User manual

Linux® driver for ST25R300

Introduction

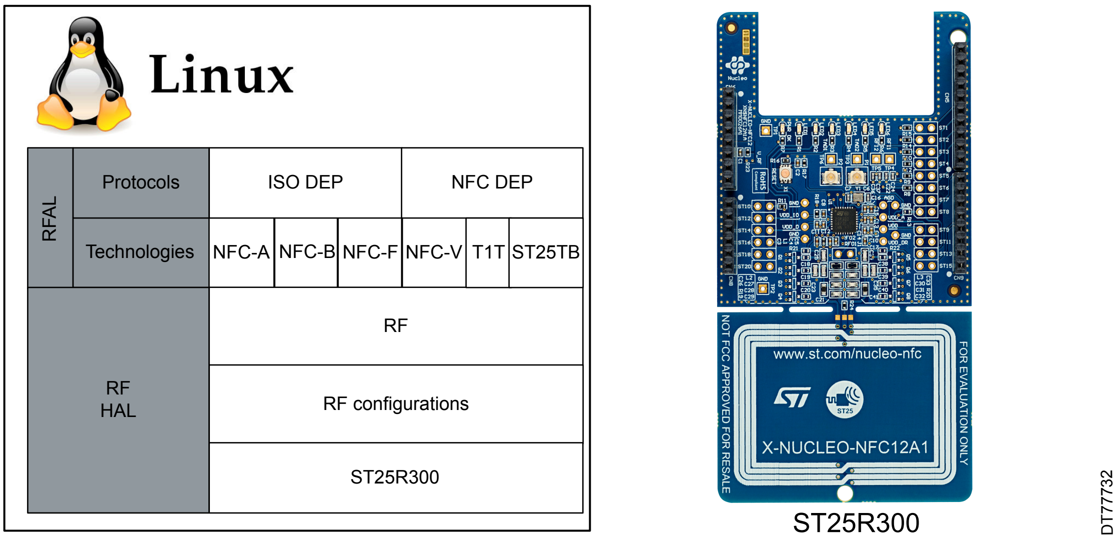
STSW-ST25R024 Linux® driver enables the Raspberry Pi® 4 to operate with the X-NUCLEO-NFC12A1 board, which contains the ST25R300 device.

This package ports the RF abstraction layer (RFAL) onto a Raspberry Pi 4 Linux platform, to operate with the board firmware, and provides a sample application detecting different types of NFC tags and mobile phones supporting P2P. The RFAL is the ST standard driver for ST25R300, high performance NFC universal devices / EMVCo readers. It is used, for instance, by the ST25R300-EVAL firmware.

STSW-ST25R024 supports all the ST25R300 lower-layer and some higher layer protocols for communication. The RFAL is written in a portable manner, so it can run on a wide range of devices based on Linux.

This document describes how the RFAL library can be used on a standard Linux system (in this case the Raspberry Pi 4) for NFC/RF communication. The code is highly portable and works with minor changes on any Linux platform.

Figure 1. RFAL library on Linux platform



1 Overview

1.1 Features

- Complete Linux user space driver (RF abstraction layer) to build NFC enabled applications using the ST25R300 device
- Linux host communication with the ST25R300 using SPI interface
- Complete RF/NFC abstraction (RFAL) for all major technologies and higher layer protocols:
  - NFC-A (ISO14443-A)
  - NFC-B (ISO14443-B)
  - NFC-F (FeliCa™)
  - NFC-V (ISO15693)
  - P2P (ISO18092)
  - ISO-DEP (ISO data exchange protocol, ISO14443-4)
  - NFC-DEP (NFC data exchange protocol, ISO18092)
  - Proprietary technologies (Kovio, B’, iClass, Calypso®)
- Sample implementation available with the X-NUCLEO-NFC12A1 expansion board, plugged into a Raspberry Pi 4
- Sample application to detect several NFC tag types and mobile phones
- Free, user-friendly license terms

1.2 Software architecture

Figure 1 shows the software architecture details of RFAL library on a Linux platform.

The RFAL is easily portable to other platforms by adapting the so-called platform files.

