UM2949

User manual

# STSW-ST25DV002 firmware for the ST25DV64KC-DISCO board

## Introduction

This document describes the functionalities of the STSW-ST25DV002 firmware, developed to take full advantage of the ST25DV64KC-DISCO board functionalities, working with any dynamic tag of the ST25DV-I2C Series.

In this document, the dynamic tag series from STMicroelectronics is referred to as ST25DV-I2C tags.

The ST25DV-I2C tags are integrated circuits (IC) for contactless applications that communicate with a microcontroller through an $I^2C$ interface; this makes this device a dynamic tag. On the RF side, the reader (for instance a smartphone) retrieves and updates the content of the tag when close by.

The reader communicates with the ST25DV64KC-DISCO board using the ISO 15693 protocol, while the STM32L476xx microcontroller device communicates with the ST25DV-I2C tag through an $I^2C$ bus.

With this discovery kit, the ST25DV-I2C tag is ready to receive any content, following the NFC Forum standard. This means that a smartphone is able to read it natively, without having previously installed any specific application.

A specific Fast transfer mode allows the user to transfer data directly to the microcontroller, without using the embedded EEPROM.

# 1 General information

This document applies to the STM32L476xx Arm®-based microcontroller devices.

*Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.*

# 2 List of acronyms and notational conventions

## 2.1 List of acronyms

Table 1. Acronyms

| Acronym | Definition |
|---|---|
| APB | Advanced peripheral bus |
| AAR | Android™ application record |
| BLE | Bluetooth® Low Energy |
| CAN | Controller area network |
| GPS | Global positioning system |
| GPO | General purpose output |
| H2R | Host to reader |
| IEC | International electrotechnical commission |
| ISO | International standards organization |
| MCU | Micro controller unit (microcontroller) |
| NFC | Near field communication |
| OOB | Out of band |
| R2H | Reader to host |
| RF | Radio frequency |
| RFID | Radio frequency identification |
| RISC | Reduced instruction set computer |
| SPI | Serial peripheral interface |
| URI | Uniform resource identifier |
| URL | Uniform resource locator |
| USB | Universal serial bus |

## 2.2 Number convention and notation

The following conventions and notations apply in this document unless otherwise stated:

- Binary numbers are represented by strings of 0 and 1 digits starting with the most significant bit (MSB) on the left followed by the least significant bit (LSB) on the right, and "0b" added at the beginning. Example: 0b11110101.
- Hexadecimal numbers are represented by using numbers 0 to 9 and characters A to F, and adding "0x" at the beginning. The MSB is shown on the left and the LSB on the right. Example: 0xF5.
- Decimal numbers are represented without any leading character. Example: 245.

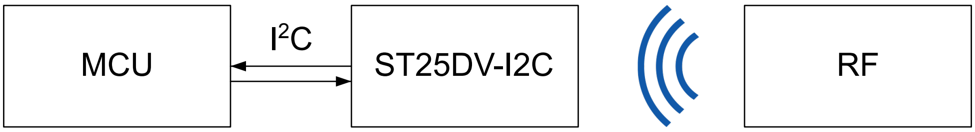
# 3 Overview

## 3.1 ST25DV-I2C tag

The ST25DV-I2C tag is a dynamic tag IC for contactless applications (ISO 15693). It manages the RF communication with a reader. It also includes frame coding, RF modulation and manages the anti-collision process.

The ST25DV-I2C tag works as an NFC Forum Type 5 tag, supporting: detection, read and write operations. As shown in Figure 1, this device communicates with a reader without any external controller.

**Figure 1. Communication scheme**



## 3.2 STM32L476xx microcontroller devices

The STM32L476xx microcontroller devices are ultra-low-power microcontrollers based on the high-performance Arm® 32-bit RISC core operating at a frequency of up to 80 MHz. The Arm® Cortex®-M4 core features a floating point unit (FPU) single precision, which supports all Arm® single-precision data-processing instructions and data types. It also implements a full set of DSP instructions and a memory protection unit (MPU) to enhance application security.

The STM32L476xx microcontroller devices embed high-speed memories (up to 1 Mbytes of Flash memory, up to 128 Kbytes of SRAM), a flexible external memory controller (FSMC) for static memories (for devices with packages of 100 pins and more), a Quad-SPI Flash memory interface (available on all packages) and an extensive range of enhanced I/Os and peripherals connected to two APB buses, two AHB buses and a 32-bit multi-AHB bus matrix.

The STM32L476xx microcontroller devices embed several protection mechanisms for embedded Flash memory and SRAM: readout protection, write protection, proprietary code readout protection and firewall.

These devices offer up to three fast 12-bit ADCs (5 Msps), two comparators, two operational amplifiers, two DAC channels, an internal voltage reference buffer, a low-power RTC, two general-purpose 32-bit timers, two 16-bit PWM timers dedicated to motor control, seven general-purpose 16-bit timers, and two 16-bit low-power timers. The STM32L476xx microcontroller devices support four digital filters for external sigma delta modulators (DFSDM).

In addition, up to 24 capacitive sensing channels are available. The devices also embed an integrated LCD driver 8x40 or 4x44, with internal step-up converter.

They also feature standard and advanced communication interfaces:

- Three I²Cs
- Three SPIs
- Three USARTs, two UARTs and one Low-power UART.
- Two SAIs (serial audio interfaces)
- One SDMMC
- One CAN
- One USB OTG full-speed
- One SWPMI (single wire protocol master interface)

For full details of the STM32L476xx microcontroller device specifications, refer to *www.st.com*.

## 3.3 ST25DV64KC-DISCO boards

The ST25DV64KC-DISCO is an evaluation kit, which allows the user to evaluate the performance of the ST25DV-I2C tag dynamic tag. The kit is composed of two boards, namely a motherboard (see Figure 2) and the ST25DV-I2C tag daughter board (see Figure 3).

### 3.3.1 ST25DV64KC-DISCO motherboard

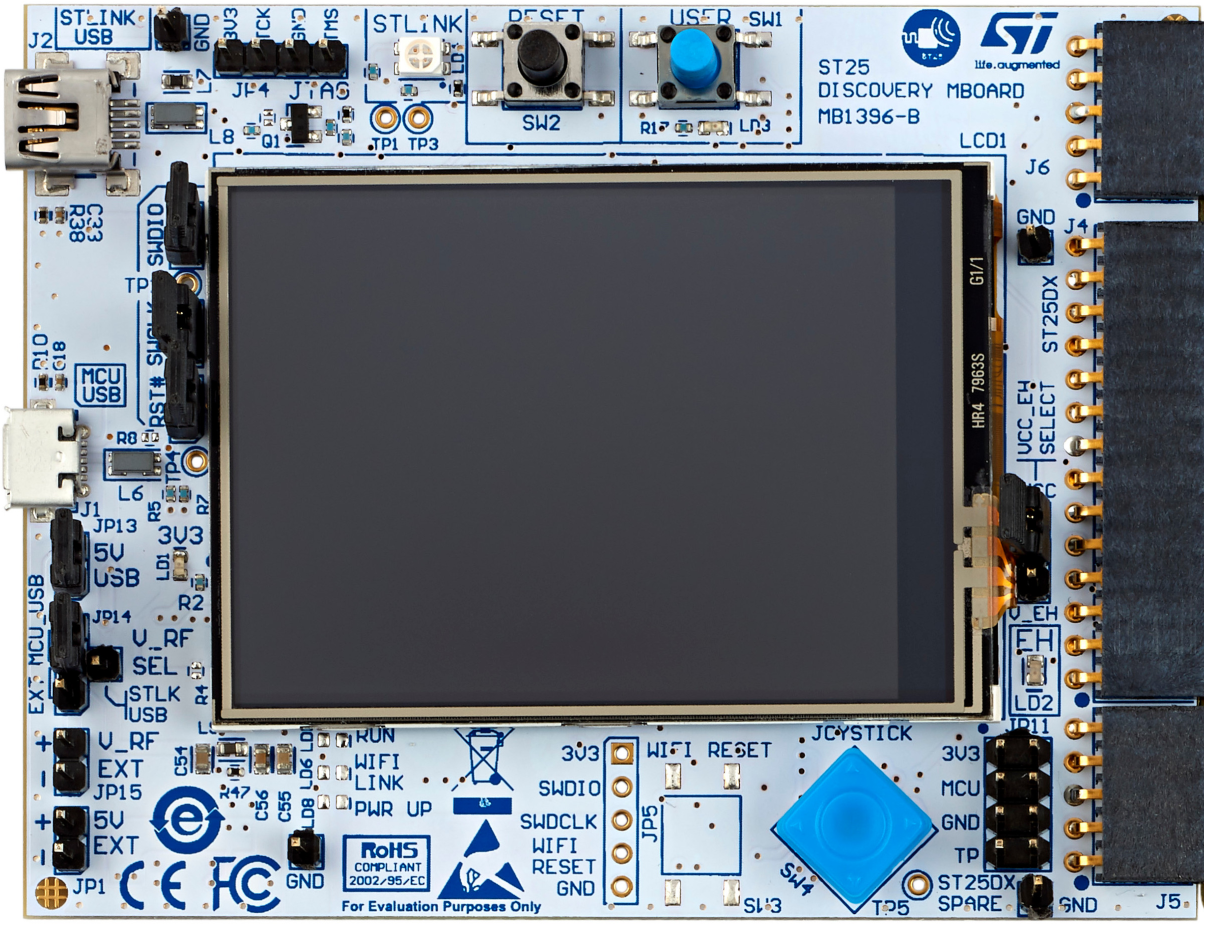
The ST25DV64KC-DISCO (illustrated in Figure 2) is powered through the USB bus (micro and mini connectors) and no external power supply is required.

This motherboard embeds the STM32L476VG microcontroller and different peripherals:

- LCD display and touchscreen to interface with the user
- USB connectors to connect to a PC (mini-USB for the ST-Link debugger and micro-USB available for the user application)
- Optional modules: Wi-Fi® and BLE to connect with a smart phone

The connectors on the right side of the board are dedicated to the daughter boards that embed the NFC tags.

Figure 2. **ST25DV64KC-DISCO motherboard MB1396**

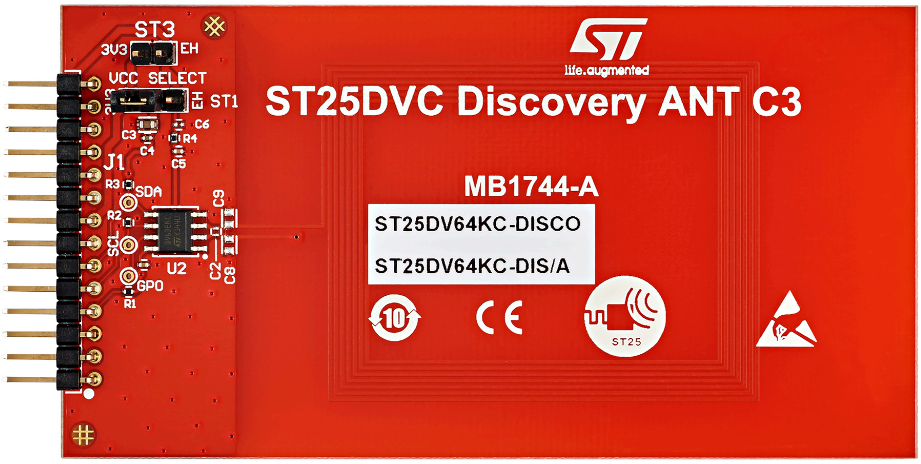


### 3.3.2 ST25DV-I2C tag daughter board

The ST25DVKC Discovery ANT C3 board (illustrated in Figure 3) embeds the ST25DV64KC device and a 40 mm x 23 mm, 13.56 MHz double layer inductive etched antenna (no need for any tuning components).

The ST25DV64KC communicates with the STM32L476VG 32-bit MCU via the I²C bus.

Figure 3. ST25DVKC Discovery ANT C3 board



# 4 Firmware description

## 4.1 Prerequisite

The ST25DV-I2C tag is a dynamic tag, whose content and configuration is driven by both a microcontroller (through I²C) and a reader through RF.

To benefit from the ST25DV64KC-DISCO demonstration kit, the user must have one of the following:

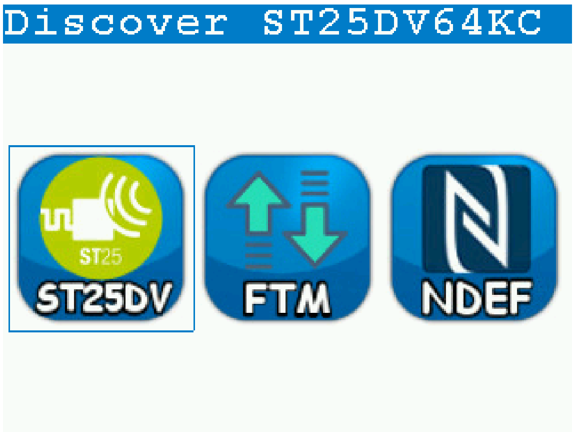
- A smart phone with the NFC enabled, combined with the ST25 NFC App installed on it. Refer to *ST25 Android™ NFC tap application* (UM2131), available on *www.st.com* for more information.
- An NFC reader, such as the CR95HF distributed with the M24LR discovery kit (M24LR-DISCOVERY). To take full advantage of the ST25DV-I2C tag features, use CR95HF transceiver board firmware version 3.7.0 or higher, combined with the ST25PC-NFC software. Refer to *Software toolbox for NFC tags* user manual (UM2444), available on *www.st.com*.

The ST25DV-I2C tag is fully compliant with the NFC Forum Type 5 standard. As this standard is recently introduced, some features may not be fully supported by older smart phones.

## 4.2 Main menu

The main menu (Figure 4) is composed of three icons, used to access sub-menus.

Figure 4. **ST25DV64KC-DISCO main menu display**



DT52442V2

Each item allows the user to enter a sub-menu containing a set of use cases as outlined. The sub-menus are summarized in Table 2:

Table 2. **Main menu item definition**

| Discover ST25DV-I2C tag | Fast transfer mode demonstration | NFC NDEF demonstrations |
|---|---|---|
| ST25<br>ST25DV | FTM | NDEF |

*Note: If the Wi-Fi® and/or the Bluetooth® Low Energy modules have been added to the board, the corresponding icons are available to access the Wi-Fi® and Bluetooth® pairing demonstrations.*

To select a category, touch the icon on the screen. To start a demonstration, touch the desired demonstration icon on the screen (see description below). A long touch on an icon displays the description of the corresponding demonstration(s) in the top line.

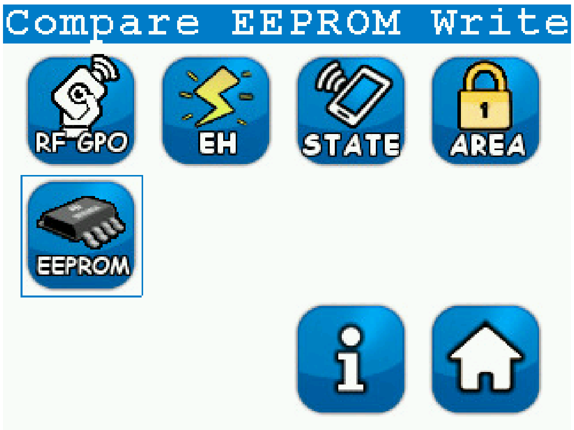
As the purpose of this document is to illustrate the firmware behavior, the descriptions are detailed from the microcontroller point of view.

### 4.2.1 Discover ST25DV-I2C tag menu

As shown in Figure 5, the "Discover ST25DV-I2C tag" menu proposes several demonstrations of the ST25DV-I2C tag specific features, among them:

- GPO interrupts in Section 4.2.2
- Energy harvesting in Section 4.2.3
- ST25DV-I2C tag states in Section 4.2.4
- Multi-areas and passwords in Section 4.2.5

**Figure 5. ST25DV64KC-DISCO features main display**


Compare EEPROM Write

### 4.2.2 RF GPO interrupt demonstration

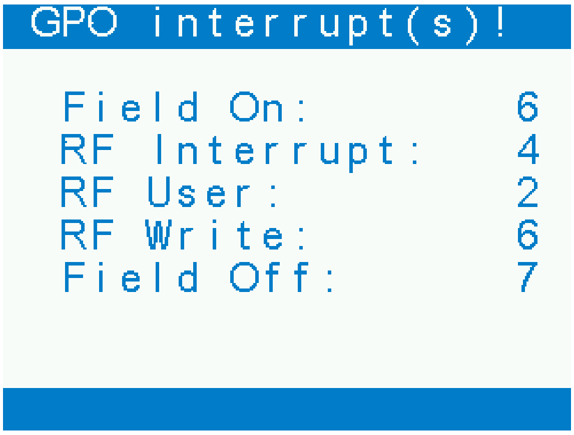
The ST25DV-I2C tag has a GPO that is used to send interrupts to the microcontroller.

Several kinds of event on the RF side may be configured to trigger the GPO interrupt. In this demonstration all possible interrupt sources are enabled, except for the "RF activity", which occurs too often for reasonable readability when using a smart phone as a reader.

To start the demonstration, select the "GPO control demo" menu, this displays the instructions until it detects a GPO interrupt.

As soon as the first GPO interrupt is received, the LCD displays the list of sources of received interrupt, and the number of each interrupt occurrences (see Figure 6).

