User manual

Getting started with the X-CUBE-NFC6 high performance HF reader/NFC initiator IC software expansion for STM32Cube

Introduction

The X-CUBE-NFC6 software expansion for STM32Cube provides complete middleware for STM32 to control applications using the ST25R3916/ST25R3916B high performance NFC front-end IC supporting NFC initiator, target, reader, and card emulation modes.

The expansion is built on top of STM32Cube software technology to ease portability across different STM32 microcontrollers.

The software comes with sample implementations of the drivers running on the X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 expansion board plugged on top of a NUCLEO-L053R8 or NUCLEO-L476RG development board.

Related links

Visit the STM32Cube ecosystem web page on www.st.com for further information

1 Acronyms and abbreviations

Table 1. List of acronyms

Acronym	Description
NFC	Near field communication
RFAL	RF abstract layer
P2P	Peer-to-peer
MCU	Microcontroller unit
BSP	Board support package
HAL	Hardware abstraction layer
LED	Light emitting diode
SPI	Serial peripheral interface
CMSIS	Arm® Cortex® microcontroller software interface standard

2 X-CUBE-NFC6 software expansion for STM32Cube

2.1 Overview

The X-CUBE-NFC6 software package expands the STM32Cube functionality.

The package key features are:

- Complete middleware to build applications using the ST25R3916/ST25R3916B high performance HF reader/NFC front end IC
- Sample application to detect NFC tags of different types and mobile phones supporting P2P, card emulation mode, and read/write
- Sample application to read and write NDEF messages
- Sample implementations available for the X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 expansion board plugged onto a NUCLEO-L053R8 or NUCLEO-L476RG development board
- Easy portability across different MCU families, thanks to STM32Cube
- Complete RF/NFC abstraction (RFAL) for all major technologies, including complete ISO-DEP and NFC-DEP layers
- Free, user-friendly license terms

This software contains high performance HF reader/NFC front end IC drivers for the ST25R3916/ST25R3916B device, running on STM32. It is built on top of STM32Cube software technology to ease portability across different STM32 microcontrollers.

This firmware package includes component device drivers, a board support package and a sample application demonstrating usage of X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 expansion board with STM32 Nucleo boards.

A sample application configures the ST25R3916/ST25R3916B in a polling loop for active and passive device detection. When a passive tag or active device is detected, the reader field signals the detected technology by switching a corresponding LED on. It is also possible to set the ST25R3916/ST25R3916B in an inductive wake-up mode by pressing the user button. During this polling loop the sample application also sets the ST25R3916/ST25R3916B in card emulation mode to detect the presence of a reader.

The demo logs all activities with ST-LINK virtual COM port to the host the system.

The supported RFID technologies in this demo are:

- ISO14443A/NFCA
- ISO14443B/NFCB
- Felica/NFCF
- ISO15693/NFCV
- Active P2P
- Card Emulation Type A and F

The second sample application uses the lib NDEF middleware to read or write NDEF messages from/to tags. It is available in a second project target for the three development environment tools.

2.2 Architecture

This fully compliant software expansion for STM32Cube lets you develop applications using the ST25R3916/ST25R3916B high performance HF reader/NFC initiator IC. It is based on the STM32CubeHAL hardware abstraction layer for the STM32 microcontroller and extends STM32Cube with a board support package (BSP) for the X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 expansion board.

Application software can access and use the X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 expansion board through the following layers:

- **STM32Cube HAL layer**: the HAL driver layer provides a simple set of generic, multi-instance APIs (application programming interfaces) to interact with the upper layers (application, libraries and stacks). These generic and extension APIs are directly built on a common architecture and allow overlying layers like middleware to implement their functions without depending on specific microcontroller unit (MCU) hardware information. This structure improves the library code reusability and guarantees easy portability across other devices.

- **Board support package (BSP) layer**: provides support for the peripherals on the STM32 Nucleo board (apart from the MCU). This set of APIs provides a programming interface for certain board-specific peripherals like the LED, the user button etc. This interface also helps you identify the specific board version.
- **Middleware NRF abstraction layer (RFAL)**: RFAL provides several functions for RF/NFC communication. It groups the different RF ICs (existing ST25R3911B product family and future ST25R391x devices) under a common and easy to use interface.