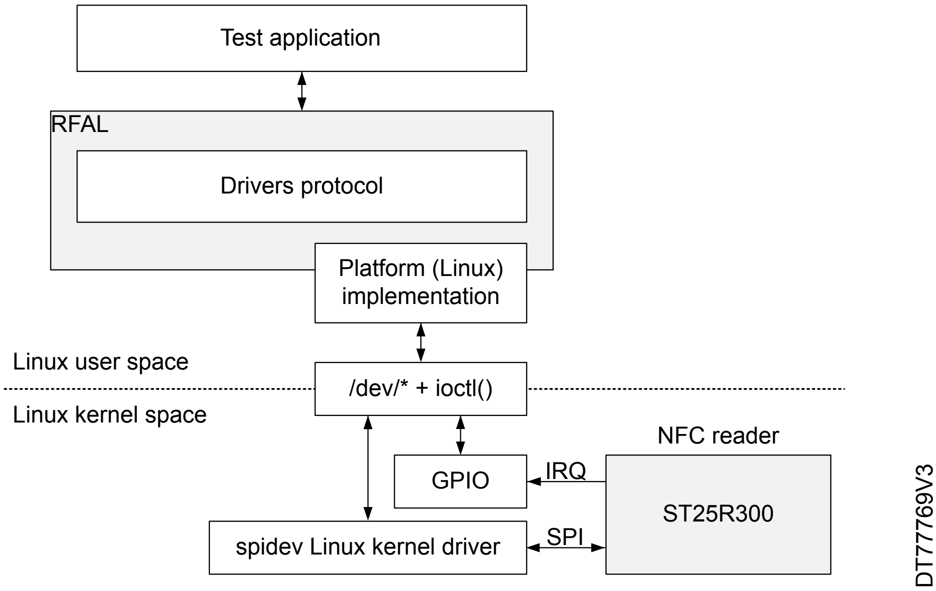
The header file *rfal_platform.h* contains macro definitions, to be provided and implemented by the platform owner. It provides platform specific settings like GPIO assignment, system resources, locks and IRQs, which are required for correct operation of the RFAL.

This demonstration implements the platform functions and provides a port of the RFAL into user space of Linux. A shared library file is generated, which is used by a demonstrative application to showcase the functionalities provided by the RFAL layer.

The Linux host uses the SPI character device interface, spidev, available from the Linux user space to perform SPI communication with the device.

Inside the Linux kernel, the spidev driver handles sending and receiving SPI frames to and from the devices.

Figure 2. RFAL software architecture on Linux



2 Hardware setup

2.1 Platform used

A Raspberry Pi 4 board with Raspberry Pi OS is used as Linux platform to build the RFAL library and interact with the ST25R300 over SPI.

The devices enable an application on Linux platform to detect and communicate with NFC devices.

2.2 Hardware requirements

- Raspberry Pi 4
- 8 GB micro SD card to boot Raspberry Pi OS (with its latest requirements)
- SD card reader
- X-NUCLEO-NFC12A1 board
- Bridge to connect the board with Raspberry Pi Arduino™ adapter for Raspberry Pi (part number ARPI600)

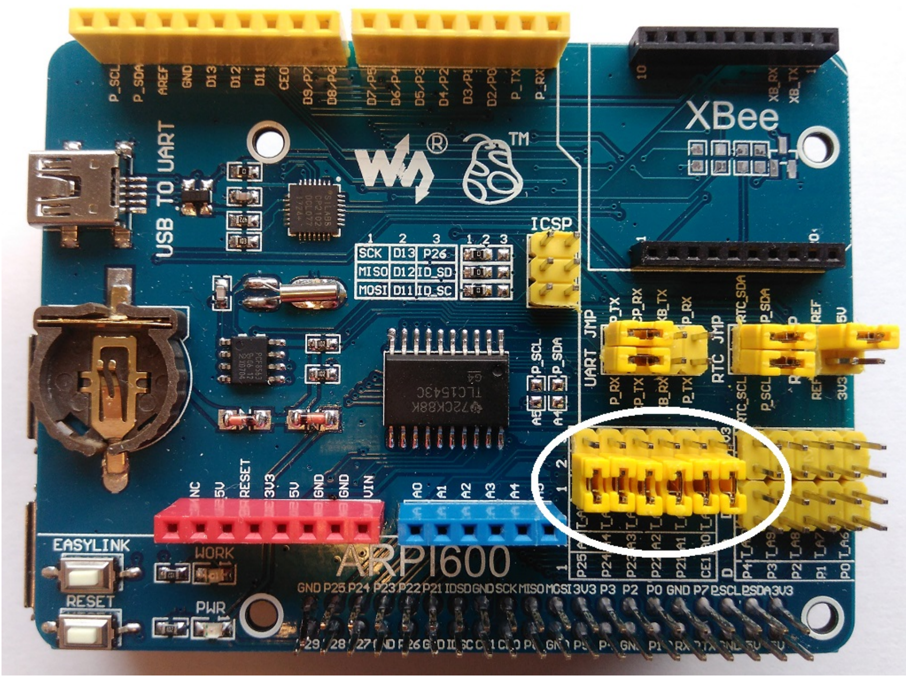
2.2.1 Hardware connections

The ARPI600 Raspberry Pi to Arduino adapter is used to connect the boards with the Raspberry Pi. The jumpers of the adapter board must be modified to connect it with the X-NUCLEO-NFC12A1 board.