Figure 6. **ST25DV64KC-DISCO interrupt generation display**



```
GPO interrupt(s)!

  Field On:         6
  RF Interrupt:     4
  RF User:          2
  RF Write:         6
  Field Off:        7
```

The event types and the way to generate them are listed below:

- Field on: place a reader near the ST25DV-I2C tag
- Field off: move a reader away from the ST25DV-I2C tag
- EEPROM written: modify the content of the ST25DV-I2C tag
- RF interrupt: requires the use of a specific reader instruction, that is sent either by a smart phone running the ST25 NFC app, or by a reader (see the instructions in Smart phone GPO management and Generating the RF and RF user interrupts using the ST25PC-NFC software with the CR95HF sections, respectively, for Android™ ST25 NFC application and for the CR95HF reader used with the ST25PC-NFC software)
- RF user: requires the use of a specific reader command, that is sent either by a smart phone running the ST25 NFC app or by a reader (see the instructions in Smart phone GPO management and Generating the RF and RF user interrupts using the ST25PC-NFC software with the CR95HF sections, respectively, for the Android™ ST25 NFC application and for the CR95HF reader used with the ST25PC-NFC software)

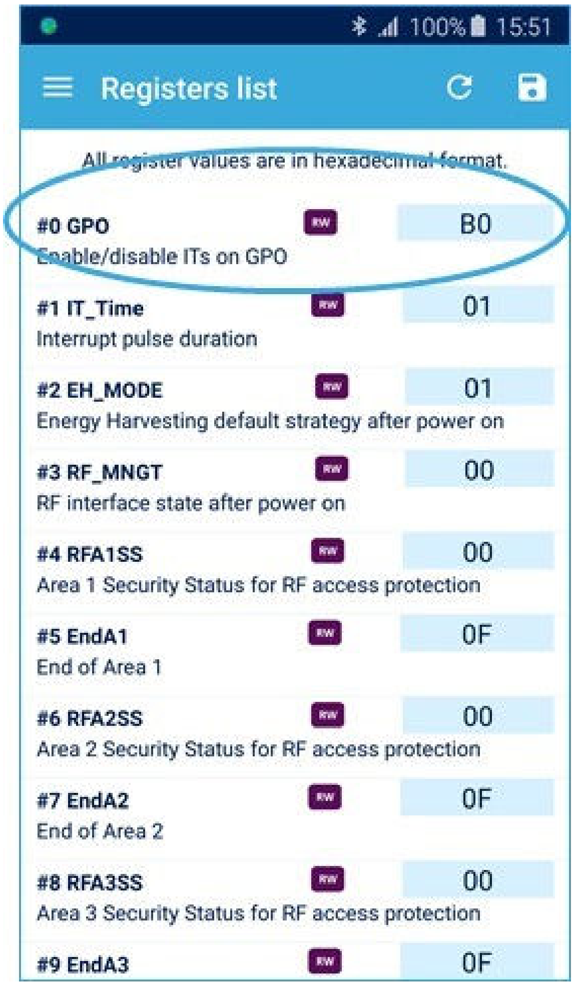
To start the Android ST25 NFC application, read the ST25DV-I2C tag with a smart phone (after entering the GPO demonstration), which automatically opens the required application (or propose the installation of the application). The automatic application selection is done using an NDEF with an AAR.

*Note: The mailbox events are also used to trigger the GPO interrupts. They are not mentioned here, as they would require placing the ST25DV-I2C tag in the Mailbox mode.*

#### Smart phone GPO management

The ST25 NFC application handles the RF interrupt and the RF user interrupt using register fields information (see Figure 7). Do not forget to present the configuration password before any register change.

Figure 7. Smart phone GPO register configuration



Registers list

All register values are in hexadecimal format.

| | | |
|---|---|---|
| **#0 GPO**<br>Enable/disable ITs on GPO | RW | B0 |
| **#1 IT_Time**<br>Interrupt pulse duration | RW | 01 |
| **#2 EH_MODE**<br>Energy Harvesting default strategy after power on | RW | 01 |
| **#3 RF_MNGT**<br>RF interface state after power on | RW | 00 |
| **#4 RFA1SS**<br>Area 1 Security Status for RF access protection | RW | 00 |
| **#5 EndA1**<br>End of Area 1 | RW | 0F |
| **#6 RFA2SS**<br>Area 2 Security Status for RF access protection | RW | 00 |
| **#7 EndA2**<br>End of Area 2 | RW | 0F |
| **#8 RFA3SS**<br>Area 3 Security Status for RF access protection | RW | 00 |
| **#9 EndA3** | RW | 0F |

The ST25 NFC application generates the RF interrupt and the RF user interrupt using features located in the Tools fragment in association with the RF GPO interrupt demonstration.

### Generating the RF and RF user interrupts using the ST25PC-NFC software with the CR95HF

These interrupts are generated by any RF reader or Android™ smart phone.

Using the CR95HF IC and ST25PC-NFC software, the user is able to send interrupts or drive the GPO (see Figure 8 and Figure 9)

- Select the ST25DV-I2C tag product
- Select "ENERGY HARVESTING" and GPO tab
- Click on "Send Interrupt Generated by GPO" to send an interrupt on GPO line
- Click on GPO set or GPO reset to be able to drive the GPO line.

For additional information about CR95HF PC software refer to the *CR95HF development software* user manual (UM1084), available on *www.st.com*.

Figure 8. ST25DV-I2C tag contextual menu

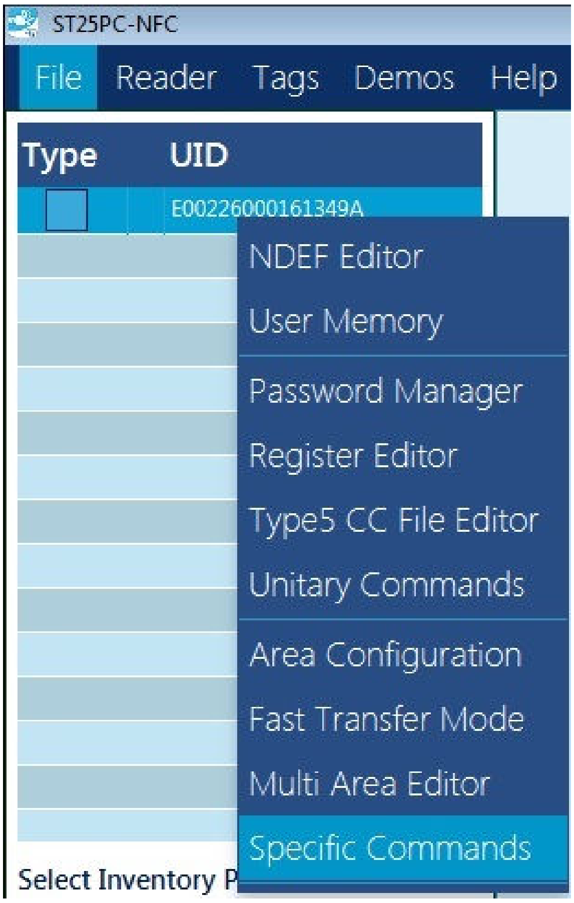
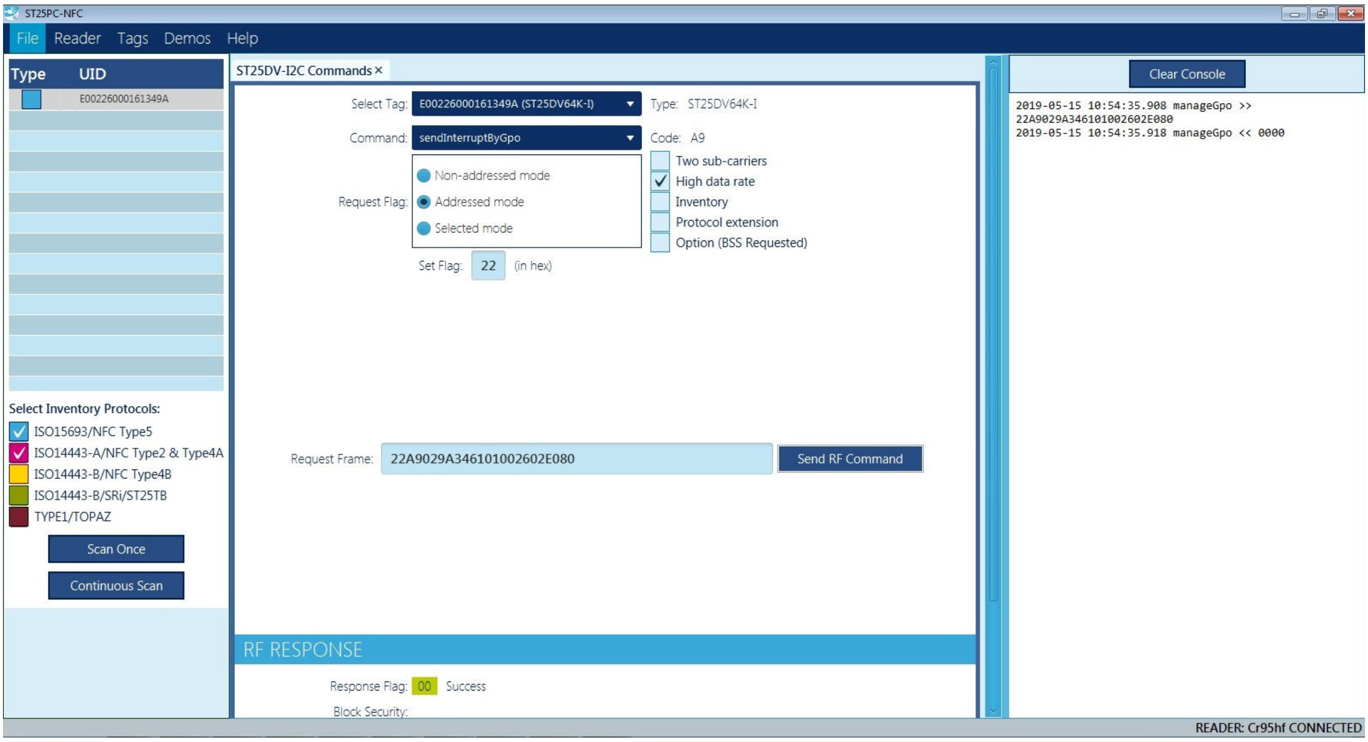


Figure 9. PC software ST25DV-I2C tag special commands interface



Select Inventory Protocols:

- ☑ ISO15693/NFC Type5
- ☑ ISO14443-A/NFC Type2 & Type4A
- □ ISO14443-B/NFC Type4B
- □ ISO14443-B/SRi/ST25TB
- □ TYPE1/TOPAZ

Select Tag: E00226000161349A (ST25DV64K-I)

Type: ST25DV64K-I

Command: sendInterruptByGpo

Code: A9

Request Flag:
○ Non-addressed mode
● Addressed mode
○ Selected mode

□ Two sub-carriers
☑ High data rate
□ Inventory
□ Protocol extension
□ Option (BSS Requested)

Set Flag: 22 (in hex)

Request Frame: 22A9029A346101002602E080

Response Flag: 00 Success

Block Security:

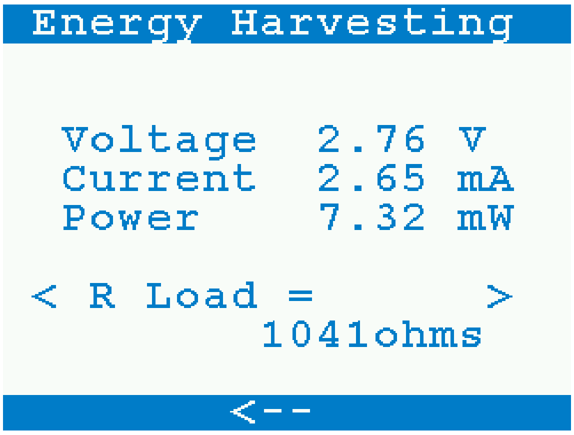
```
2019-05-15 10:54:35.908 manageGpo >>
22A9029A346101002602E080
2019-05-15 10:54:35.918 manageGpo << 0000
```

### 4.2.3 ST25DV-I2C tag energy harvesting demonstration

The ST25DV-I2C is able to harvest the energy provided by the RF to power other devices.

This demonstration displays the voltage, current and power provided by the ST25DV-I2C (see Figure 10).

Figure 10. ST25DV64KC-DISCO energy harvesting measurement display



```
Energy Harvesting

  Voltage   2.76 V
  Current   2.65 mA
  Power     7.32 mW

< R Load =       >
        1041ohms

      <--
```

DT52447V2

A LED is also turned on to simulate the effect of the energy harvesting when the RF field is approaching or when it leaves.

The energy harvesting pin is connected to a load with which is used to measure the current delivered by the tag. This load is configurable (shown in Figure 10) from 540 Ω to 4320 Ω. The objective is to simulate the current consumed by additional external circuitry.

### 4.2.4 ST25DV-I2C tag states demonstration

This menu illustrates the possibility to change the ST25DV-I2C tag state, for example the power save demonstration. This demonstration proposes three different modes:

1. RF disabled mode: RF commands are interpreted but not executed. The ST25DV-I2C tag responds with the error code 0x0F.
2. Sleep mode: all RF communications are disabled, the RF interface does not interpret the commands, but minimizes consumption of RF interface.
3. Low-power down: the ST25DV-I2C tag is fully inactive from Host - I$^2$C side, but the memory content is accessible on the RF side.

For all above cases, the demonstration consists in writing an NDEF with the ST25 URL, and changing the ST25DV-I2C tag state. A message is displayed that invites the user to try to read the content of the ST25DV-I2C tag, and check that it is not possible.

The content of the ST25DV-I2C tag can be read again as soon as the demonstration screen is passed.

*Note: For the Low-power mode, the ST25DXSPARE jumper must be present on the ST25DV64KC-DISCO board's MCU pin, and a 12-pin package must be used for the ST25DV-I2C tag.*

### 4.2.5 ST25DV-I2C tag multi area and password demonstration

The ST25DV-I2C tag defines up to four different areas in memory. Each area may have a custom security level, requiring one of the three passwords to be provided in order to read and/or write the memory.

This demonstration sets two different areas:

1. "Area 1" contains an NDEF with a vCard. This area is readable by anyone, but can only be written after presenting "password 1".
2. "Area 2" contains an NDEF with a different vCard. This area cannot be read without the "password 1", and cannot be written even after the password presentation.

By default, all the passwords are set to 0000 0000 0000 0000. These values are only updated from the RF side.

To execute the demonstration, the user must first read the ST25DV-I2C tag without any specific application. The phone displays the vCard stored in "Area 1".

Then the user opens the ST25 NFC application, and taps the tag again, the application detects two areas, and displays the "Area "1 vCard.

The user may try to write a different NDEF to the "Area 1", and check that the write fails unless "password 1" is presented.

If the user selects the "Area 2" NDEF, the application requests "password 1" before displaying the "Area 2" vCard.

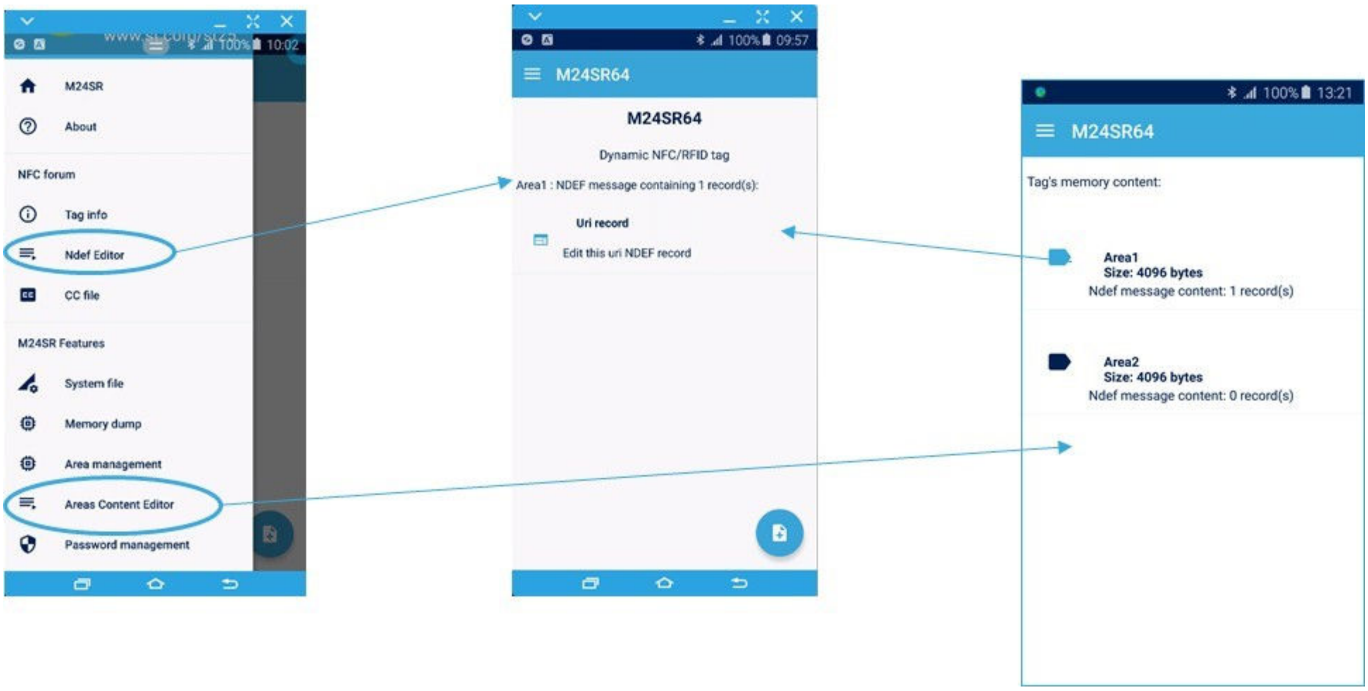
The user may try to write a different NDEF in "Area 2", this does not work even after presenting "password 1". This is expected, as the security level of "Area 2" prevents any write access to the memory.

#### Smart phone and multi areas

When the tag is configured in multi areas, the ST25 NFC application then handles the areas and presents areas and content to the user. The user then chooses the desired area by selecting the corresponding menu in the application drawer. This is illustrated in Figure 11.

In case special protection is needed, the application displays security status information. On click, a popup is used to present the area password needed to access area content.

