  -r, --rssi                    Enable RSSI (no argument)
  -p, --power <LEVEL>           Specify power level
  -l, --lpcd                    Enable Low Power Card Detection (no argument)
  -g, --rx-gain <GAINS>         Set RX gain (comma-separated values allowed)
  -x, --temp-calib              Enable temperature sensor calibration
  -h, --help                    Show this help message
SW reset ... OK
Closing stack ... OK
Successful log written to IotRd_Int_Log_XXX.txt
Good Bye
```

2.4.1 Signal Detector

The command-line interface (CLI) application will perform the **PCD Rx Threshold** measurement for PSL (Proximity Signal Level) when the `-s/--signal-detector` flag is set.

Example:

```
./badger_cli -i I2C -d /dev/i2c-11 -s
```

2.4.1.1 Functionality

The user specifies the technology type using the `-t / --tech-type` parameter. Supported values include:

- Type A106, Type A424, Type A212, Type A848
- Type B106, Type B212, Type B424, Type B848
- Type V
- Type F212, Type F424

The application executes the PCD Rx Threshold measurement. After a successful execution, the resulting threshold value is printed to the CLI.

If the test fails, an error message is displayed.

2.4.2 Q-measurement

2.4.2.1 Description

The command-line interface (CLI) application will perform the **PSL Q-Measurement** when the `-q/--q-measurement` parameter is set.

Example:

```
./badger_cli -i I2C -d /dev/i2c-11 -q
```

2.4.2.2 Functionality

Users can optionally specify the **Frequency Step Size** using the `-f/--frequency-step-size` parameter.

- Accepted values match those in PTX1xxR IOT Config Tool.
- Defaults to **50kHz** if not set.

Upon successful execution, the following measurement results are printed on the CLI:

- maximum amplitude frequency
- maximum amplitude
- lower/upper 3dB frequency
- lower/upper 3dB amplitude

If the test fails, an error message is displayed.

If the test succeeds, a **CSV** file is generated and stored on the file system.

- The file contains measurement values identical to those displayed in the PTX1xxR IOT Config Tool graph.

The Q-measurement data can be visualized with the following python example:

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the CSV file
file_path = 'qmeasure.csv'  # Change path if needed
df = pd.read_csv(file_path)

# Convert columns to numeric types
df['freq[kHz]'] = pd.to_numeric(df['freq[kHz]'], errors='coerce')
df['ampl[dBfs]'] = pd.to_numeric(df['ampl[dBfs]'], errors='coerce')
```

```
df['I-CH[LSB]'] = pd.to_numeric(df['I-CH[LSB]'], errors='coerce')
df['Q-CH[LSB]'] = pd.to_numeric(df['Q-CH[LSB]'], errors='coerce')

# Plot frequency vs amplitude
plt.figure(figsize=(10, 5))
plt.plot(df['freq[kHz]'], df['ampl[dBfs]'], marker='o')
plt.title('Frequency vs Amplitude')
plt.xlabel('Frequency (kHz)')
plt.ylabel('Amplitude (dBfs)')
plt.grid(True)
plt.tight_layout()
plt.show()

# Plot I and Q channels
plt.figure(figsize=(10, 5))
plt.plot(df['freq[kHz]'], df['I-CH[LSB]'], label='I-Channel')
plt.plot(df['freq[kHz]'], df['Q-CH[LSB]'], label='Q-Channel')
plt.title('Frequency vs I/Q Channels')
plt.xlabel('Frequency (kHz)')
plt.ylabel('LSB Value')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

2.4.3 RSSI Measurement

2.4.3.1 Description

The command-line interface (CLI) application will perform the **PSL RSSI** measurement when the `-r/--rssi` parameter is set.

Example:

```
./badger_cli -i I2C -d /dev/i2c-11 -r
```

2.4.3.2 Functionality

Requires the `-p/--power` parameter, which accepts a value in the range [0%, 100%].

Upon successful execution, the resulting threshold value is printed on the CLI. If the test fails, an error message is displayed.

2.4.4 LPCD I/Q Limit Measurement

2.4.4.1 Description

The command-line interface (CLI) application will perform the **PSL LPCD I/Q Limit** measurement when the `-l/--lpcd` parameter is set.

Example:

```
./badger_cli -i I2C -d /dev/i2c-11 -l
```

2.4.4.2 Functionality

Upon successful execution, the resulting **I (In-Phase)** and **Q (Quadrature)** values are printed on the CLI. If the test fails, an error message is displayed.

2.4.5 Rx-Gain Measurement

2.4.5.1 Description

The command-line interface (CLI) application will perform the **PSL Rx-Gain** measurement when the `-g/--rx-gain` parameter is set.

Example:

```
./badger_cli -i I2C -d /dev/i2c-11 -g 10,20,10,20
```

2.4.5.2 Functionality

Supports the following parameters:

- `pga1Min, pga1Max, pga2Min, pga2Max`
- Accepted values are the same as in the **PTX1xxR IOT Config Tool**.

Upon successful execution, the resulting **PGA1** and **PGA2** values are printed on the CLI. If the test fails, an error message is displayed.

3. Hardware

The documentation listed below describes the necessary hardware components required to run the AIS PSL application on an Android OS.

- Qualcomm RB3 platform: Qualcomm RB3 platform
- PTX130R-EB-ST: PTX130R-EB-ST-QFN56-POS

4. Conclusions

The CLI-based application provides a comprehensive suite of PSL measurement functionalities, enabling users to execute various tests efficiently through command-line parameters. It supports Q-Measurement, RSSI Measurement, Rx-Gain Measurement, LPCD I/Q Limit Measurement, and PCD Rx Threshold Measurement, each with configurable parameters to ensure flexibility and accuracy. The application ensures **clear** CLI output, displaying measurement results upon successful execution or relevant error messages if a test fails. Additionally, for Q-Measurement, a CSV file is generated to store the measured values for further analysis.

5. Revision History

Revision	Date	Description
1.00	May 19, 2025	Initial release.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.