Jumper setting

The jumpers for A5, A4, A3, A2, A1 and A0 shown in Figure 2 must be changed, respectively, to P25, P24, P23, P22, P21 and CE1. With this setting Raspberry's GPIO pin number 7 is used as interrupt line for X-NUCLEO-NFC12A1.

Figure 3. **Position of jumpers A5, A4, A3, A2, A1, and A0 on the adapter board**



Currently, this RFAL library demo uses the pin GPIO7 as the interrupt line (according to the jumper settings). If there is a requirement to change the interrupt line from GPIO7 to a different GPIO, the platform specific code (in file *pltf_gpio.h*) must be modified to change the definition of macro ST25R_INT_PIN from 7 to the new GPIO pin, to be used as interrupt line.

With the above jumper settings, the adapter board can be used to connect the X-NUCLEO NFC12A1 with Raspberry Pi board, as shown in the following figures.

Figure 4. **Hardware setup top view**

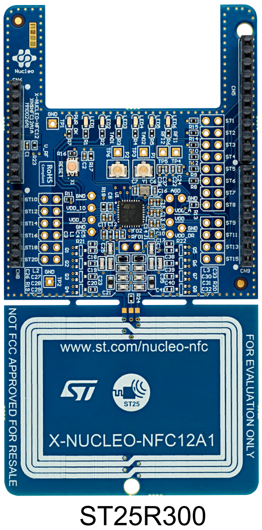
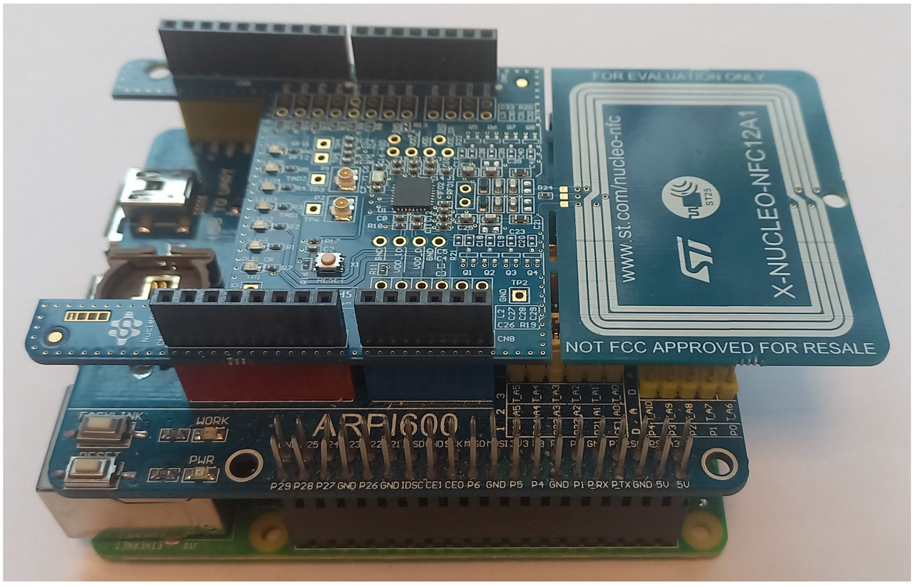


Figure 5. **Hardware setup side view (X-NUCLEO-NFC12A1)**



3 Linux environment setup

3.1 Booting Raspberry Pi

To setup the Linux environment, first install and boot the Raspberry Pi with Raspberry Pi OS, as explained below:

Step 1

Download the latest Raspberry Pi OS image from https://www.raspberrypi.com, then choose Raspberry Pi OS Desktop. For the tests below the version 2024-10-22-raspios-bookworm-arm64-full.img.xz (October 2024) has been used.

Step 2

Unzip the Raspberry Pi OS image and write it onto the SD card by following the instructions available in the section named “Writing an image to the SD card”.