Figure 11. **Smart phone and area display**



M24SR64

Dynamic NFC/RFID tag

Area1 : NDEF message containing 1 record(s):

Uri record

Edit this uri NDEF record

Tag's memory content:

Area1
Size: 4096 bytes
Ndef message content: 1 record(s)

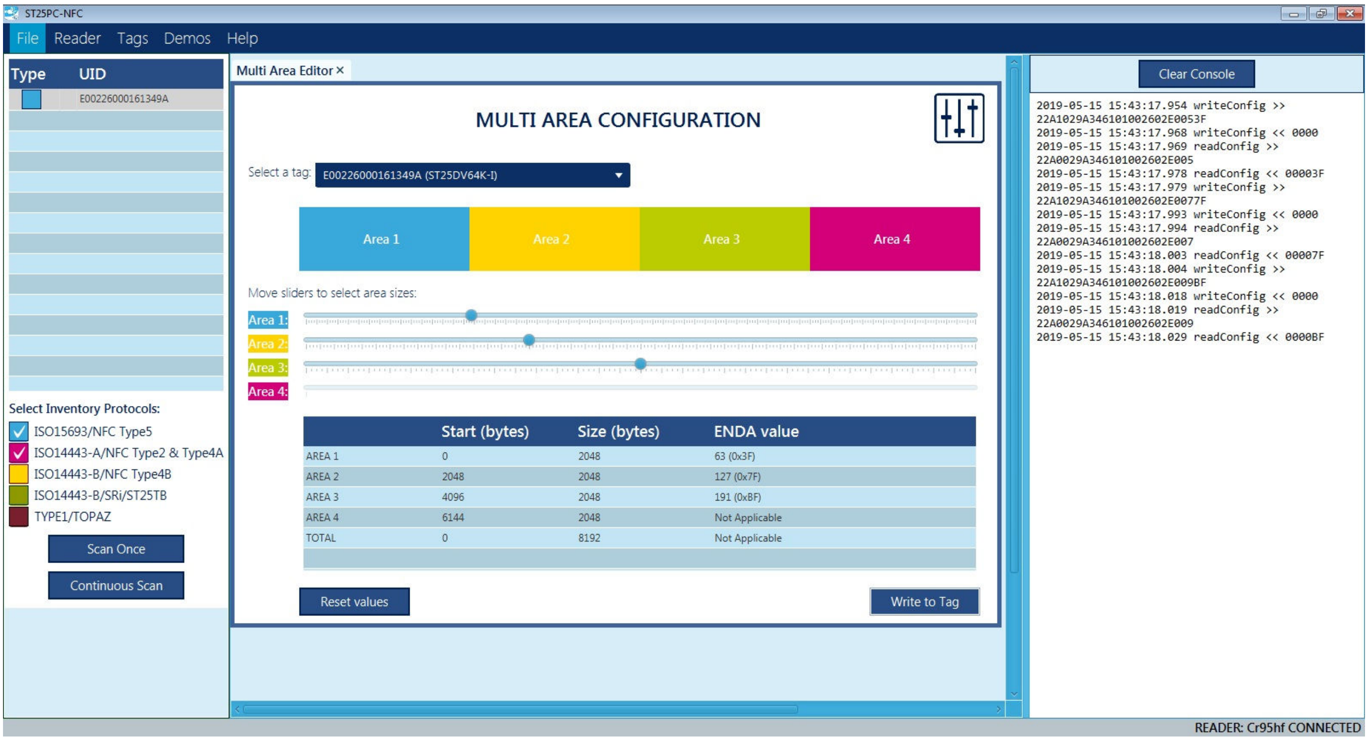
Area2
Size: 4096 bytes
Ndef message content: 0 record(s)

### ST25PC-NFC software and multi areas

The ST25PC-NFC software is used to create up to four areas on the ST25DV-I2C tag EEPROM:

1. From the Inventory side panel, right-click on the ST25DV-I2C tag tag (see Figure 8).
2. In the contextual menu, select the "Multi Area Editor" item.
3. Move the sliders for each area until the desired value is reached (see Figure 12 for an example of an EEPROM cut into four equal areas of 2048 bytes segments).
4. The start address, size of the area and ENDA register values are updated in real time inside the bottom table.

Figure 12. **ST25DV-I2C tag multi area configuration interface of the ST25PC-NFC software**



Select a tag: E00226000161349A (ST25DV64K-I)

Select Inventory Protocols:

- ☑ ISO15693/NFC Type5
- ☑ ISO14443-A/NFC Type2 & Type4A
- □ ISO14443-B/NFC Type4B
- □ ISO14443-B/SRi/ST25TB
- □ TYPE1/TOPAZ

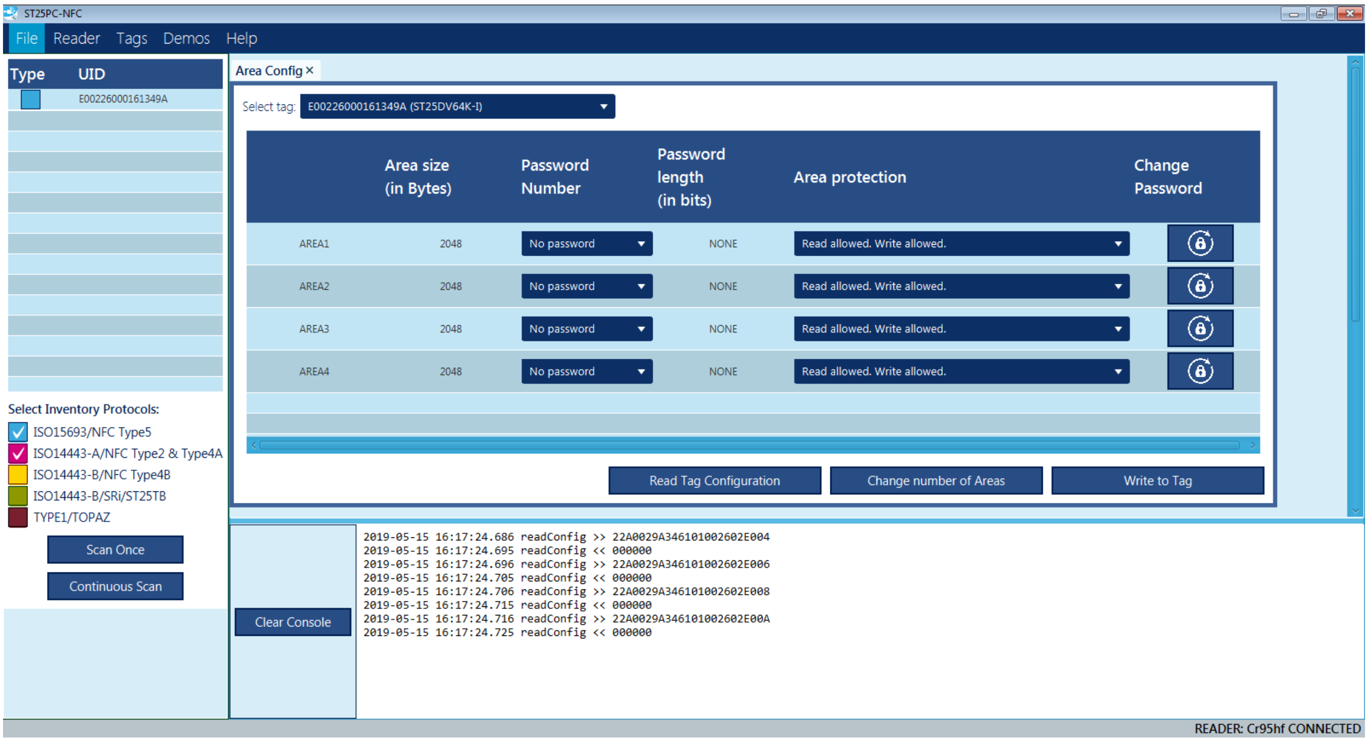
| | Start (bytes) | Size (bytes) | ENDA value |
|---|---|---|---|
| AREA 1 | 0 | 2048 | 63 (0x3F) |
| AREA 2 | 2048 | 2048 | 127 (0x7F) |
| AREA 3 | 4096 | 2048 | 191 (0xBF) |
| AREA 4 | 6144 | 2048 | Not Applicable |
| TOTAL | 0 | 8192 | Not Applicable |

```
2019-05-15 15:43:17.954 writeConfig >>
22A1029A346101002602E0053F
2019-05-15 15:43:17.968 writeConfig << 0000
2019-05-15 15:43:17.969 readConfig >>
22A0029A346101002602E005
2019-05-15 15:43:17.978 readConfig << 00003F
2019-05-15 15:43:17.979 writeConfig >>
22A1029A346101002602E0077F
2019-05-15 15:43:17.993 writeConfig << 0000
2019-05-15 15:43:17.994 readConfig >>
22A0029A346101002602E007
2019-05-15 15:43:18.003 readConfig << 00007F
2019-05-15 15:43:18.004 writeConfig >>
22A1029A346101002602E009BF
2019-05-15 15:43:18.018 writeConfig << 0000
2019-05-15 15:43:18.019 readConfig >>
22A0029A346101002602E009
2019-05-15 15:43:18.029 readConfig << 0000BF
```

Access rights for each area of the tag are set from the "Area configuration" screen (see Figure 13):

1. From the table, select the area to protect (up to four areas).
2. In the "Area protection" column, set one of four options for read and write operations (only three are available for "Area 1").
3. Additionally, the password number can be set for each area.

Figure 13. **Area configuration interface of the ST25PC-NFC software**



Select tag: E00226000161349A (ST25DV64K-I)

Select Inventory Protocols:

- ☑ ISO15693/NFC Type5
- ☑ ISO14443-A/NFC Type2 & Type4A
- □ ISO14443-B/NFC Type4B
- □ ISO14443-B/SRi/ST25TB
- □ TYPE1/TOPAZ

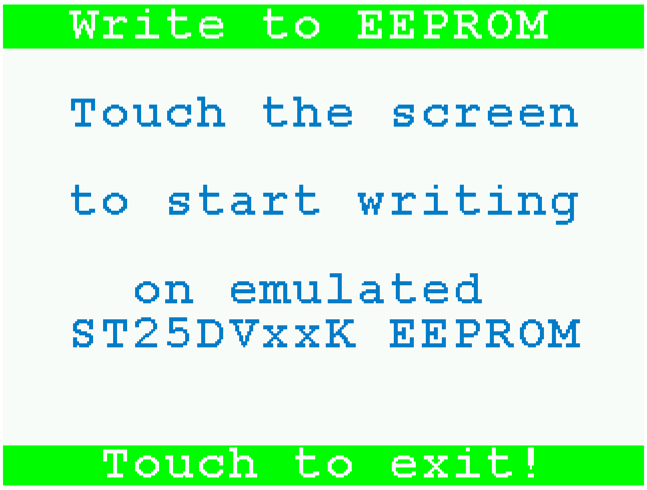
| | Area size (in Bytes) | Password Number | Password length (in bits) | Area protection | Change Password |
|---|---|---|---|---|---|
| AREA1 | 2048 | No password | NONE | Read allowed. Write allowed. | 🔒 |
| AREA2 | 2048 | No password | NONE | Read allowed. Write allowed. | 🔒 |
| AREA3 | 2048 | No password | NONE | Read allowed. Write allowed. | 🔒 |
| AREA4 | 2048 | No password | NONE | Read allowed. Write allowed. | 🔒 |

```
2019-05-15 16:17:24.686 readConfig >> 22A0029A346101002602E004
2019-05-15 16:17:24.695 readConfig << 000000
2019-05-15 16:17:24.696 readConfig >> 22A0029A346101002602E006
2019-05-15 16:17:24.705 readConfig << 000000
2019-05-15 16:17:24.706 readConfig >> 22A0029A346101002602E008
2019-05-15 16:17:24.715 readConfig << 000000
2019-05-15 16:17:24.716 readConfig >> 22A0029A346101002602E00A
2019-05-15 16:17:24.725 readConfig << 000000
```

### 4.2.6 ST25DV-I2C tag compare EEPROM write

This menu shows the difference between writing to the 4-byte ST25DVxxK EEPROM page and to the 16-byte ST25DVxxKC EEPROM page. When entering this menu the following screens show the difference by emulating an ST25DVxxKC EEPROM page write and performing the actual write on the ST25DVxxKC EEPROM.

Figure 14. ST25DVxxK emulated EEPROM write screen



Write to EEPROM

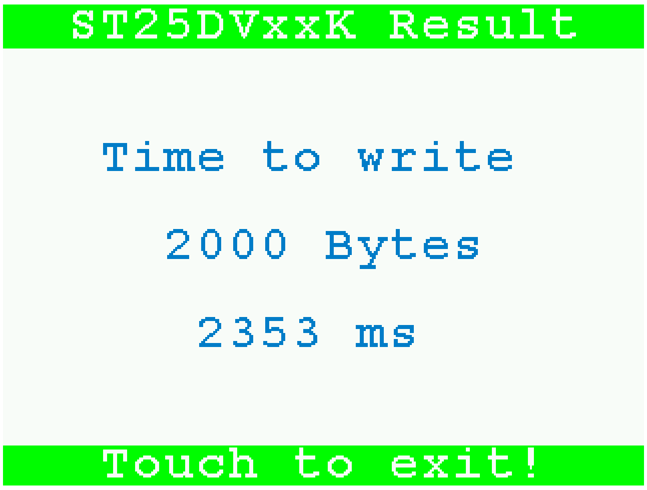
Touch the screen

to start writing

on emulated
ST25DVxxK EEPROM

Touch to exit!

Figure 15. ST25DVxxK emulated EEPROM write results



ST25DVxxK Result

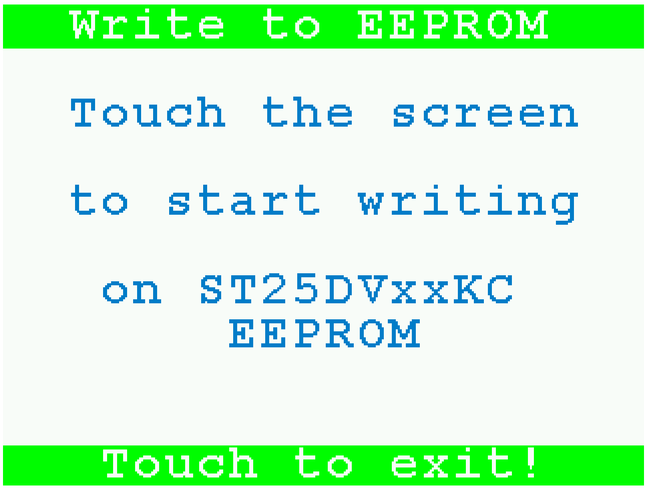
Time to write

2000 Bytes

2353 ms

Touch to exit!

Figure 16. ST25DVxxKC EEPROM write screen



Write to EEPROM

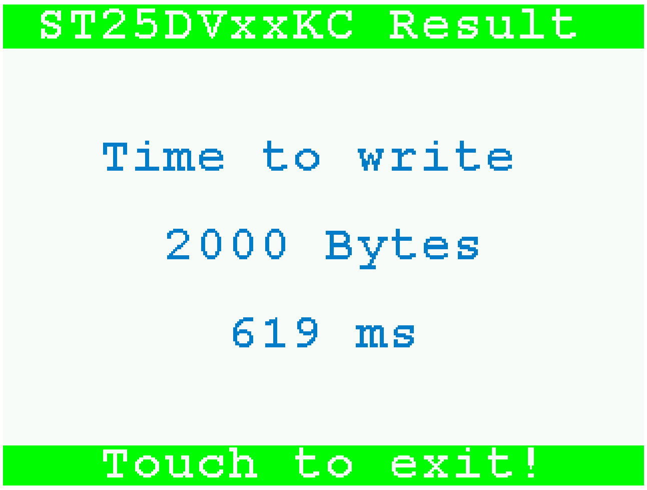
Touch the screen

to start writing

on ST25DVxxKC
EEPROM

Touch to exit!

Figure 17. ST25DVxxKC EEPROM write results



ST25DVxxKC Result

Time to write

2000 Bytes

619 ms

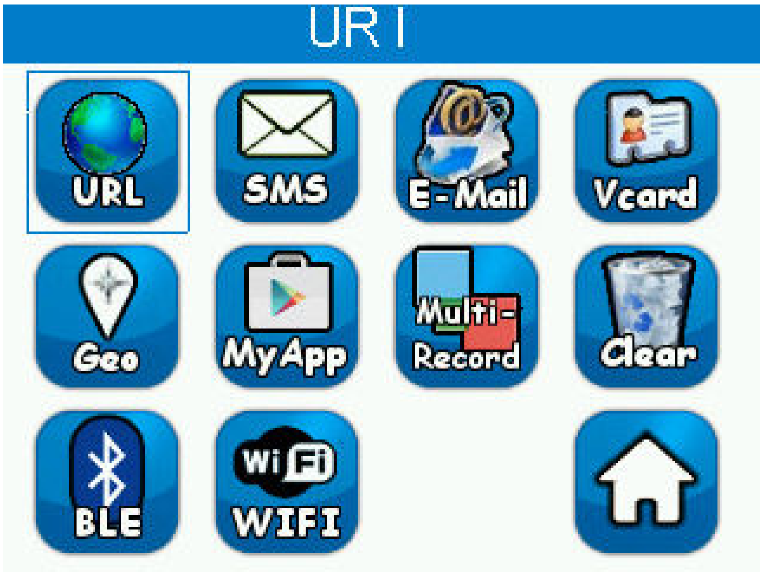
Touch to exit!

### 4.2.7 NFC NDEF demonstration menu

The NFC NDEF demonstrations menu (see Figure 18) proposes several demonstrations concerning NDEF messages:

- URI NDEF message (URL and phone number)
- SMS NDEF message
- Email NDEF message
- vCard NDEF message
- Geo-location NDEF message
- Custom (MyApp) NDEF message
- Multi record NDEF message
- Bluetooth® Low Energy OOB NDEF message (requires an additional Bluetooth® Low Energy module)
- Wi-Fi® OOB NDEF message (requires an additional Wi-Fi® module).

Figure 18. **ST25DV64KC-DISCO NFC NDEF demonstrations menu display**

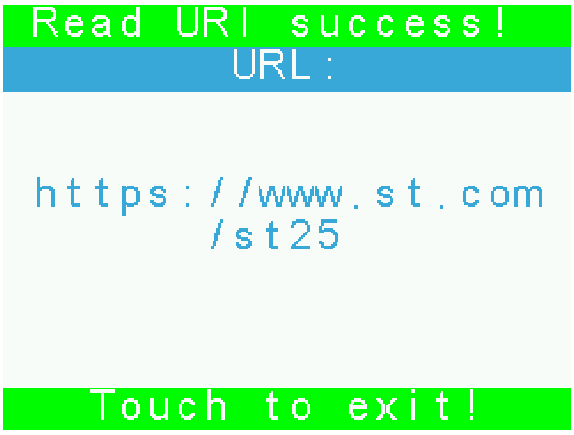


#### URI demonstration

This menu (see Figure 19) shows how to manage URI content in the ST25DV-I2C tag.

With the MCU, the user:

- Stores an NDEF message containing a URL or a phone number (it is then be read with an NFC reader or a smart phone).
- Reads any URI that is stored in the ST25DV-I2C tag. The display shows the content of the URI.



Read URI success!

URL:

https://www.st.com
/st25

Touch to exit!