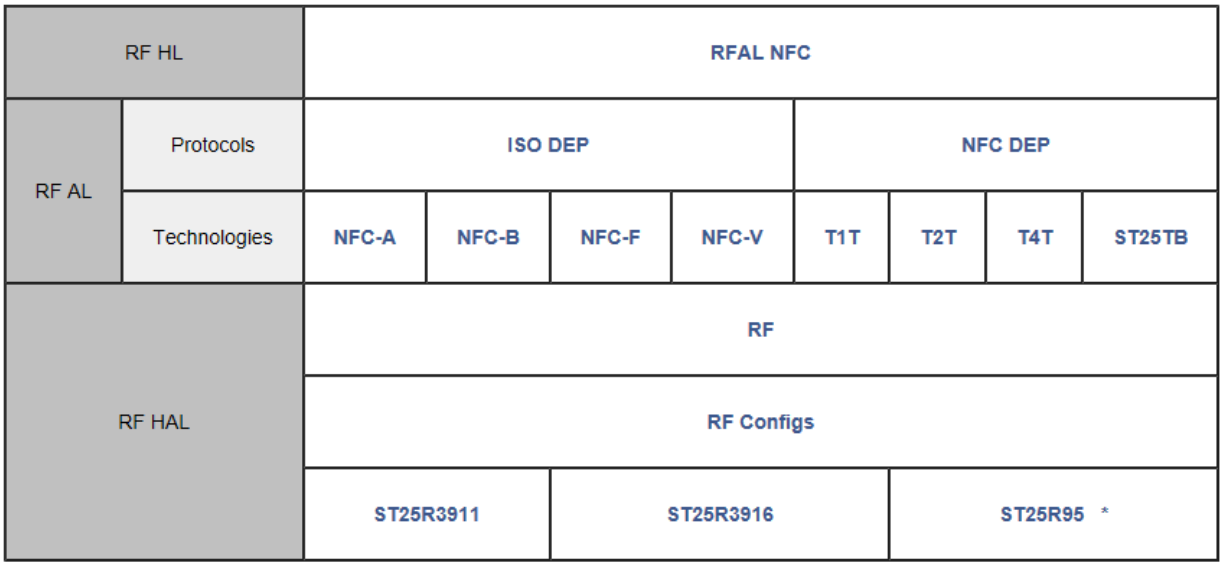
The protocols provided by RFAL are:

- ISO-DEP (ISO14443-4 Data Link Layer, T=CL)
- NFC-DEP (ISO18092 Data Exchange Protocol)
- NFC-A \ ISO14443A (T1T, T2T, T4TA)
- NFC-B \ ISO14443B (T4TB)
- NFC-F \ FeliCa (T3T)
- NFC-V \ ISO15693 (T5T)
- P2P \ ISO18092 (NFCIP1, Passive-Active P2P)
- ST25TB (ISO14443-2 Type B with Proprietary Protocol)

Internally, the RFAL is divided into three sub layers:

- RF HL - RF higher layer
- RF HAL- RF hardware abstraction layer
- RF AL - RF abstraction layer

Figure 1. RFAL block diagram



* Future devices added to the ST25R family

The modules in the RF HAL are chip-dependent, they implement the RF IC driver, configuration tables and specific instructions for the HW to perform the physical RF functions.

The interface for the caller is a shared RF header file which provides the same interface for upper layers (for all chips).

The RFAL can be broken down into two further sub layers:

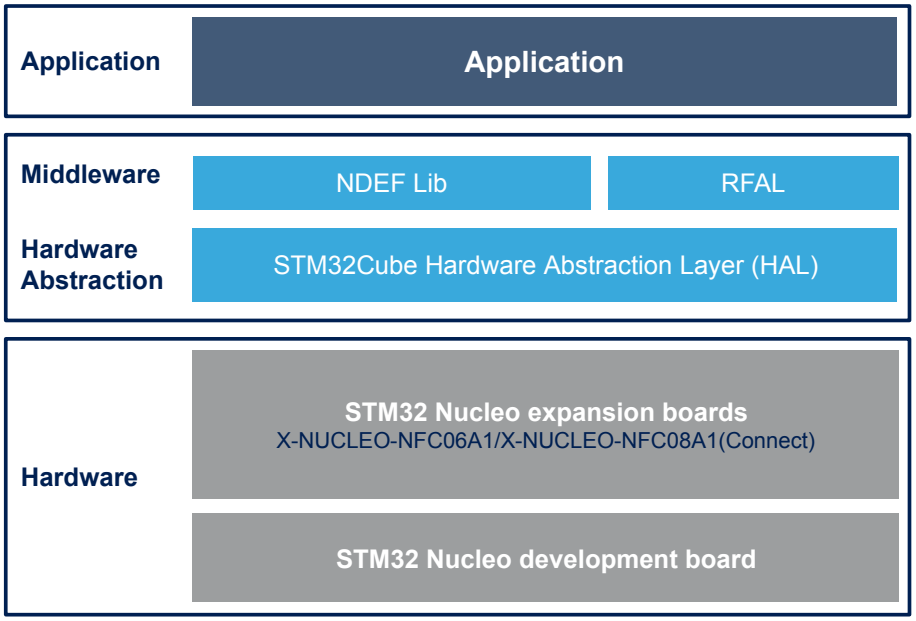
- Technologies: technology modules which implement all the specifics, framing, timings, etc.
- Protocols: protocol implementation including all the framing, timings, error handling, etc.