Step 3

Connect the hardware:

- Connect the Raspberry Pi 4 to a monitor using a standard HDMI cable.
- Connect mouse and keyboard to Raspberry Pi’s USB ports.

It is also possible to work with Raspberry Pi using ssh. In this case. it is not required to connect the monitor, keyboard and mouse with Raspberry Pi. The only requirement is to have the PC with ssh inside the same network as the Raspberry Pi, and configure the IP address accordingly.

Step 4

Boot the Raspberry Pi 4 with an SD card. After booting, a Debian based Linux desktop appears on monitor.

3.2 Enable SPI on Raspberry Pi

The SPI driver inside the kernel communicates with the X-NUCLEO-NFC12A1 board through SPI. It is important to check if SPI is already enabled in the Raspbian Pi OS kernel configuration.

Check if /dev/spidev0.0 is visible in the Raspberry Pi environment. If it is not visible, enable the SPI interface using the utility “raspi-config” by following the steps described below.

Step 1

Open a new terminal on the Raspberry Pi and run the command “raspi-config” as root:

```
sudo raspi-config
```

This step opens a graphical interface.

Step 2

Select in the graphical interface the option named “Interfacing Options”.

Step 3

Select the option named “SPI”.
A new window appears with following text asking:
“Would you like the SPI interface to be enabled?”

Step 4

Select <Yes> in this window to enable SPI.

Step 5

Reboot Raspberry Pi.
The above steps will enable the SPI interface in Raspberry Pi environment after a reboot.

4 Build RFAL library and application

The RFAL demonstration of Linux is provided in an archive, such as en.STSW-ST25R024.xz

To build the RFAL library and application on Raspberry Pi, go through the following steps:

Step 1

Unzip the package on Raspberry Pi using the following command from the home directory

```
tar -xJvf en.STSW-ST25R024.xz
```

Step 2

Install cmake (if not done before) using the command

```
apt-get install cmake
```

RFAL library and application build system are based on cmake, for this reason it is required to install cmake to compile the package.

Step 3

To build the RFAL library and application, go to the build directory

```
cd ST25R300_RFAL_v4.0.2_Linux_demo_v1.0/
```

From there, run the command

```
cmake ..
```

In the above command “..” indicates that top level CMakeLists.txt exists in the parent directory.

This command creates the makefile used in the next step to build the library and application.

Step 4

Run the make command to build the RFAL library and application:

```
make
```

This command first builds the RFAL library, and then the application on top of it.

5 How to run the application

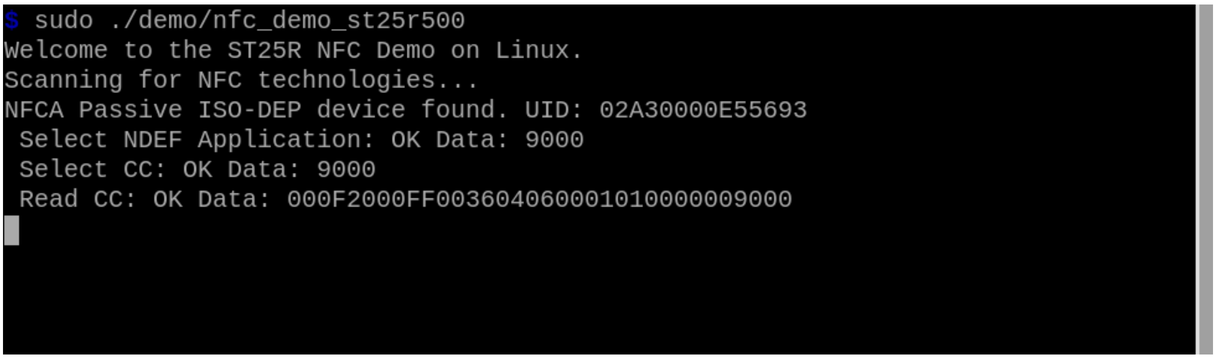
A successful build-up generates an executable named “nfc_demo_st25r500” at location /build/demo.

By default, the application needs to be run with root rights from the path ST25R300_RFAL_v4.0.2_Linux_demo_v1.0:

```
sudo ./nfc_demo_st25r500
```

The application starts to poll for NFC tags and mobile phones, then displays the found devices with their UID, as shown in Figure 6.

Figure 6. Display of found devices



To terminate the application press Ctrl + C.

Revision history

Table 1. **Document revision history**

Date	Revision	Changes
27-Aug-2025	1	Initial release.

Contents

List of tables

List of figures