Figure 19. ST25DV64KC-DISCO read URI display

*Note: If the MCU reads the ST25DV-I2C tag memory, it does not contain a URI, an error message is displayed.*

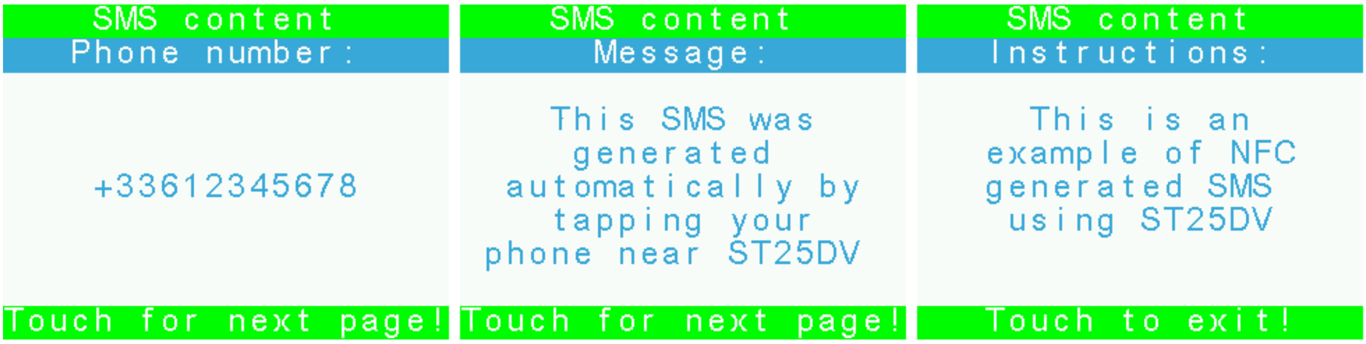
#### Smart phone and URI NDEF

Refer to Reading and writing NDEF on a smart phone with the Android™ ST25 application section.

#### SMS demonstration

This menu indicates how to manage SMS content in the ST25DV-I2C tag.
With the MCU, the user:

- Stores an NDEF message containing an SMS (it is then be read with an NFC reader or with a smart phone)
- Reads any SMS that is stored in the ST25DV-I2C tag memory (the SMS is displayed on the screen).



SMS content
Phone number:

+33612345678

Touch for next page!

SMS content
Message:

This SMS was
generated
automatically by
tapping your
phone near ST25DV

Touch for next page!

SMS content
Instructions:

This is an
example of NFC
generated SMS
using ST25DV

Touch to exit!

Figure 20. ST25DV64KC-DISCO read SMS content display

*Note: If the MCU reads the ST25DV-I2C tag memory, it does not contain an SMS, an error message is displayed.*

#### Smart phone and SMS NDEF

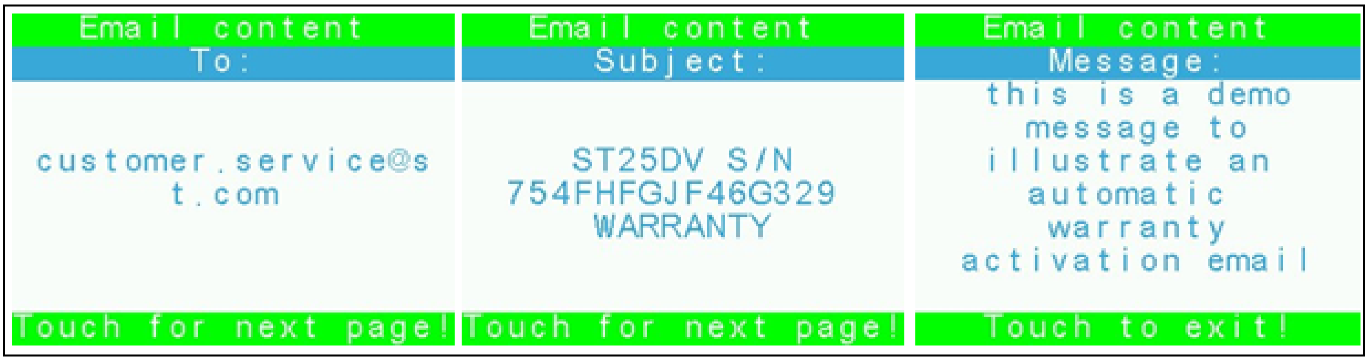
Refer to Reading and writing NDEF on a smart phone with the Android™ ST25 application section.

#### Email demonstration

The menu illustrated in Figure 21 indicates how to manage e-mail content in the ST25DV-I2C tag.
With the MCU, the user:

- Stores an NDEF message containing an e-mail (it is then be read with an NFC reader or with a smart phone).
- Reads any e-mail that is stored in the ST25DV-I2C tag memory. The content of the e-mail is displayed on the screen.

Figure 21. **ST25DV64KC-DISCO read Email content display**



Email content
To:
customer.service@st.com
Touch for next page!

Email content
Subject:
ST25DV S/N 754FHFGJF46G329 WARRANTY
Touch for next page!

Email content
Message:
this is a demo message to illustrate an automatic warranty activation email
Touch to exit!

*Note: If the MCU reads the ST25DV-I2C tag memory, it does not contain an e-mail, an error message is displayed.*

**Smart phone and e-mail NDEF**

Refer to Reading and writing NDEF on a smart phone with the Android™ ST25 application section.

**vCard demonstration**

The menu illustrated in Figure 22 indicates how to manage vCard content in the ST25DV-I2C tag.
With the MCU, the user:

- Stores an NDEF message containing a vCard. Depending on the size of the ST25DV-I2C tag, this vCard may or may not embed a picture
- Reads the ST25DV-I2C tag content.

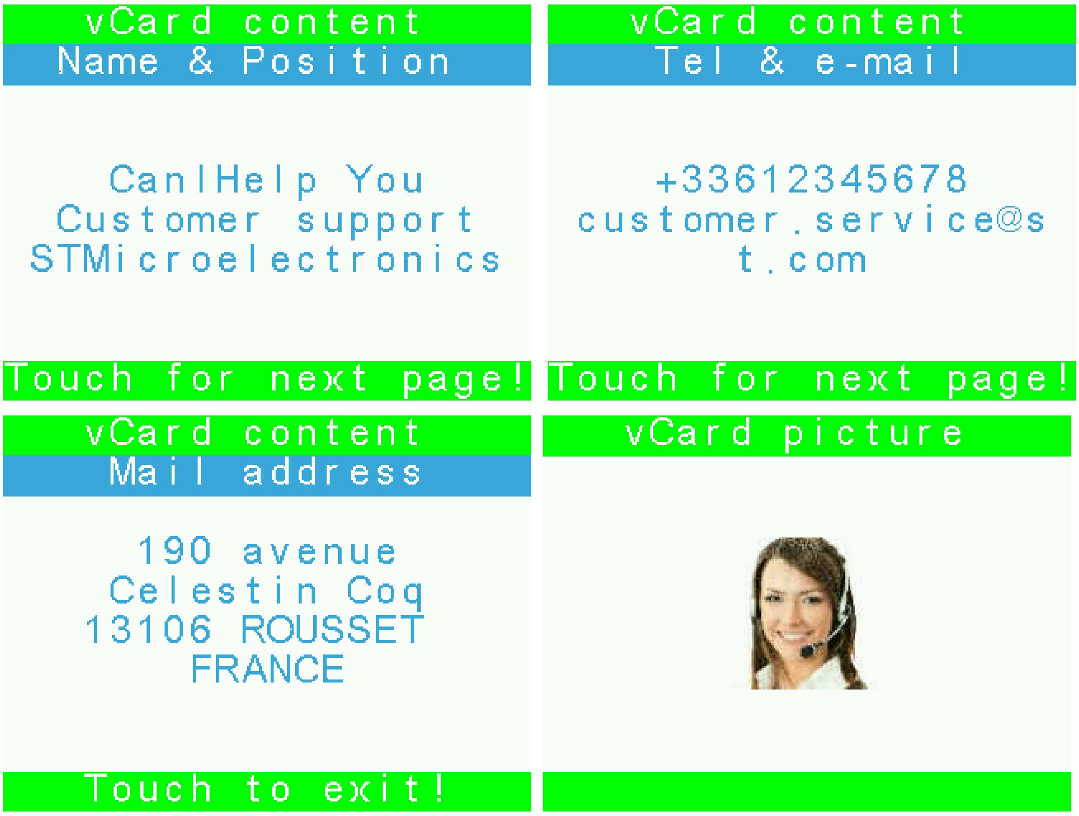
If the vCard was previously stored, the following information is displayed on the board screen: name, title, organization, cell phone number, work address and work e-mail.

*Note: The vCard with an embedded picture needs around 3 Kbytes of memory and cannot be stored in the ST25DV-I2C tag devices with smaller memory sizes, the ST25DV64 (with 64 kbits) must be used.*

*If the MCU reads the ST25DV-I2C tag memory, it does not contain a vCard, an error message is displayed.*

*Only vCard 2.1 is implemented in this firmware version. If the user stores a vCard 3.0 in the ST25DV-I2C tag, the firmware issues an error message.*

Figure 22. ST25DV64KC-DISCO read vCard content display



vCard content
Name & Position

CanIHelp You
Customer support
STMicroelectronics

Touch for next page!

vCard content
Tel & e-mail

+33612345678
customer.service@s
t.com

Touch for next page!

vCard content
Mail address

190 avenue
Celestin Coq
13106 ROUSSET
FRANCE

Touch to exit!

vCard picture

### Smart phone and vCard NDEF

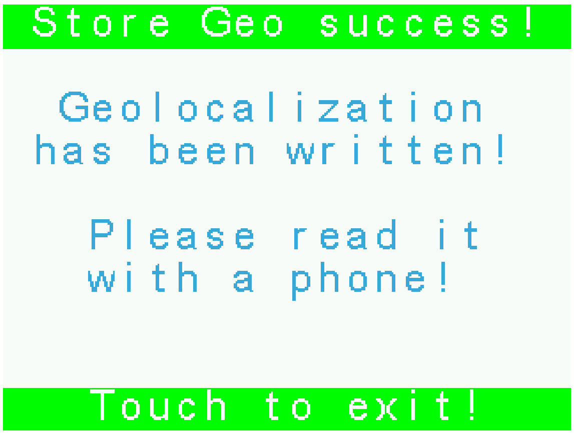
Refer to Reading and writing NDEF on a smart phone with the Android™ ST25 application section.

### Geolocation demonstration

The menu illustrated in Figure 23 indicates how to manage the geolocation content in the ST25DV-I2C tag. With the MCU, the user:

- Stores an NDEF message containing a geolocation (it is then read with an NFC reader or with a smart phone)
- Reads any geolocation that is stored in the ST25DV-I2C tag memory, the content is displayed on the screen.

Figure 23. ST25DV64KC-DISCO read geolocation display



Store Geo success!

Geolocalization
has been written!

Please read it
with a phone!

Touch to exit!

*Note: If the MCU reads the ST25DV-I2C tag memory, it does not contain a geolocation, an error message is displayed.*

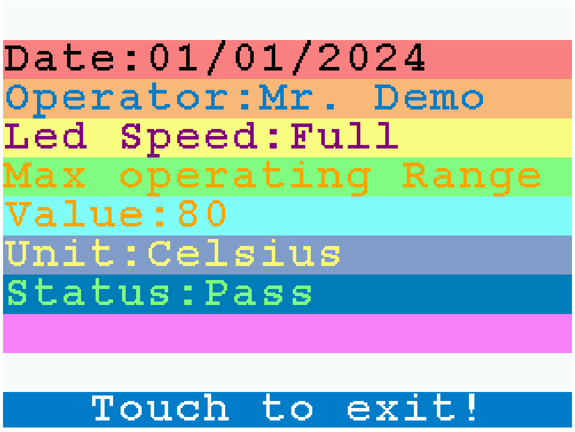
### MyApp record demonstration

This example shows the construction and usage of proprietary NDEF record. As the message format follows the NFC forum NDEF standard, it is written or read by any device supporting this standard.

On the discovery board, start the MyApp record by entering into the "Write MyApp record" menu, this writes the message in the Tag memory. Then read the message using the "Read MyApp record" menu.

This, in turn displays the content of the NDEF record as in Figure 24.

Figure 24. **ST25DV64KC-DISCO MyApp demonstration screen**



```
Date:01/01/2024
Operator:Mr. Demo
Led Speed:Full
Max operating Range
Value:80
Unit:Celsius
Status:Pass
```

The message and the display colors displayed on the LCD screen can be changed, by updating values written in the NDEF record.

The data structure is composed of:

- Five bytes for LED configuration
- Eight times 25 bytes for text and color displayed of the LCD lines.

The individual byte definitions are as follows:

- LED1 enable (1 byte, bit 0 for led on, bit 1 for flashing)
- LED2 enable (1 byte, LED not available on this discovery board)
- LED3 enable (1 byte, LED not available on this discovery board)
- LED4 enable (1 byte, LED not available on this discovery board)
- LEDs blinking speed (1 byte, value 0 to 3)
- Line number on LCD (1 byte, value 0 to 9)
- Background Color (2 bytes)
- Foreground Color (2 bytes)
- Text to display on this line (max 20 bytes, completed with spaces 0x20)
- ... (from line number to text to display repeated 7 more times)

To see changes on the screen, go back by touching the screen and reenter in the same menu.

*Note: It is possible to refresh the LCD content in live by using the GPO feature RF write. This informs the MCU when an EEPROM write has been processed, by capturing this event, the screen simply needs refreshing directly with the new content.*

### Multi record demonstration

This menu indicates how to manage a multi record NDEF in the ST25DV-I2C tag.

With the MCU, the user:

- Stores an NDEF message containing the ST25 URI (it is then be read with an NFC reader or a smart phone).
- Adds the "ST25 Android™ Application Record" (ST25 AAR) to the existing NDEF message.

For this demonstration, the user is requested to read the URI with a smart phone, and check that its preferred browser is open with the ST25 URL (this is the default behavior when URI is natively supported by the smart phone).

Then, after adding the ST25 AAR, when the ST25DV-I2C tag is read with a smart phone, the browser is no longer open, while the ST25 application is. The URI appears as the first record in the application.

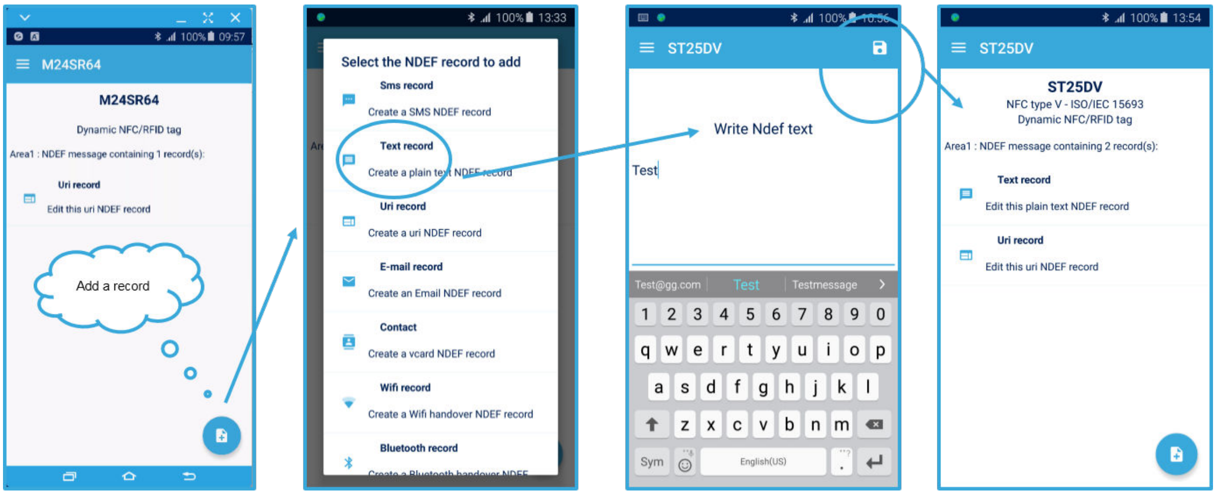
### Smart phone and multi records NDEF

This section provides dedicated indications how to build multi records NDEFs (see Figure 25).

How to build each record:

1. From "Ndef Editor" menu, click on the '+' at the bottom right corner to add a record. A popup appears allowing the user to choose the record type.
2. Fill the record and click on the floppy icon, in the top right corner, to save the record to the tag.
3. Proceed in the same way to add more records.

Figure 25. **Smart phone and multi records editor display**


M24SR64
M24SR64
Dynamic NFC/RFID tag
Area1 : NDEF message containing 1 record(s):
Uri record
Edit this uri NDEF record
Add a record
Select the NDEF record to add
Sms record
Create a SMS NDEF record
Text record
Create a plain text NDEF record
Uri record
Create a uri NDEF record
E-mail record
Create an Email NDEF record
Contact
Create a vcard NDEF record
Wifi record
Create a Wifi handover NDEF record
Bluetooth record
Create a Bluetooth handover NDEF
ST25DV
Write Ndef text
Test
Test@gg.com
Test
Testmessage
Sym
English(US)
ST25DV
ST25DV
NFC type V - ISO/IEC 15693
Dynamic NFC/RFID tag
Area1 : NDEF message containing 2 record(s):
Text record
Edit this plain text NDEF record
Uri record
Edit this uri NDEF record

### Bluetooth® Low Energy pairing demonstration (requires an additional Bluetooth® Low Energy module)

This menu indicates how to manage a Bluetooth® Low Energy pairing content in the ST25DV-I2C tag. By default, the ST25DV64KC-DISCO board does not feature the Bluetooth® Low Energy functionality, and is added using the ST BlueNRG module.

For this demonstration, the user reads the ST25DV-I2C tag content with a smart phone having the Bluetooth® Low Energy OOB support (e.g. Android™ 6.0).

With the MCU, the user:

- Stores an NDEF message containing the Bluetooth® Low Energy Out-Of-Band (OOB)
- Reads a Bluetooth® or Bluetooth® Low Energy OOB previously stored in the ST25DV-I2C tag memory.

When the user reads the ST25DV-I2C tag with a smart phone having Bluetooth® Low Energy support, the smart phone automatically asks confirmation and connects to the BlueNRG module present on the demonstration board. The MAC address provided by the smart phone is displayed on the screen of the demonstration board as soon as the pairing is completed.

The BlueNRG is configured to act as a human interface device (HID); it is displayed on the smart phone with this name: HID.

The HID protocol is natively supported by Android™ 6.0 smart phones, and this demonstration allows the user to control a pointer on the paired phone, by touching the screen of the ST25DV64KC-DISCO board. The screen acts as a touch-pad, the user button as a select button.

Touch the bottom line of the LCD to exit this demonstration.

The user selects the "Change BLE Address" command to prevent spurious connections from previously paired smart phones.

This changes the Bluetooth® Low Energy module device address, and is seen by smart phones as a different device (however the device name - HID - is left unchanged).