On top of these, the application layer uses RFAL functions like NFC Forum Activities (NFCC), EMVCo, DISCO/ NUCLEO demo, etc.

The RFAL NFC module provides an interface to perform the common activities as poller/listener device.

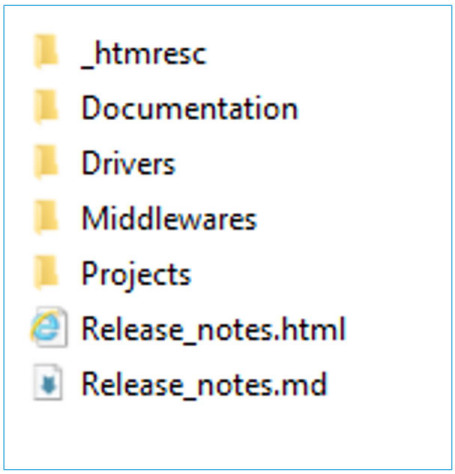
Access to the lowest functions of the ICs is granted by the RF module. The caller can make direct use of any of the RF technology or protocol layers without requiring any specific hardware configuration data.

Figure 2. **X-CUBE-NFC6 software architecture**



2.3 Folder structure

Figure 3. **X-CUBE-NFC6 package folders structure**



The following folders are included in the software package:

- **Documentation**: this folder contains a compiled HTML file generated from the source code which details the software components and APIs.
- **Drivers**: this folder contains the HAL drivers, the board-specific drivers for each supported board or hardware platform, including the on-board components, and the CMSIS vendor-independent hardware abstraction layer for the Cortex-M processor series.
- **Middlewares**: this folder contains RFAL (RF abstraction layer). RFAL provides several functions required to perform RF/NFC communication.
  The RFAL groups the different RF ICs (ST25R3911/ST25R3916/ST25R3916B and future ST25R391x devices) under a common and easy to use interface.

- **Projects:** this folder contains two sample application examples:
  - Tag Detect-Card emulation
  - Read and Write of NDEF messages

  They are provided for the NUCLEO-L476RG or NUCLEO-L053R8 platform for three development environments (IAR Embedded Workbench for ARM, Keil Microcontroller Development Kit (MDK-ARM), and STM32CubeIDE.

2.4 APIs

Detailed technical information about the APIs available to the user can be found in a compiled CHM file located inside the “RFAL” folder of the software package where all the functions and parameters are fully described.

Detailed technical information about the NDEF APIs is available in the .chm file stored in the "doc" folder.

2.5 Sample application

A sample application using the X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 expansion board with the NUCLEO-L476RG or NUCLEO-L053R8 development board is provided in the “Projects” directory. Ready-to-build projects are available for multiple IDEs.

In this application, NFC tags of different types of mobile phones supporting P2P are detected by the ST25R3916/ST25R3916B high performance HF reader/NFC front end IC (for further details, refer to the CHM documentation file generated from the source code).

After system initialization and clock configuration, LED101, LED102, LED103, LED104, LED105 and LED106 blink for 3 times. Then LED106 glows to indicate the reader field has been activated.

When a tag is detected in the proximity, a LED is switched on as listed below.

Table 2. LED Lit on tag detection

NFC tag type	LED lit on tag detection
NFC TYPE F	LED101/Type F
NFC TYPE B	LED102/Type B
NFC TYPE A	LED103/Type A
NFC TYPE V	LED104/Type V
NFC TYPE AP2P	LED105/Type AP2P