*Note: Some smart phones require a specific action from the user to enable the HID control (this setting may be located in the smart phone's Bluetooth® menu).*

#### Smart phone and Bluetooth® Low Energy

Refer to Reading and writing NDEF on a smart phone with the Android™ ST25 application section.

#### Wi-Fi® pairing demonstration (requires an additional Wi-Fi® module)

This menu indicates how to manage a Wi-Fi® pairing content in the ST25DV-I2C tag. By default, the ST25DV64KC-DISCO board does not feature the Wi-Fi® module. To access this feature add the ST SPWF01 module which acts as a mini Wi-Fi® access point.

For this demonstration, the ST25DV-I2C tag content is read with a smart phone using Wi-Fi®.

With the MCU, the user:

- Stores an NDEF message containing a Wi-Fi® protected setup (WPS) OOB descriptor.
- Reads a WPS OOB previously stored in the ST25DV-I2C tag memory.

When the user reads the ST25DV-I2C tag with a smart phone having the Wi-Fi® support, the smart phone automatically asks a confirmation and connects to the SWPF_AS01 Wi-Fi® network. The MAC address provided by the smart phone is displayed on the screen of the demonstration board as soon as the connection is completed.

*Note: For the demonstration no Internet access is provided by the board mini access point.*

*Note: If, when the MCU reads the ST25DV-I2C tag memory, it does not contain a WPS OOB, an error message is displayed.*

#### Smart phone and Wi-Fi® pairing

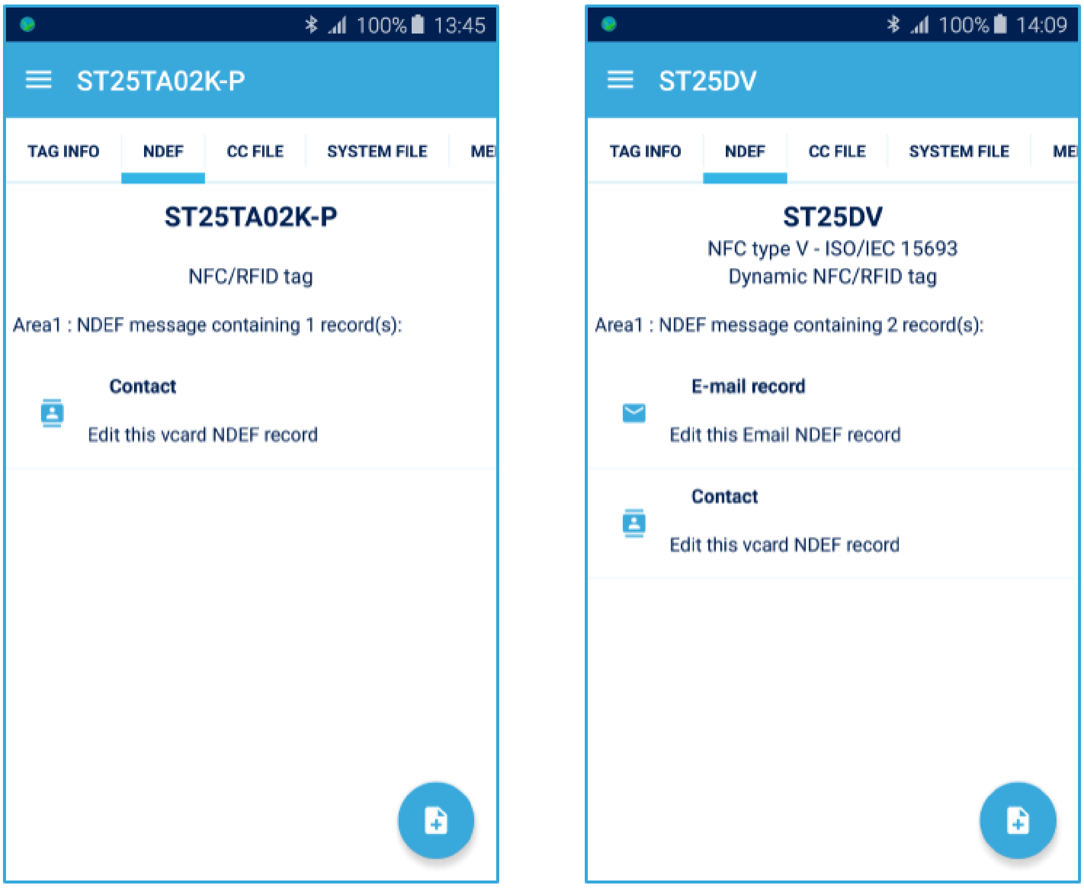
Refer to Reading and writing NDEF on a smart phone with the Android™ ST25 application section.

#### Reading and writing NDEF on a smart phone with the Android™ ST25 application

The ST25 NFC App enables to read or write NDEF messages.

- Read: tap the tag, if a managed NDEF is discovered, content is accessible from the NDEF tab or NDEF editor (refer to the Android™ application user manual for more details), see Figure 26.
- Write an NDEF: select "NDEF editor" and choose the NDEF type has to be written. An editor according to the selected type is displayed (see Figure 27).

Figure 26. Smart phone and NDEF discover display



13:45

ST25TA02K-P

TAG INFO | NDEF | CC FILE | SYSTEM FILE | ME

ST25TA02K-P

NFC/RFID tag

Area1 : NDEF message containing 1 record(s):

Contact

Edit this vcard NDEF record

14:09

ST25DV

TAG INFO | NDEF | CC FILE | SYSTEM FILE | ME

ST25DV

NFC type V - ISO/IEC 15693

Dynamic NFC/RFID tag

Area1 : NDEF message containing 2 record(s):

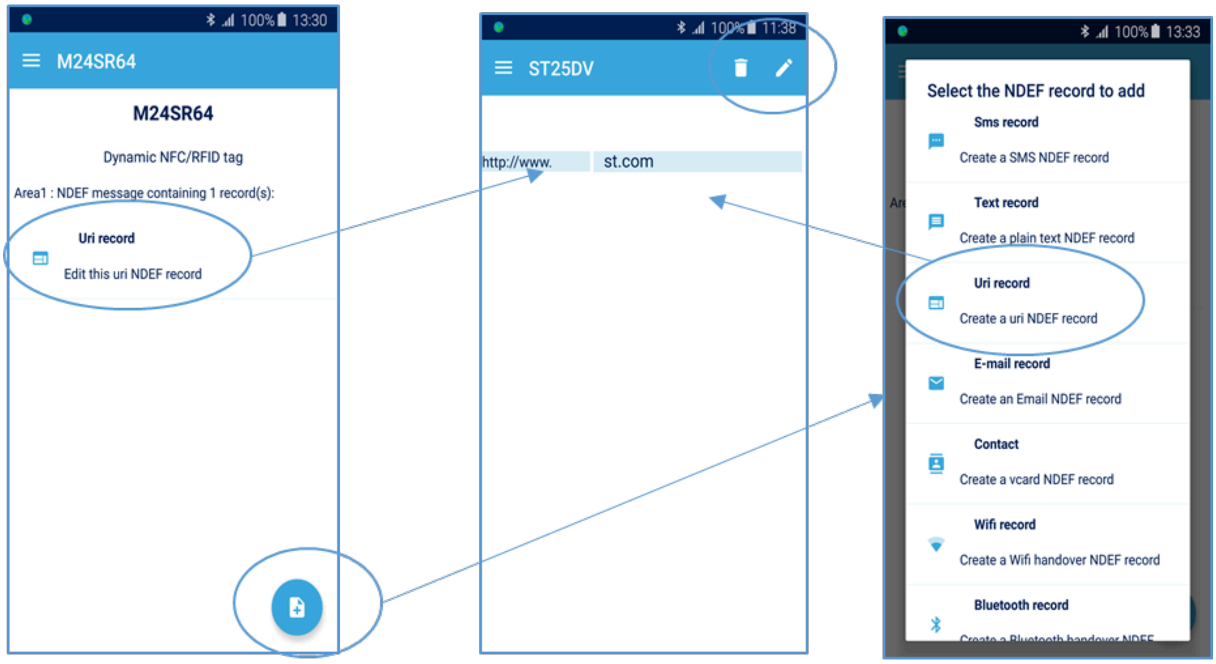
E-mail record

Edit this Email NDEF record

Contact

Edit this vcard NDEF record

Figure 27. Smart phone and URL NDEF write menu display



13:30

M24SR64

M24SR64

Dynamic NFC/RFID tag

Area1 : NDEF message containing 1 record(s):

Uri record

Edit this uri NDEF record

11:38

ST25DV

http://www. | st.com

13:33

Select the NDEF record to add

Sms record

Create a SMS NDEF record

Text record

Create a plain text NDEF record

Uri record

Create a uri NDEF record

E-mail record

Create an Email NDEF record

Contact

Create a vcard NDEF record

Wifi record

Create a Wifi handover NDEF record

Bluetooth record

Create a Bluetooth handover NDEF

### Reading and writing NDEF with the ST25PC-NFC software

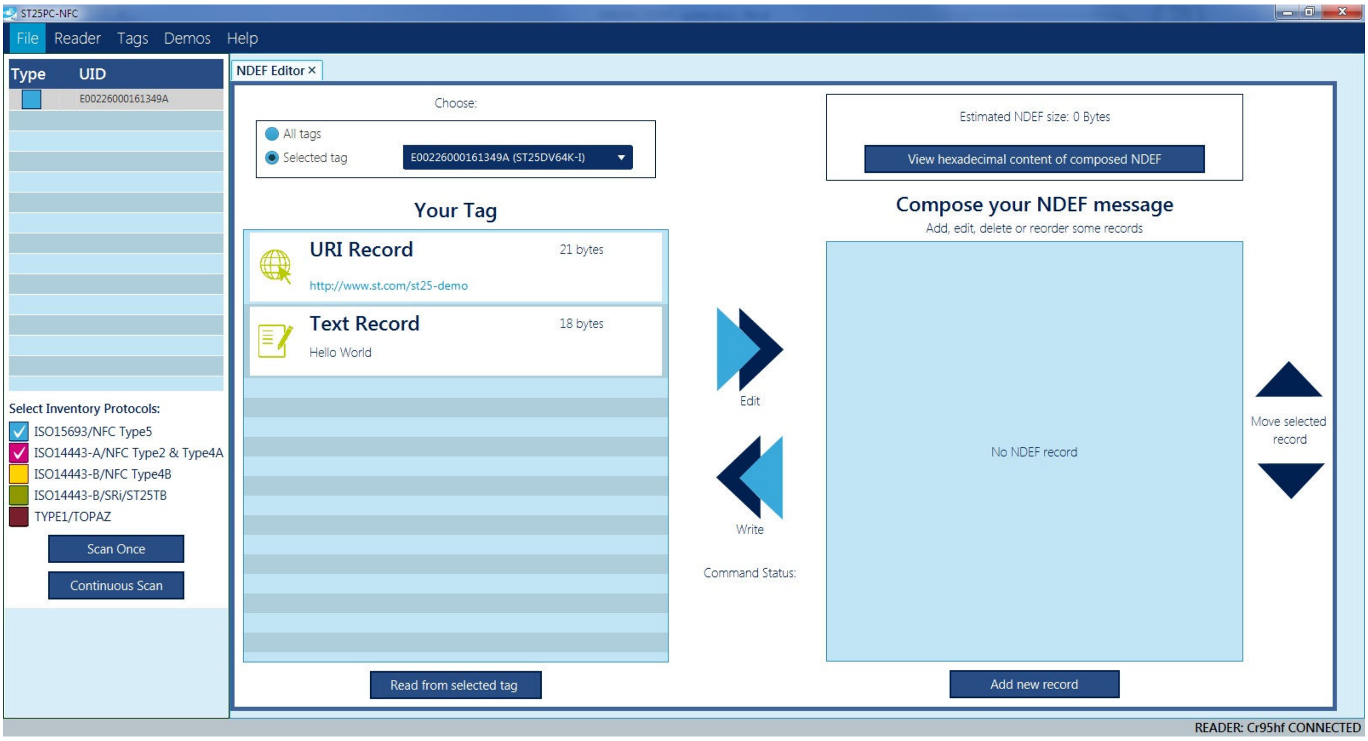
To access the NDEF editor screen, start by right-clicking on a ST25DV-I2C tag tag in the "Inventory Pane" (see Figure 8), then select the "NDEF Editor" option in the menu. The NDEF interface is illustrated in Figure 28.

The first list on the left of the NDEF Editor screen displays the content of the tag. Click on the "Read from selected tag" button to refresh the list content.

The right-most list helps the user compose an NDEF message. The current tag content is then copied by clicking on the "Edit" arrow located between the two lists. Add a new record by clicking on the "Add new record" button at the bottom right side of the screen. To remove a record from the composition list, click on the "X" icon.

To write the NDEF message to the tag, click on the "Write" icon located between the two lists.

Figure 28. **ST25PC-NFC software NDEF interface**



E00226000161349A

Selected tag: E00226000161349A (ST25DV64K-I)

URI Record 21 bytes
http://www.st.com/st25-demo

Text Record 18 bytes
Hello World

☑ ISO15693/NFC Type5
☑ ISO14443-A/NFC Type2 & Type4A
□ ISO14443-B/NFC Type4B
□ ISO14443-B/SRi/ST25TB
□ TYPE1/TOPAZ

### Clearing the EEPROM

This menu allows the user to clear the EEPROM content. The three choices are:

1. Write an empty NDEF
2. Erase the CC file (replace the first four bytes of memory with 0xFF)
3. Clear EEPROM (fill the whole memory with 0xFF).

## 4.2.8 Fast transfer mode demonstration menu

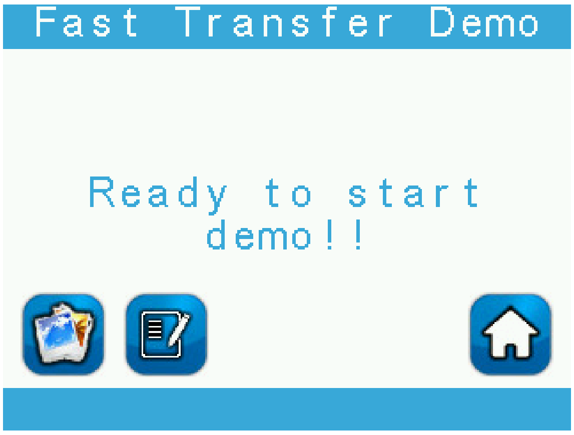
This demonstration is intended to show how the ST25DV-I2C tag improves the data exchange between the reader and the MCU using a proprietary protocol based on the ISO15693 standard as opposed to exchanging data through the EEPROM. To perform the data exchange the ST25DV-I2C tag shares a 256-byte RAM buffer available for the reader and the MCU (also called mailbox).

Data exchange is only carried out in one direction at a time (half duplex). The mailbox contains indicators giving the reader status and generates pulses on a GPO, which must be connected to an external interrupt (GPIO input) of the microcontroller. For more details on the mailbox feature, refer to the ST25DV-I2C tag datasheet.

When the user selects the Fast transfer mode icon (see Figure 4), a message displays "Starting Demo". During this time, the firmware is initialized, enabling the mailbox functionality and erasing the Flash memory on the MCU (to store data downloaded through NFC, dedicated demonstration Flash memory area). Then, when the demonstration has finished its initialization, the message "Ready to start demo ! !" is displayed, as shown in Figure 29.

*Note: When the Mailbox feature is enabled, the EEPROM is in read only mode. To modify the EEPROM disable the "Mailbox" feature.*

Figure 29. **ST25DV64KC-DISCO Fast transfer mode display**



Fast Transfer Demo

Ready to start demo!!

The picture and note icons in Figure 29, on the bottom left of the screen, point to a sub-menu that allows the user to choose a picture or a random data buffer, which is to be transferred to a reader (see Host to reader image upload and Host to Reader data transfer sections).

The home icon (bottom right) allows the user to go back to the main menu (see Figure 4).

At this step, the firmware is waiting for an action from the reader or from the user to start. These actions are described in the following sections.

### Smart phone and Fast transfer mode use cases

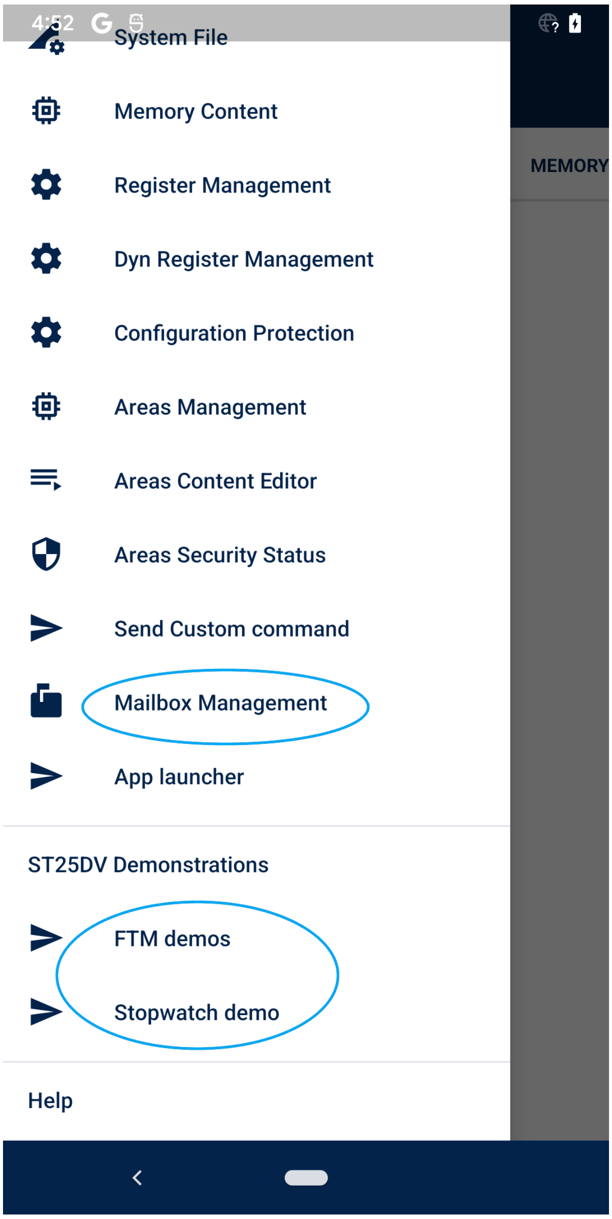
Available actions can be accessed after tag has been taped, and are available in the ST25DV-I2C tag features or demonstrations menus within the application "Drawer".

The available use cases are:

- Mailbox management, enable/disable the mailbox and display mailbox register fields status
- Data transfer, to demonstrate basics transfers
- Firmware upgrade, to demonstrate the firmware upgrade use case
- Picture transfer, to demonstrate how to upload or download pictures
- Stopwatch synchronization, to demonstrate fast transfer with a smart phone.

The display is illustrated in Figure 30.

Figure 30. **Smart phone and Fast transfer mode use cases display**



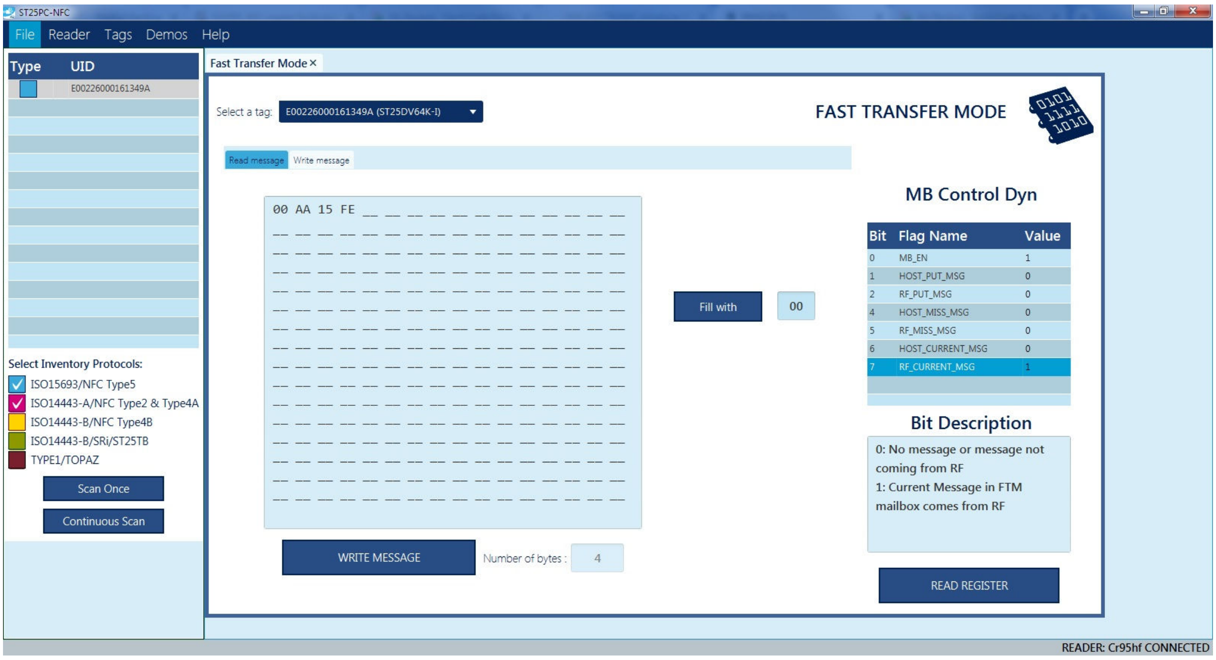
### PC software and Fast transfer mode use cases

As with Android™ phones, all the Fast transfer mode demonstrations are played using the ST25PC-NFC software. The CR95HF is used as a RF reader, for more information refer to *Software toolbox for NFC tags* user manual (UM2444), available on *www.st.com*.

To access the Fast transfer mode screen, start by right-clicking on a ST25DV-I2C tag tag in the "Inventory" pane (see Figure 31), then select the "Fast Transfer Mode" option in the menu. Two tabs appear below the tag selection box "Read Message" (by default) and "Write Message". Click on the "Write Message" tab (see Figure 31).

The message area allows the user to write up to 256 bytes of data. A "Fill with" button fills up the whole buffer with the specified byte value. After the message is written it is sent to the tag mailbox by clicking on the "Write Message" button.

Click on "Read Register" to the right of the screen to read the values of the tag dynamic registers.



```
00 AA 15 FE
```

MB Control Dyn

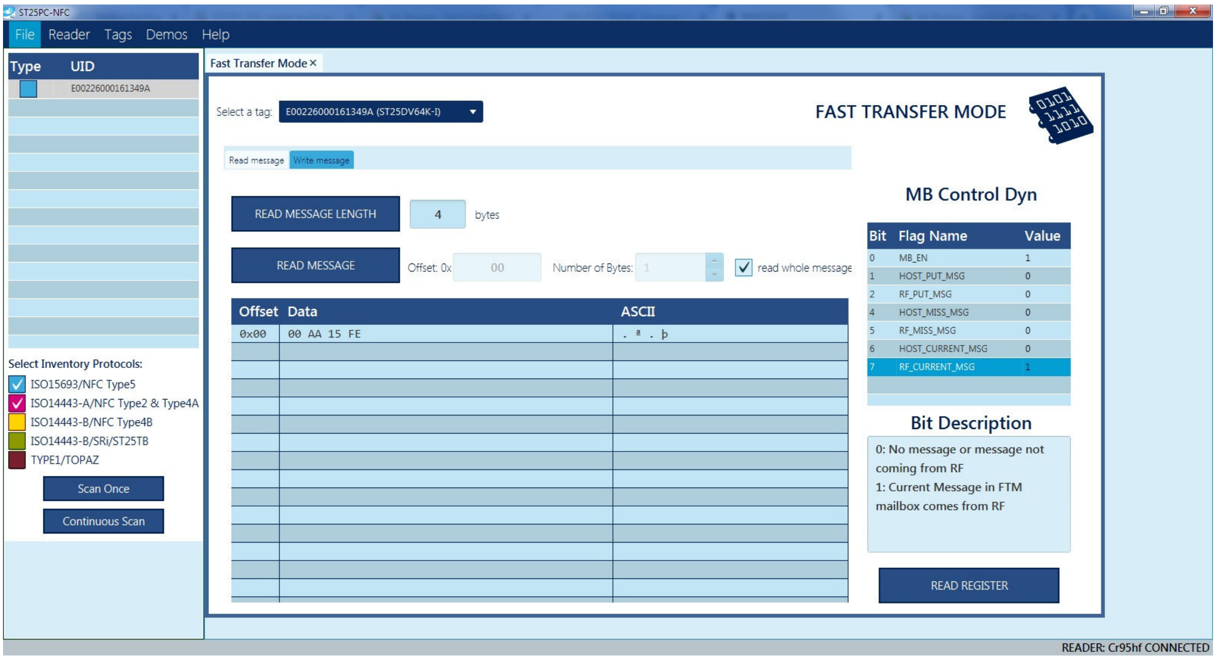
| Bit | Flag Name | Value |
|---|---|---|
| 0 | MB_EN | 1 |
| 1 | HOST_PUT_MSG | 0 |
| 2 | RF_PUT_MSG | 0 |
| 4 | HOST_MISS_MSG | 0 |
| 5 | RF_MISS_MSG | 0 |
| 6 | HOST_CURRENT_MSG | 0 |
| 7 | RF_CURRENT_MSG | 1 |

Bit Description

0: No message or message not coming from RF
1: Current Message in FTM mailbox comes from RF

## Figure 31. ST25PC-NFC software - ST25DV-I2C tag FTM interface Write tab

Once the tag mailbox is filled , switch to the "Read message" tab (see Figure 32).



| Offset | Data | ASCII |
|---|---|---|
| 0x00 | 00 AA 15 FE | . ª . þ |

MB Control Dyn

| Bit | Flag Name | Value |
|---|---|---|
| 0 | MB_EN | 1 |
| 1 | HOST_PUT_MSG | 0 |
| 2 | RF_PUT_MSG | 0 |
| 4 | HOST_MISS_MSG | 0 |
| 5 | RF_MISS_MSG | 0 |
| 6 | HOST_CURRENT_MSG | 0 |
| 7 | RF_CURRENT_MSG | 1 |

Bit Description

0: No message or message not coming from RF
1: Current Message in FTM mailbox comes from RF

## Figure 32. ST25PC-NFC software - ST25DV-I2C tag FTM interface Read tab

Click on the "Read Message Length" button to get the size in bytes of the mailbox content, and on the "Read Message" button to display the current content of the mailbox.

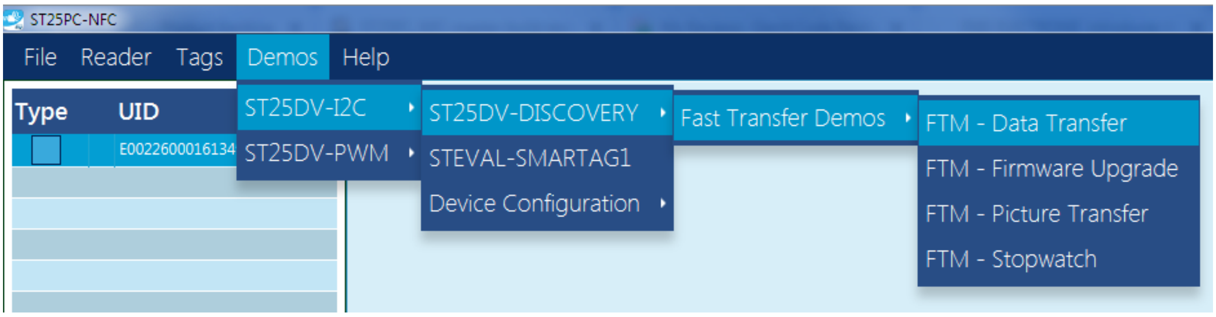
In the ST25PC-NFC software, the full FTM set of demonstrations are accessible from the menu. Figure 33 illustrates how to access them.

Available demonstrations are:

- "Reader-to-Host" and "Host-to-Reader" R2H data transfer: transfer of binary file to/from PC software (using the CR95HF board) to and from the ST25DV64KC-DISCO board.

- Reader-to-Host" firmware upgrade: transfer of binary file containing a new STM32 firmware from PC software (using CR95HF board) to ST25DV64KC-DISCO board, the ST25DV64KC-DISCO firmware is upgraded.
- "Reader-to-Host" and "Host-to-Reader" image transfer: transfer of JPG image to/from PC software (using CR95HF board) to and from the ST25DV64KC-DISCO board, the image is displayed on ST25DV64KC-DISCO LCD.

Figure 33. ST25PC-NFC software - Access to ST25V-I2C FTM demonstrations

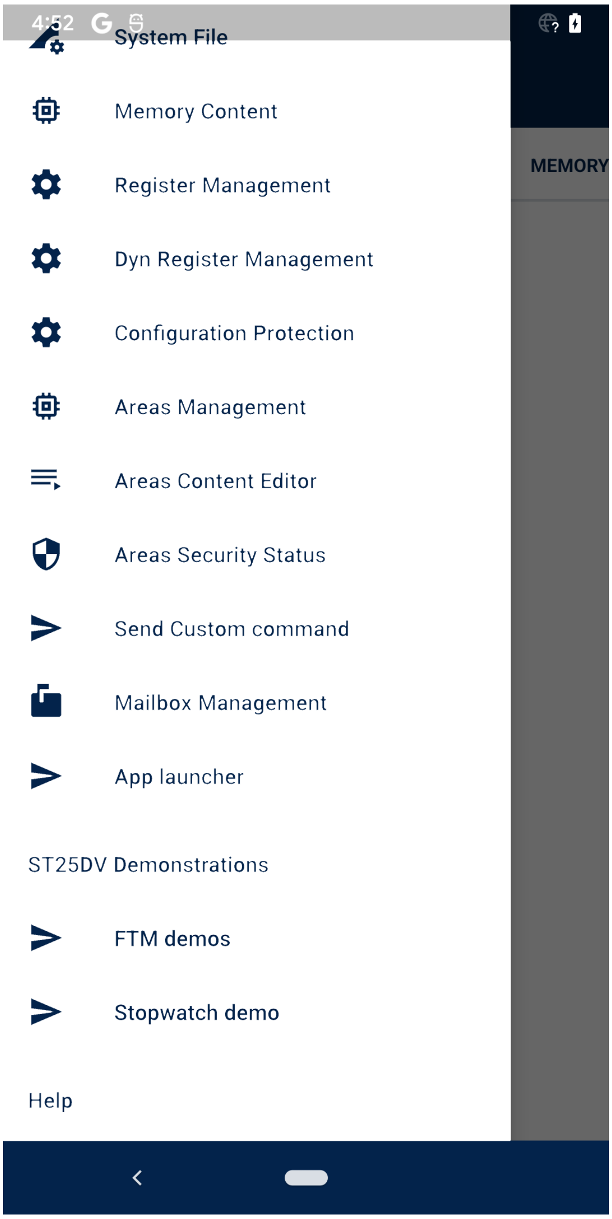


### Smart phone and FTM data transfers

Several menus accessible by the drawer menu are dealing with the Fast transfer mode data transfers:

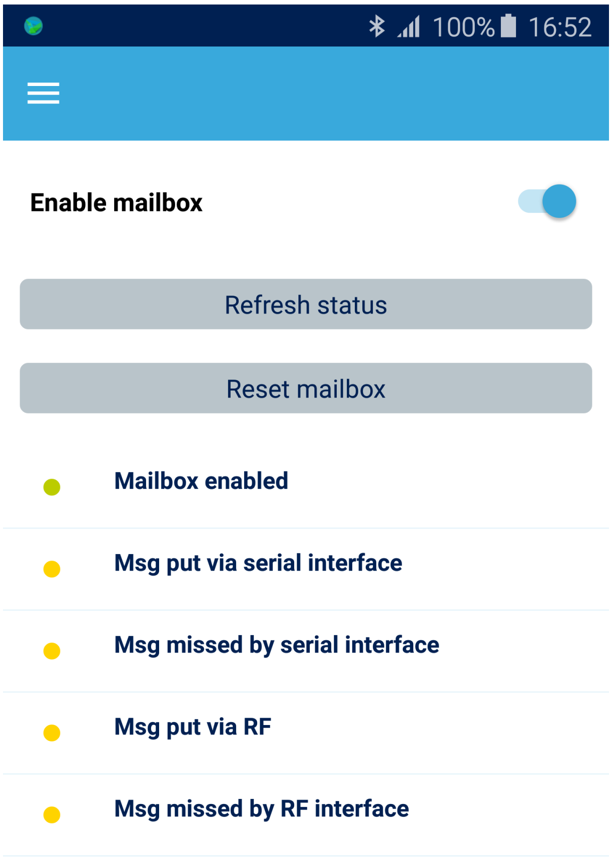
- Mailbox management. Radio buttons give the current status of the mailbox, and this menu permits to enable/disable the mailbox feature
- FTM demonstrations:
  - Send or receive data
  - Firmware update
  - Send or receive pictures.
- Stopwatch transfers: This demonstration shows a chronometer synchronized between the smart phone and the tag using the mailbox. It shows the low latency.

**Figure 34. ST25DV drawer menu (Mailbox management, FTM demonstration, Stopwatch demonstration)**



As prerequisite to any data transfer, the mailbox has to be enabled, without any pending message. The correct status is shown in Figure 35.

Figure 35. Mailbox status



● Mailbox enabled

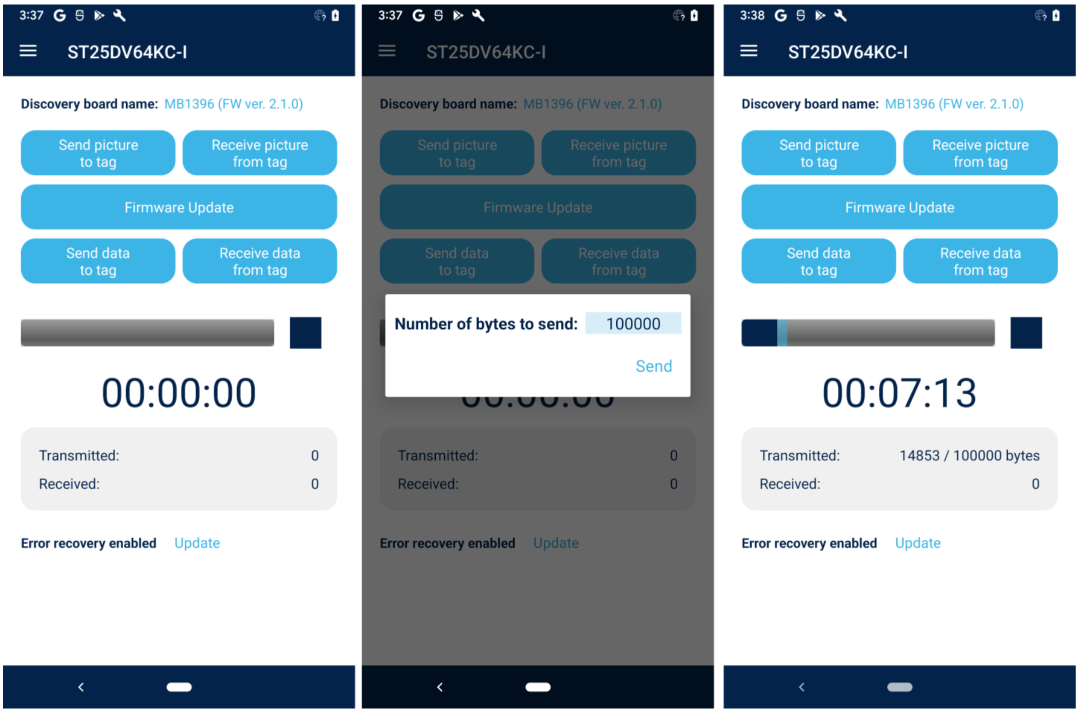
● Msg put via serial interface

● Msg missed by serial interface

● Msg put via RF

● Msg missed by RF interface

Figure 36. Data transfers from reader to tag of a random buffer



Discovery board name: MB1396 (FW ver. 2.1.0)

00:00:00

Transmitted: 0

Received: 0

Number of bytes to send: 100000

00:07:13

Transmitted: 14853 / 100000 bytes

Received: 0

*Note: According to data size selected, the corresponding payload data is initialized with random values on which a CRC is computed to be sure that the host computes the same data after each transfer.*

*Note: For every transfers done with the Android™ application, the user can choose between two modes:*

- *Error recovery enabled. With this mode, the protocol is able to detect transmission errors and to recover (by retransmitting the faulty packets). This is the default (and recommended) mode.*
- *Error recovery disabled. Data transfer is done at max throughput. There is no recovery in case of transmission error.*