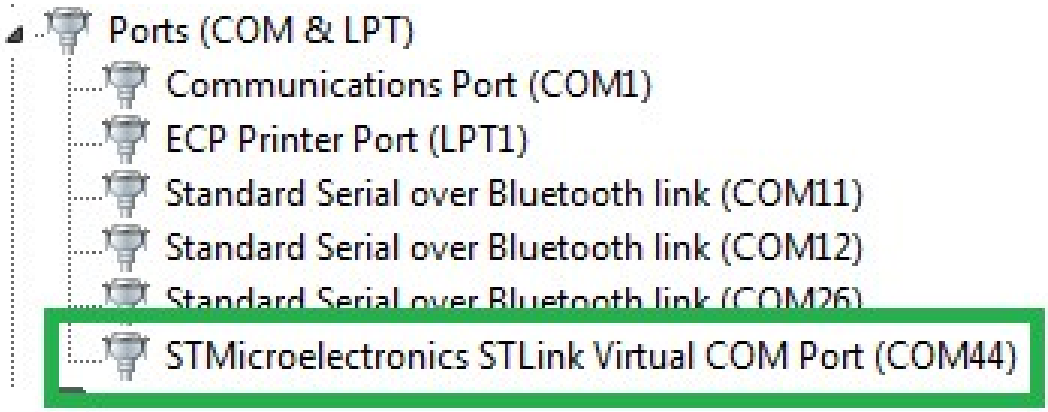
If a reader approaches the X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 expansion board, the software enters card emulation mode and, depending on the command type sent, it switches NFC TYPE A and/or NFC TYPE F LED on.

By default, the X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1 does not write any data to the tag, but this possibility can be enabled by a pre-processor define in the file demo.h.

Card emulation and poller mode can also be enabled/disabled with the same procedure.

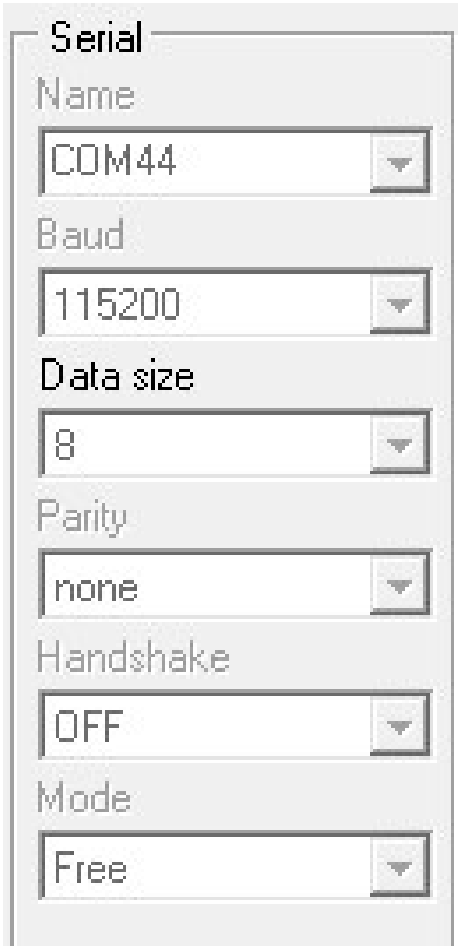
The ST virtual communication port interface is also included in the package. Once the board is powered on, the board is initialized and enumerated as STLink virtual COM port.

Figure 4. **Virtual COM port enumeration**



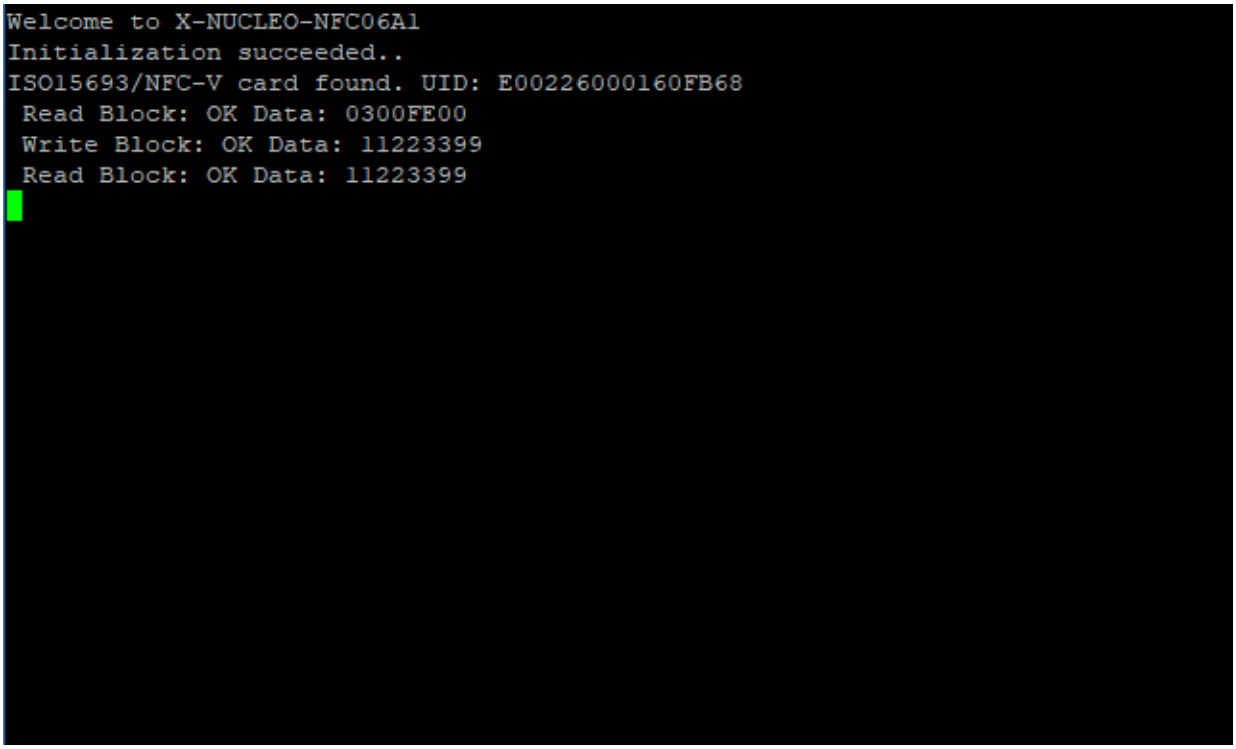
After checking the virtual COM port number, open a Windows terminal (HyperTerminal or similar) with the configuration shown below (enable option: Implicit CR on LF, if available).

Figure 5. **UART serial communication configuration**



The terminal window returns several messages similar to those shown below to confirm successful connection.

Figure 6. **X-NUCLEO-NFC06A1 expansion board successful initialization**



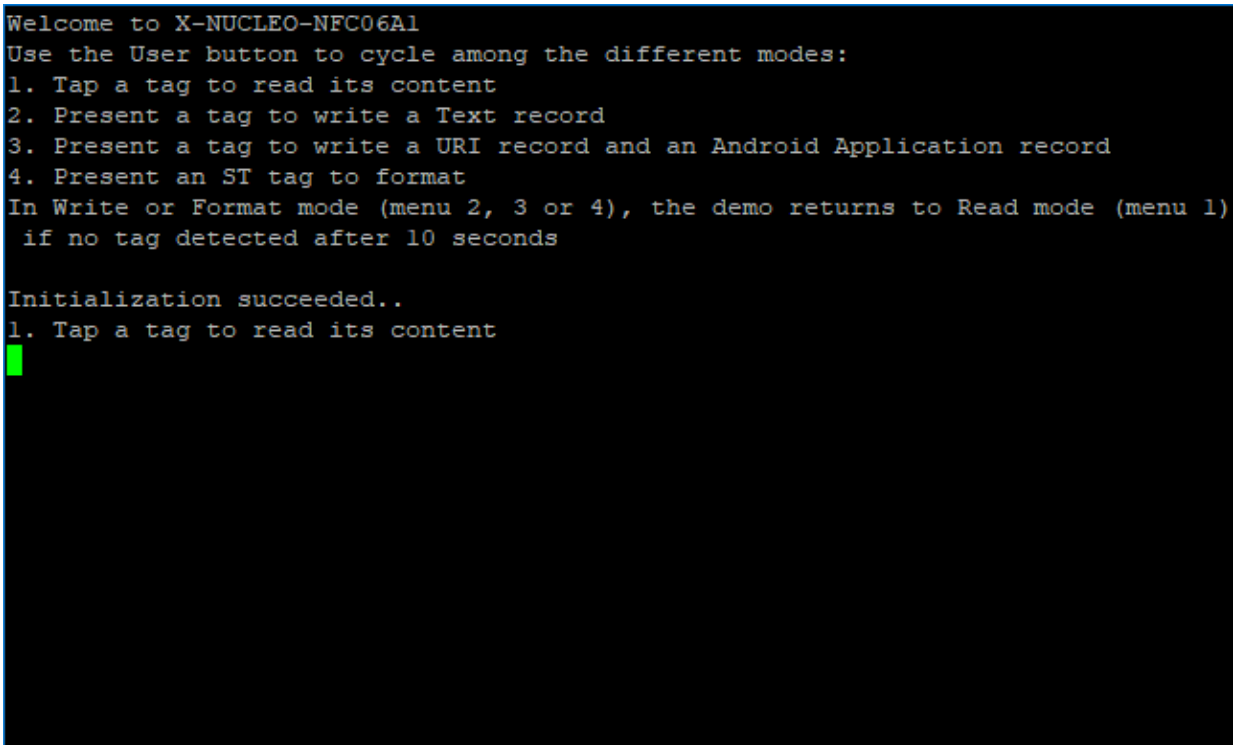
The second sample application is available by selecting the second project target called “STM32L476RG-Nucleo_PollingTagDetectNdef”. This application manages NDEF message on tags.