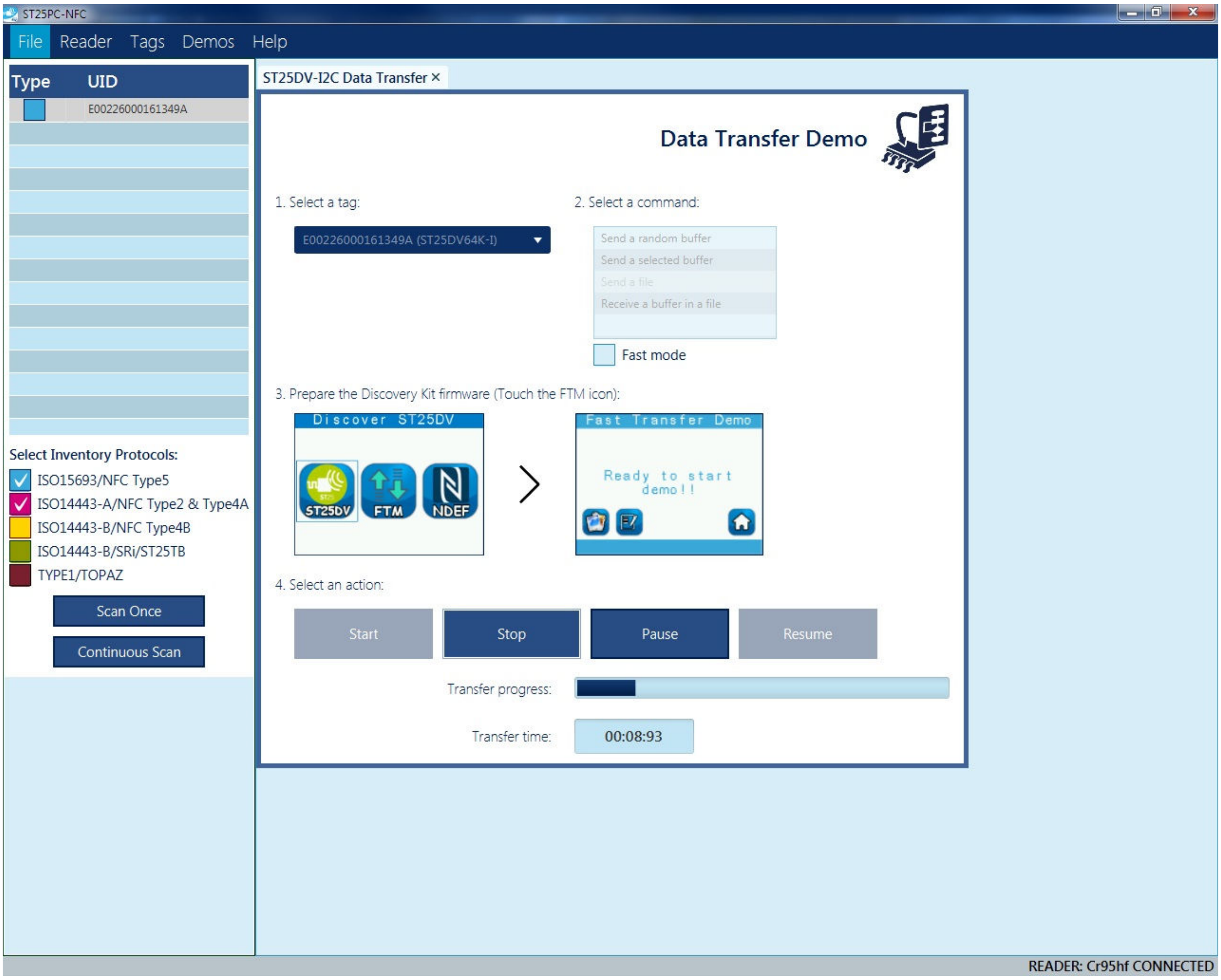
### PC software and FTM data transfers

In the ST25DV-I2C tag/ST25DV-DISCOVERY "Demos" menu (see Figure 33), select the "FTM – Data Transfer" item which leads to the "Data. Transfer Demo" in Figure 37.

To execute a R2H data transfer, select the "Send a file" item from the command list. A window pops-up asking the user to select a file to transfer to the ST25DV64KC-DISCO board. CRC computation is done when loading the file.

Then click on the "Start" button to start transferring data to the board.

Figure 37. **Software R2H data transfer display**



1. Select a tag: E00226000161349A (ST25DV64K-I)

2. Select a command:
Send a random buffer
Send a selected buffer
Send a file
Receive a buffer in a file

□ Fast mode

3. Prepare the Discovery Kit firmware (Touch the FTM icon):

4. Select an action:

Transfer progress:

Transfer time: 00:08:93

READER: Cr95hf CONNECTED

A progress bar and a timer display the transfer progress.

At the end of the transfer, the CRC calculated by the ST25DV64KC-DISCO board is compared with the CRC calculated by the PC software. Demonstration duration is displayed too.

### Reader to host firmware upgrade

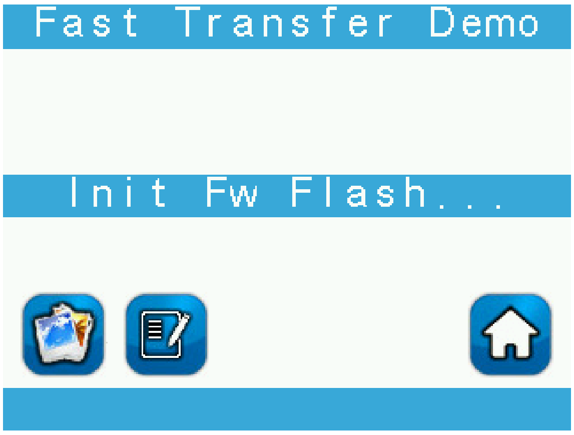
This function allows the user to send data (that is manipulated as binary) from the reader to the STM32L476xx microcontroller device. The data is stored in the Flash memory and at the end of the transfer, the following instruction code jumps to the new firmware location, demonstrating the capability to transfer large mounts of binary data. For this example, the new firmware is stored in a secondary area in the Flash memory and it is not intended to be used for new demonstrations. To come back to the initial firmware, press the reset button and it reboots the reader with the with this firmware.

To prepare the ST25DV64KC-DISCO for the firmware upgrade demonstration, power up the board, then touch the FTM icon to start the fast transfer mode demonstration (Figure 29).

The kit is now waiting for a reader action (see Smart phone and firmware upgrade and PC software and firmware upgrade sections) to continue the demonstration. The dedicated firmware is located in: **ST25DV-DISCOVERY\Demonstrations\ST25DVDemo_FwUpgd\Binary** folder

To execute the firmware download, the user first has to send (from the reader) a correct password (for this demonstration the password is 0x12345678). During the password check, the Flash memory area where the firmware is to be stored is erased (Figure 38).

Figure 38. ST25DV64KC-DISCO firmware upgrade: password check

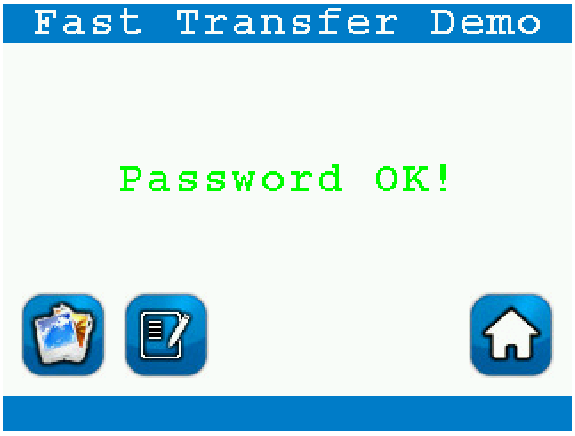


Fast Transfer Demo

Init Fw Flash...

If the password is correct, the user is authorized to start the transfer (Figure 39), if not, the user has to enter a new password to continue (Figure 40).

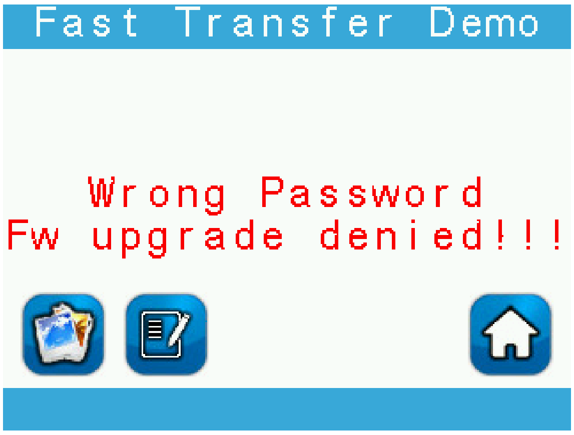
Figure 39. ST25DV64KC-DISCO firmware upgrade: password OK



Fast Transfer Demo

Password OK!

Figure 40. ST25DV64KC-DISCO firmware upgrade: wrong password

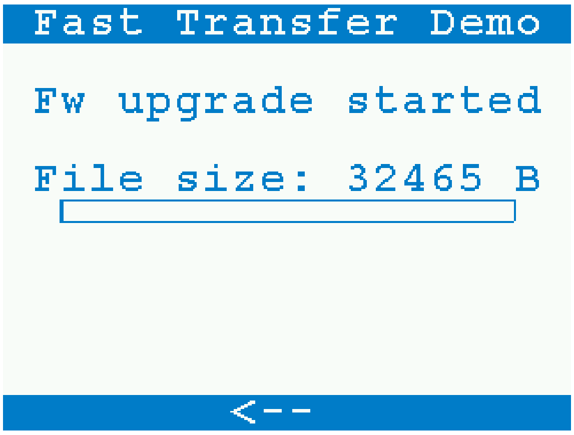


Fast Transfer Demo

Wrong Password
Fw upgrade denied!!!

During the transfer the full length of data transmitted is displayed, and a bar indicates the progress (see Figure 41).

Figure 41. **ST25DV64KC-DISCO software transfer progress status display**



Fast Transfer Demo

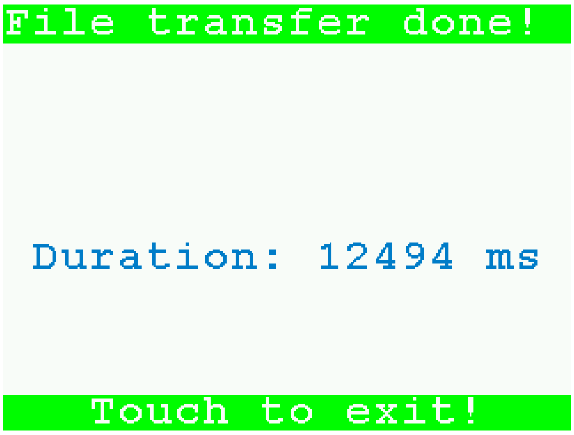
Fw upgrade started

File size: 32465 B

DT52467V2

When the transfer is successfully completed, the transfer time is displayed (see Figure 42). In case of failure the display shows an error message.

Figure 42. **ST25DV64KC-DISCO firmware upgrade: transfer done**



File transfer done!

Duration: 12494 ms

DT52468V2

To start the new firmware, simply tap the screen.

### Smart phone and firmware upgrade

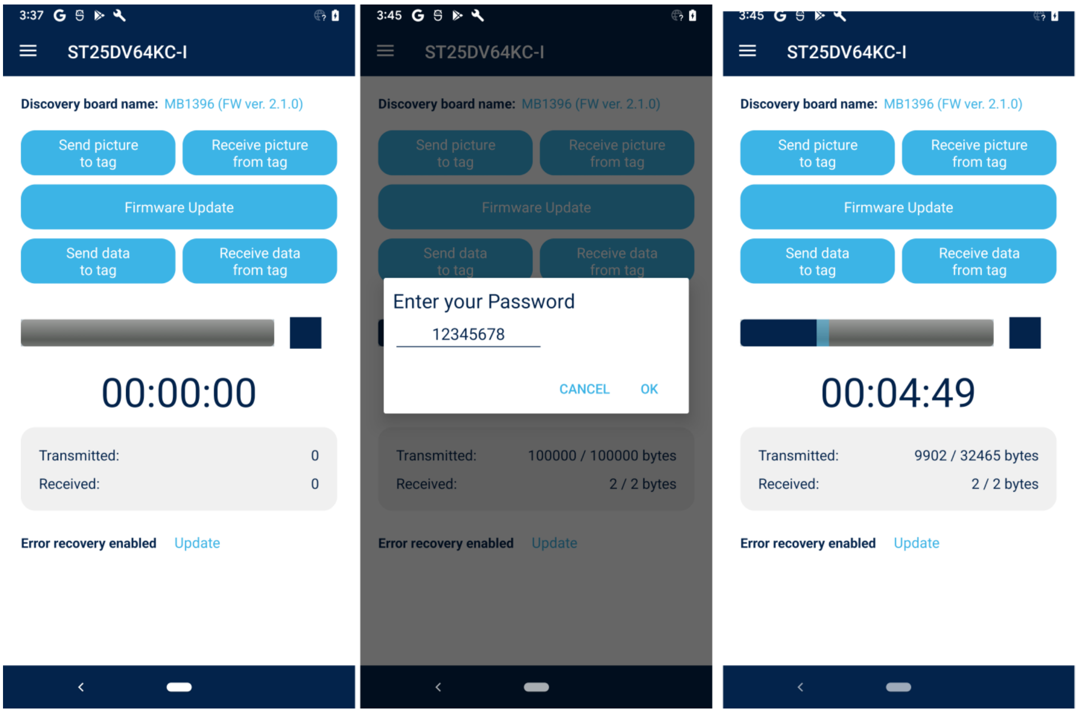
Features:

- Upgrade a new firmware to the ST25DV64KC-DISCO board.

Demonstration steps, see the screen illustration in Figure 43:

1. Check the mailbox status (Must be enable, withtout any pending messages in the mailbox).
2. Select the “Firmware demos” in the drawer menu and click on “Firmware update button”.
3. Select the file to be downloaded.
4. A pop up asks for a password. By default, and for demonstration purpose the password is 12345678.
5. Start download = Start transfer button.
6. Smart phone and firmware download.

Figure 43. Smart phone and firmware upgrade display



Discovery board name: MB1396 (FW ver. 2.1.0)

00:00:00

Transmitted: 0
Received: 0

Error recovery enabled

Discovery board name: MB1396 (FW ver. 2.1.0)

Enter your Password
12345678

Transmitted: 100000 / 100000 bytes
Received: 2 / 2 bytes

Error recovery enabled

Discovery board name: MB1396 (FW ver. 2.1.0)

00:04:49

Transmitted: 9902 / 32465 bytes
Received: 2 / 2 bytes

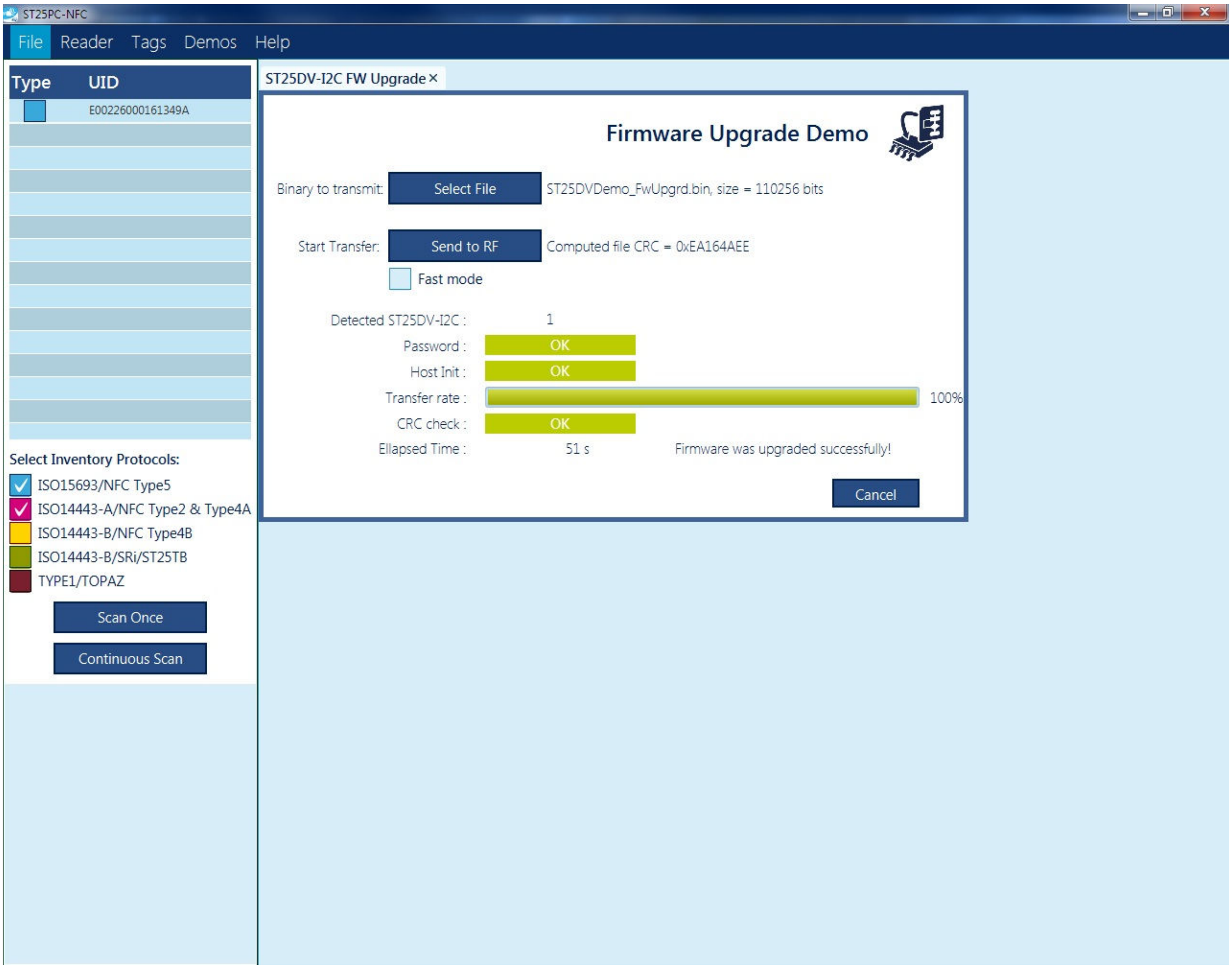
Error recovery enabled

### PC software and firmware upgrade

In the ST25DV-I2C tag/ST25DV-DISCOVERY "Demos" menu (see Figure 33), select the "FTM – Firmware Upgrade" item.

To execute a R2H firmware upgrade, first select a binary file to send with the "Select File" button (see Figure 44). CRC computation is done when loading the file, and is displayed on the screen.