When the firmware starts, a menu is displayed on the console log.

The user button allows you to cycle through several options, including read NDEF content, write a text record, write an URI record and format tag for NDEF content.

After selecting the demo, tap a tag to see the demo running.

Figure 7. **X-NUCLEO-NFC06A1 expansion board user button options**



3 System setup guide

3.1 Hardware description

3.1.1 STM32 Nucleo

STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line.

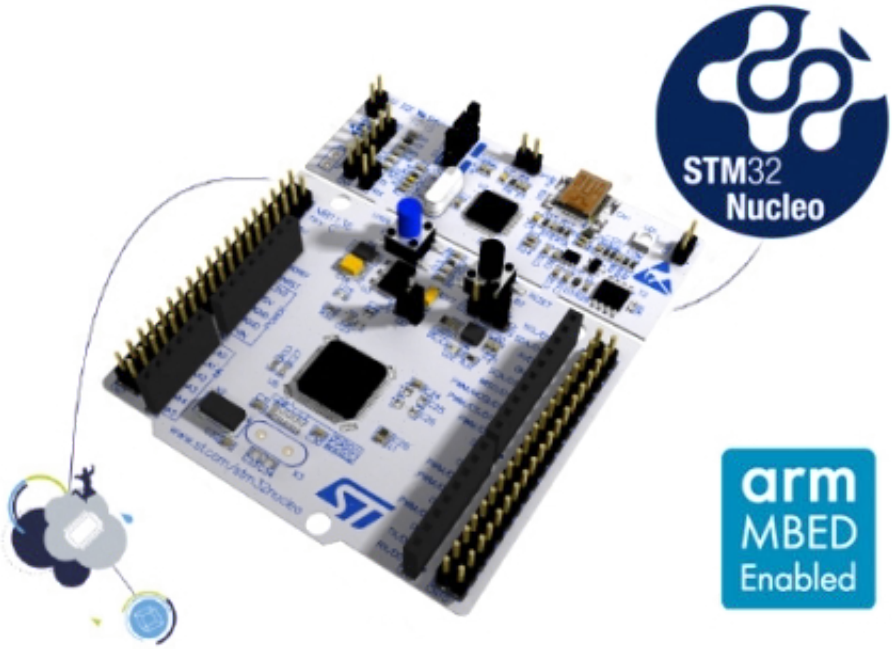
The Arduino connectivity support and ST morpho connectors make it easy to expand the functionality of the STM32 Nucleo open development platform with a wide range of specialized expansion boards to choose from.

The STM32 Nucleo board does not require separate probes as it integrates the ST-LINK/V2-1 debugger/ programmer.

The STM32 Nucleo board comes with the comprehensive STM32 software HAL library together with various packaged software examples for different IDEs (IAR EWARM, Keil MDK-ARM, STM32CubeIDE, mbed and GCC/ LLVM).

All STM32 Nucleo users have free access to the mbed online resources (compiler, C/C++ SDK and developer community) at www.mbed.org to easily build complete applications.

Figure 8. STM32 Nucleo board



3.1.2 X-NUCLEO-NFC06A1 expansion board

The X-NUCLEO-NFC06A1 NFC card reader expansion board is based on the ST25R3916 device.

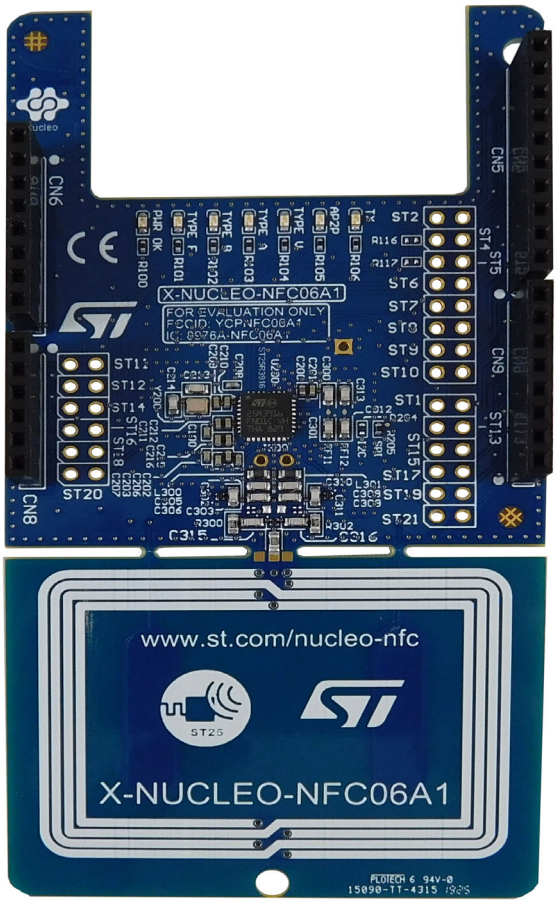
The expansion board is configured to support ISO14443A/B, ISO15693, FeliCa™ and AP2P communication.

The ST25R3916 manages frame coding and decoding in reader mode for standard applications, such as NFC, proximity and vicinity HF RFID standards. It supports ISO/IEC 14443 Type A and B, ISO/IEC 15693 (single subcarrier only) and ISO/IEC 18092 communication protocols as well as the detection, reading and writing of NFC Forum Type 1, 2, 3, 4 and 5 tags.

The on-board low power capacitive sensor performs ultra-low power wake-up without switching the reader field on and traditional inductive wake-up to select amplitude or phase measurement.

The automatic antenna tuning (AAT) technology enables operation close to metallic parts and/or in changing environments.

Figure 9. **X-NUCLEO-NFC06A1 expansion board**



3.1.3 X-NUCLEO-NFC08A1 expansion board

The X-NUCLEO-NFC08A1 NFC card reader expansion board is based on the ST25R3916B device.

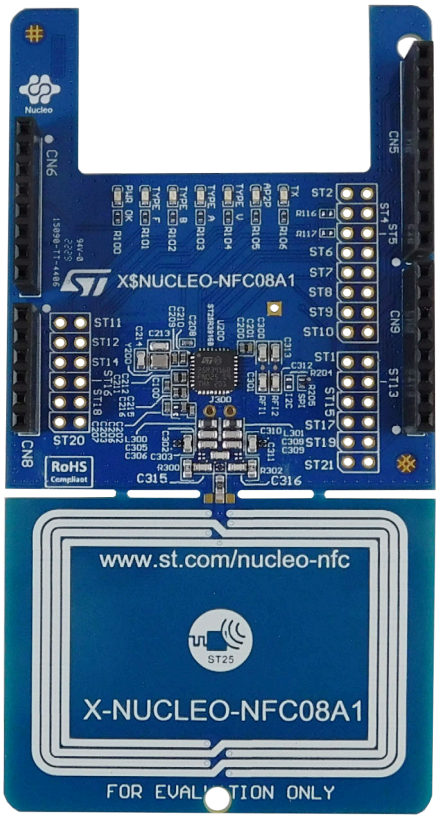
The expansion board is configured to support ISO14443A/B, ISO15693, FeliCa™, and AP2P communication.

The ST25R3916B manages frame coding and decoding in reader mode for standard applications, such as NFC, proximity, and vicinity HF RFID standards. It supports ISO/IEC 14443 type A and B, ISO/IEC 15693 (single subcarrier only) and ISO/IEC 18092 communication protocols as well as the detection, reading and writing of NFC forum type 1, 2, 3, 4, and 5 tags.

The on-board low-power capacitive sensor performs ultra-low power wake-up without switching the reader field on and traditional inductive wake-up to select amplitude or phase measurement.

The automatic antenna tuning (AAT) technology enables operation close to metallic parts and/or in changing environments.

Figure 10. **X-NUCLEO-NFC08A1 expansion board**



3.2 Software description

The following software components are needed in order to setup the suitable development environment for creating applications for the STM32 Nucleo equipped with the NFC expansion board:

- X-CUBE-NFC6: an expansion for STM32Cube dedicated to NFC applications development. The X-CUBE-NFC6 firmware and related documentation is available on www.st.com.
- Development tool-chain and Compiler. The STM32Cube expansion software supports the three following environments:
  - IAR Embedded Workbench for ARM® (EWARM) toolchain + ST-LINK
  - Keil Microcontroller Development Kit (MDK-ARM) toolchain + ST-LINK
  - STM32CubeIDE + ST-LINK

3.3 Hardware setup

The following hardware components are required:

- One STM32 Nucleo development platform (suggested order code: NUCLEO-L476RG or NUCLEO-L053R8)
- One ST25R3916/ST25R3916B high performance HF reader/NFC front end IC expansion board (order code: X-NUCLEO-NFC06A1/X-NUCLEO-NFC08A1)
- One USB type A to Mini-B USB cable to connect the STM32 Nucleo to the PC

3.4 Software setup

3.4.1 Development tool-chains and compilers

Select one of the integrated development environments (IDE) supported by the STM32Cube expansion software and read the system requirements and setup information provided by the IDE provider.

3.5 System setup

3.5.1 STM32 Nucleo and X-NUCLEO-NFC06A1 expansion board setup

The STM32 Nucleo board integrates the ST-LINK/V2-1 debugger/programmer. You can download the ST-LINK/V2-1 USB driver at STSW-LINK009.

The X-NUCLEO-NFC06A1 expansion board is easily plugged onto the STM32 Nucleo development board through the Arduino™ UNO R3 extension connector.

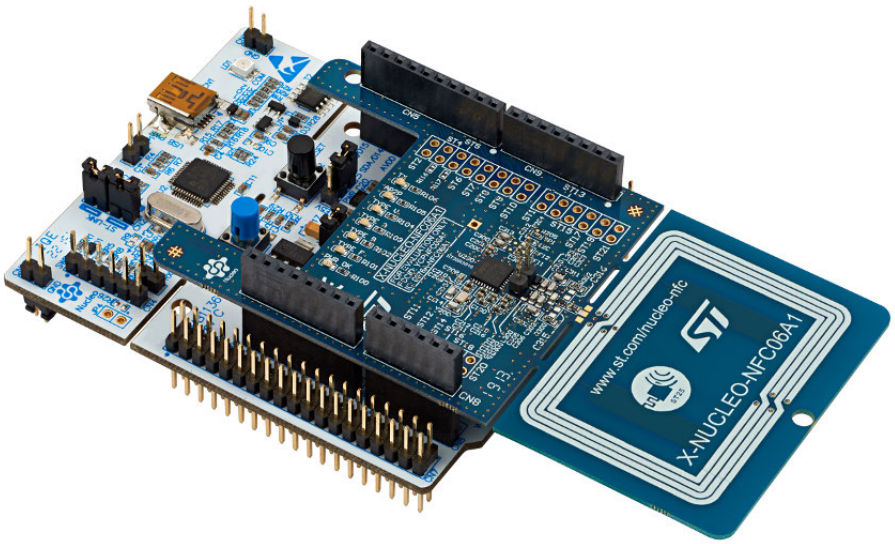
It interfaces with the STM32 microcontroller on STM32 Nucleo board through the SPI transport layer.

I²C communication is also possible, but it requires the following hardware modifications:

- solder ST2 and ST4 jumpers
- solder R116 and R117 pull-up resistors
- remove the SPI solder bridge
- put the I²C solder bridge

You have to use the pre-processor compilation flag `RFAL_USE_I2C` and rename `USE_HAL_SPI_REGISTER_CALLBACKS` by `USE_HAL_I2C_REGISTER_CALLBACKS`, if needed, to activate the I²C driver compilation.

Figure 11. X-NUCLEO-NFC06A1 expansion board plus NUCLEO-L476RG development board



3.5.2 STM32 Nucleo and X-NUCLEO-NFC08A1 expansion board setup

The STM32 Nucleo board integrates the ST-LINK/V2-1 debugger/programmer. You can download the ST-LINK/V2-1 USB driver at STSW-LINK009.

The X-NUCLEO-NFC08A1 expansion board is easily plugged onto the STM32 Nucleo development board through the Arduino™ UNO R3 extension connector.

It interfaces with the STM32 microcontroller on STM32 Nucleo board through the SPI transport layer. I²C communication is also possible.

Revision history

Table 3. **Document revision history**

Date	Version	Changes
18-Jul-2019	1	Initial release.
19-Oct-2022	2	Updated introduction, Section 2.1 Overview, Section 2.2 Architecture, Section 2.3 Folder structure, Section 2.5 Sample application, Section 3.2 Software description, Section 3.3 Hardware setup, and Section 3.5.1 STM32 Nucleo and X-NUCLEO-NFC06A1 expansion board setup. Added Section 3.1.3 X-NUCLEO-NFC08A1 expansion board and Section 3.5.2 STM32 Nucleo and X-NUCLEO-NFC08A1 expansion board setup.

Contents

List of tables

List of figures