To start the firmware upgrade, click on the "Send to RF" button. Status of the transfer is updated in real time illustrated by the "Transfer rate" progress bar.



| Type | UID |
|---|---|
|  | E00226000161349A |

Select Inventory Protocols:

- ☑ ISO15693/NFC Type5
- ☑ ISO14443-A/NFC Type2 & Type4A
- ☐ ISO14443-B/NFC Type4B
- ☐ ISO14443-B/SRi/ST25TB
- ☐ TYPE1/TOPAZ

Firmware Upgrade Demo

Binary to transmit: Select File ST25DVDemo_FwUpgrd.bin, size = 110256 bits

Start Transfer: Send to RF Computed file CRC = 0xEA164AEE

☐ Fast mode

Detected ST25DV-I2C : 1

Password : OK

Host Init : OK

Transfer rate : 100%

CRC check : OK

Ellapsed Time : 51 s Firmware was upgraded successfully!

Figure 44. PC software R2H successful firmware upgrade

### Reader to host image download

This function sends data from the reader to the STM32L476xx microcontroller device. In this demonstration, data is composed of jpeg images, stored in the Flash memory and displayed when the transfer is successful.

To prepare the ST25DV64KC-DISCO for the R2H picture transfer demonstration, power on the board, then touch the FTM icon to start the fast transfer mode demonstration (see Figure 29).

It is now waiting for a reader action to continue the demonstration, as illustrated in the Smart phone and picture transfers and Smart phone and picture download transfer sections.

During the transfer, the full length of data transmitted is displayed and a progress bar indicates the progression. When the transfer is successfully done, the computed CRC and transfer duration are displayed on screen. To display the downloaded picture touch the screen. To return to the Fast transfer mode demonstration simply touch the screen.

### Smart phone and picture transfers

Features:

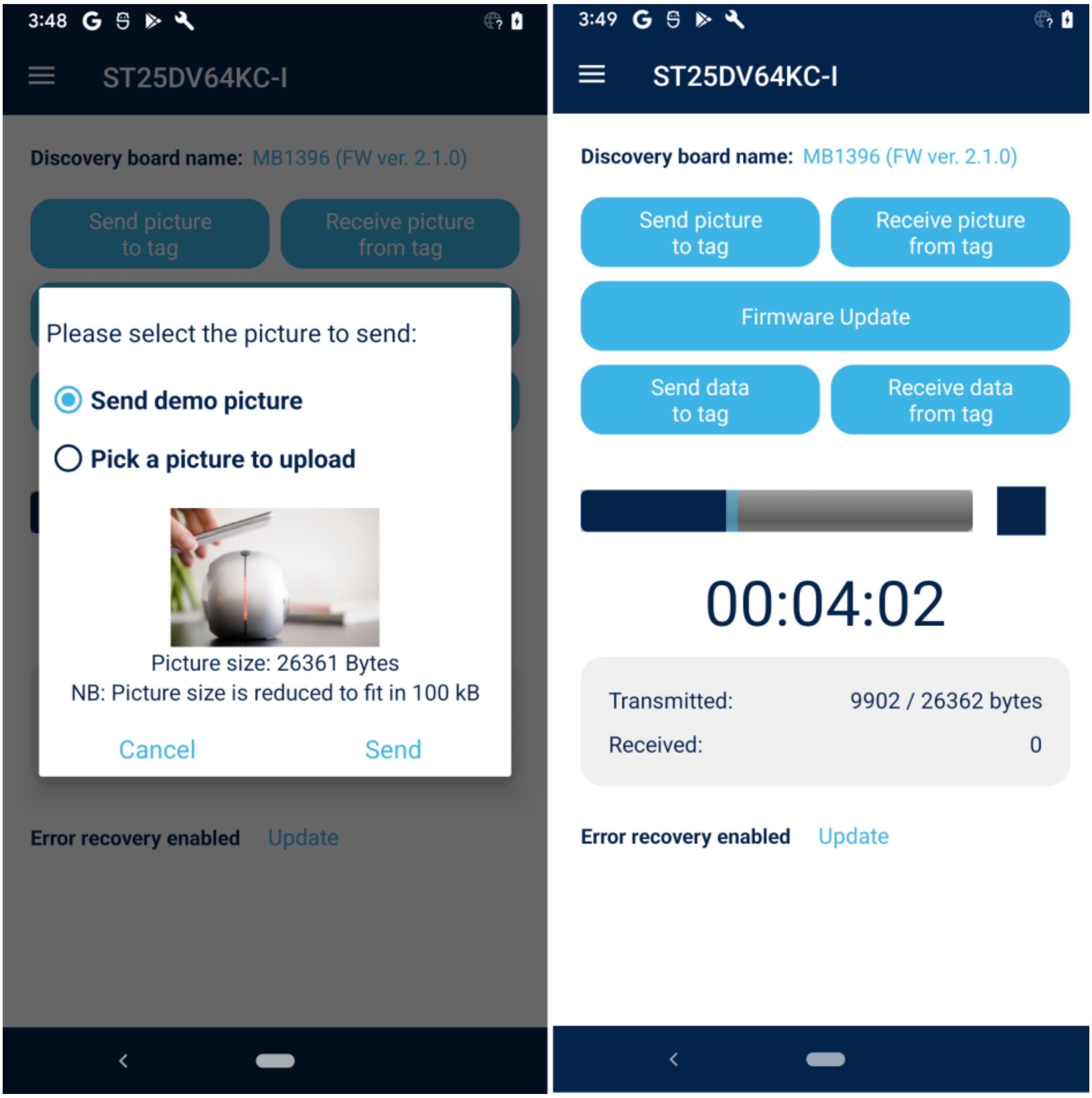
- Send a picture to the tag.
- Receive a picture from the tag.

Picture upload steps:

1. Check the mailbox status (c.f. drawer menus).
2. Select “FTM demos” in drawer menu.

3. Click on “Send picture to tag” Select a picture in phone memory.
The upload display screens are illustrated in Figure 45 and .

Figure 45. Transfer from smart phone to tag



3:48

ST25DV64KC-I

Discovery board name: MB1396 (FW ver. 2.1.0)

Please select the picture to send:

● Send demo picture

○ Pick a picture to upload

Picture size: 26361 Bytes
NB: Picture size is reduced to fit in 100 kB

Cancel Send

Error recovery enabled Update

3:49

ST25DV64KC-I

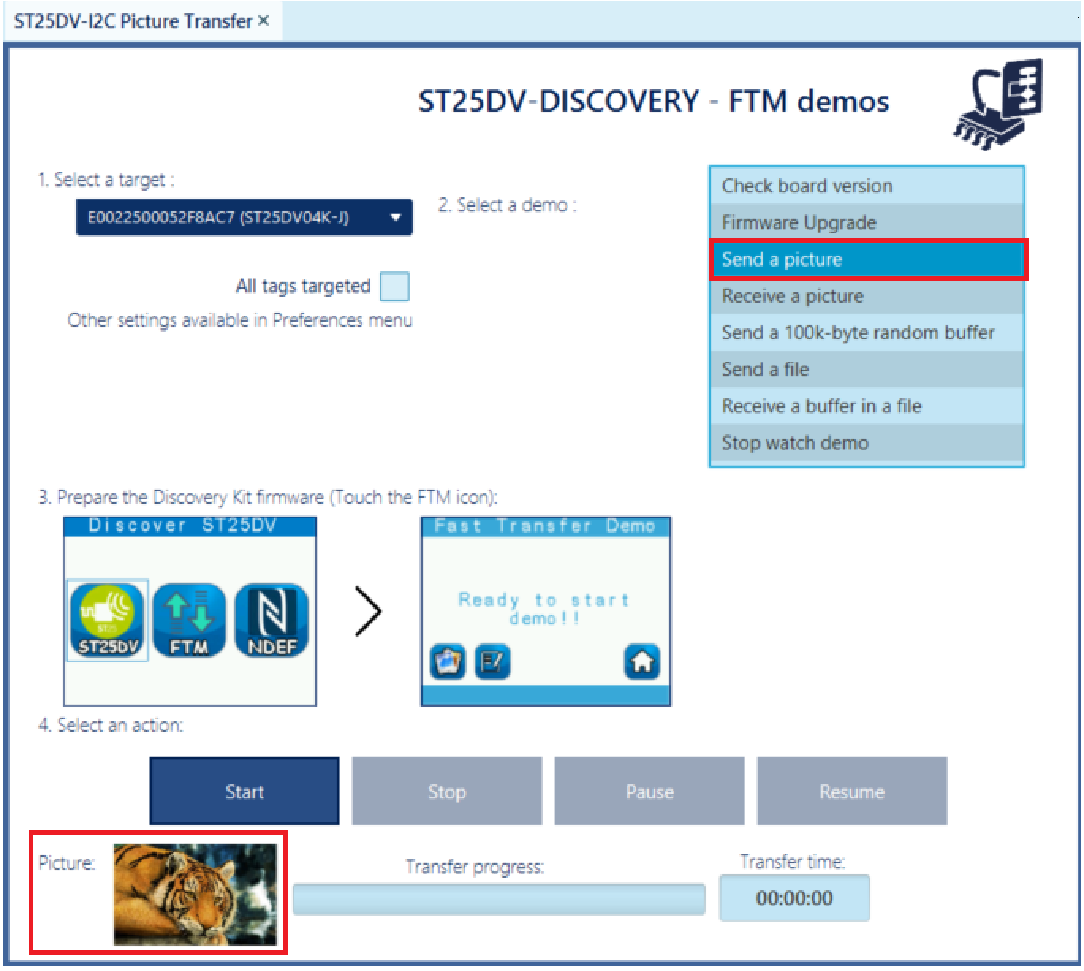
Discovery board name: MB1396 (FW ver. 2.1.0)

00:04:02

Transmitted: 9902 / 26362 bytes
Received: 0

Error recovery enabled Update

Figure 46. Image transfer demonstration: send a picture

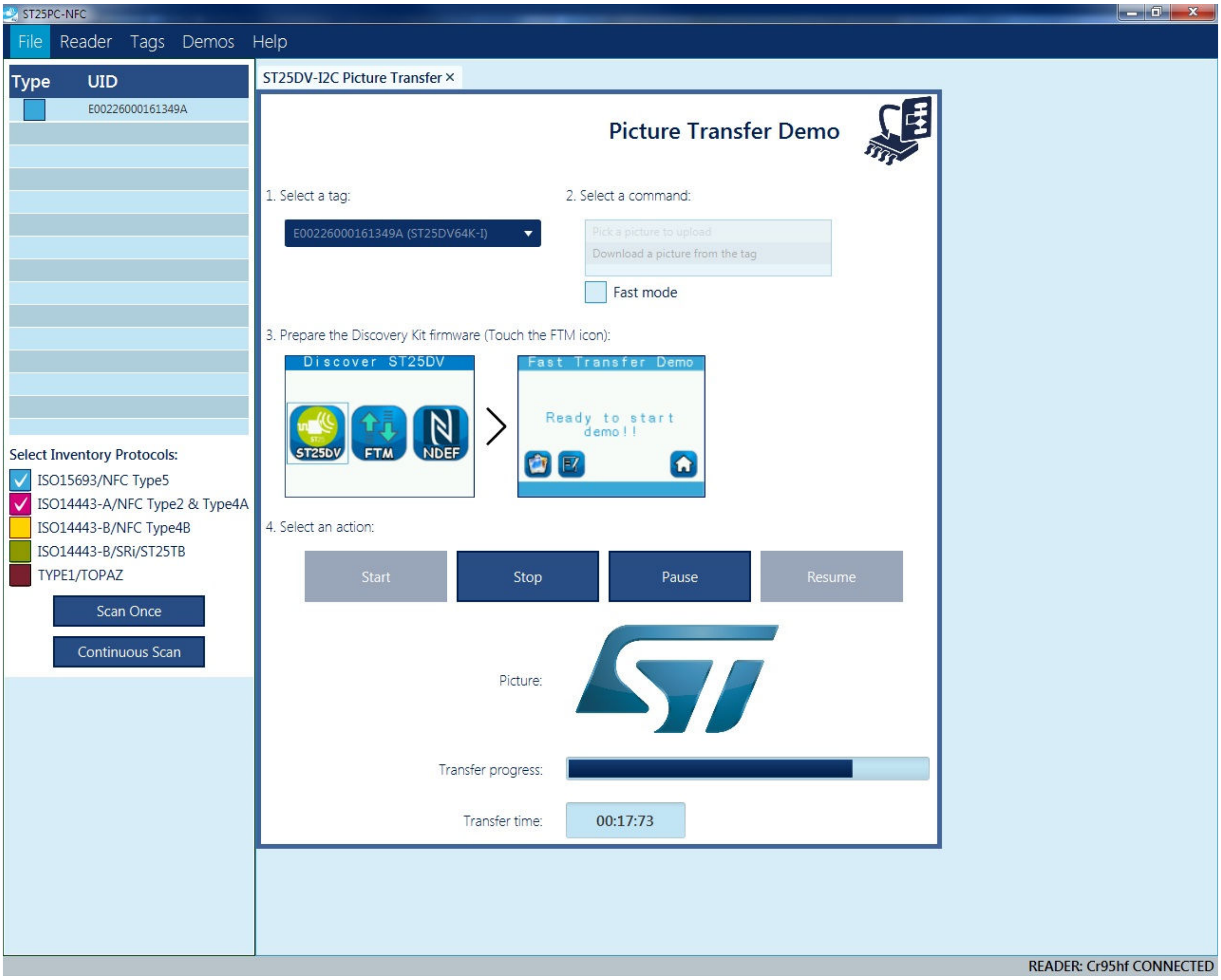


### PC software and image transfer

In the ST25DV-I2C tag/ST25DV-DISCOVERY "Demos" menu (see Figure 33), select the "FTM – Picture Transfer" item.

To run a R2H transfer (from CR95HF reader to Host board), first select the "Pick a picture to upload" item from the command list (see Figure 47). Choose a file from the window that pops up, then click on the "Start" button.

Figure 47. **PC software R2H image transfer display**



| Type | UID |
|---|---|
| ■ | E00226000161349A |

Select Inventory Protocols:

☑ ISO15693/NFC Type5
☑ ISO14443-A/NFC Type2 & Type4A
□ ISO14443-B/NFC Type4B
□ ISO14443-B/SRi/ST25TB
□ TYPE1/TOPAZ

1. Select a tag: E00226000161349A (ST25DV64K-I)

2. Select a command: Pick a picture to upload / Download a picture from the tag

□ Fast mode

3. Prepare the Discovery Kit firmware (Touch the FTM icon):

4. Select an action:

Picture:

Transfer progress:

Transfer time: 00:17:73

READER: Cr95hf CONNECTED

### Host to reader image upload

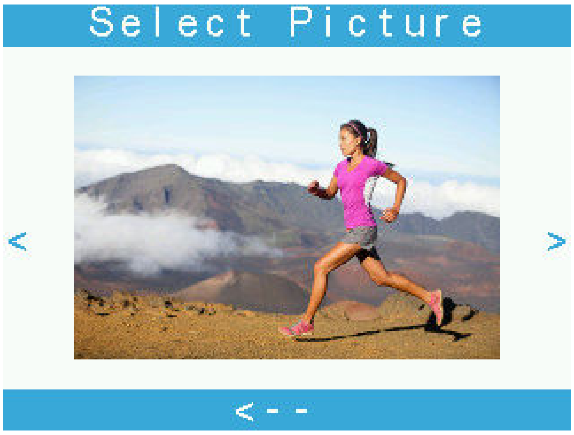
This function sends data from the STM32L476xx microcontroller device to the reader (H2R).

In this demonstration, the data is composed of jpeg images, it is written to the mailbox each time it is free, and the GPO returns the information to the MCU when the reader reads the mailbox message.

To start this demonstration the user must perform an action on the firmware side.

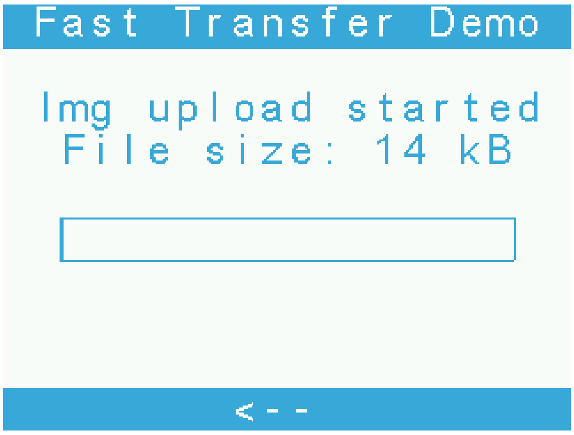
The picture icon at bottom left allows the user to enter a new menu (see Figure 48), which lets the user choose the pictures to send to the reader. By touching the screen on the left or right border the user scrolls between the different pictures available for transfer, simply touch the image on screen to select it. To come back and cancel the action, touch the arrow at the bottom of the screen.

Figure 48. ST25DV64KC-DISCO select picture to upload



After selecting the picture to transfer, the firmware writes the first chunk in the mailbox and waits for the message to be read. During the transfer, a file size and a progress bar are displayed, as shown in Figure 49.

Figure 49. ST25DV64KC-DISCO image upload start

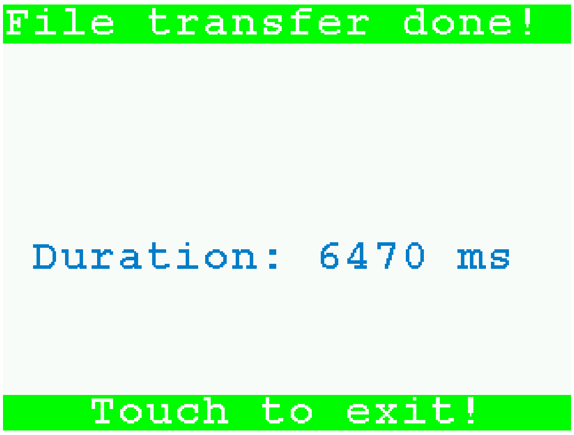

Fast Transfer Demo

Img upload started
File size: 14 kB

The kit is now waiting for the reader to read the message in the mailbox (see Smart phone and picture transfers and PC software and picture upload transfer sections) to continue the demonstration.

When the transfer is successfully completed, the transfer time is displayed on the screen (Figure 50). In case of failure, the display shows an error message.

Figure 50. ST25DV64KC-DISCO image upload successful


File transfer done!

Duration: 6470 ms

Touch to exit!

DT52479V2

To return to the Fast transfer mode demonstration simply touch the screen.

### Smart phone and picture download transfer

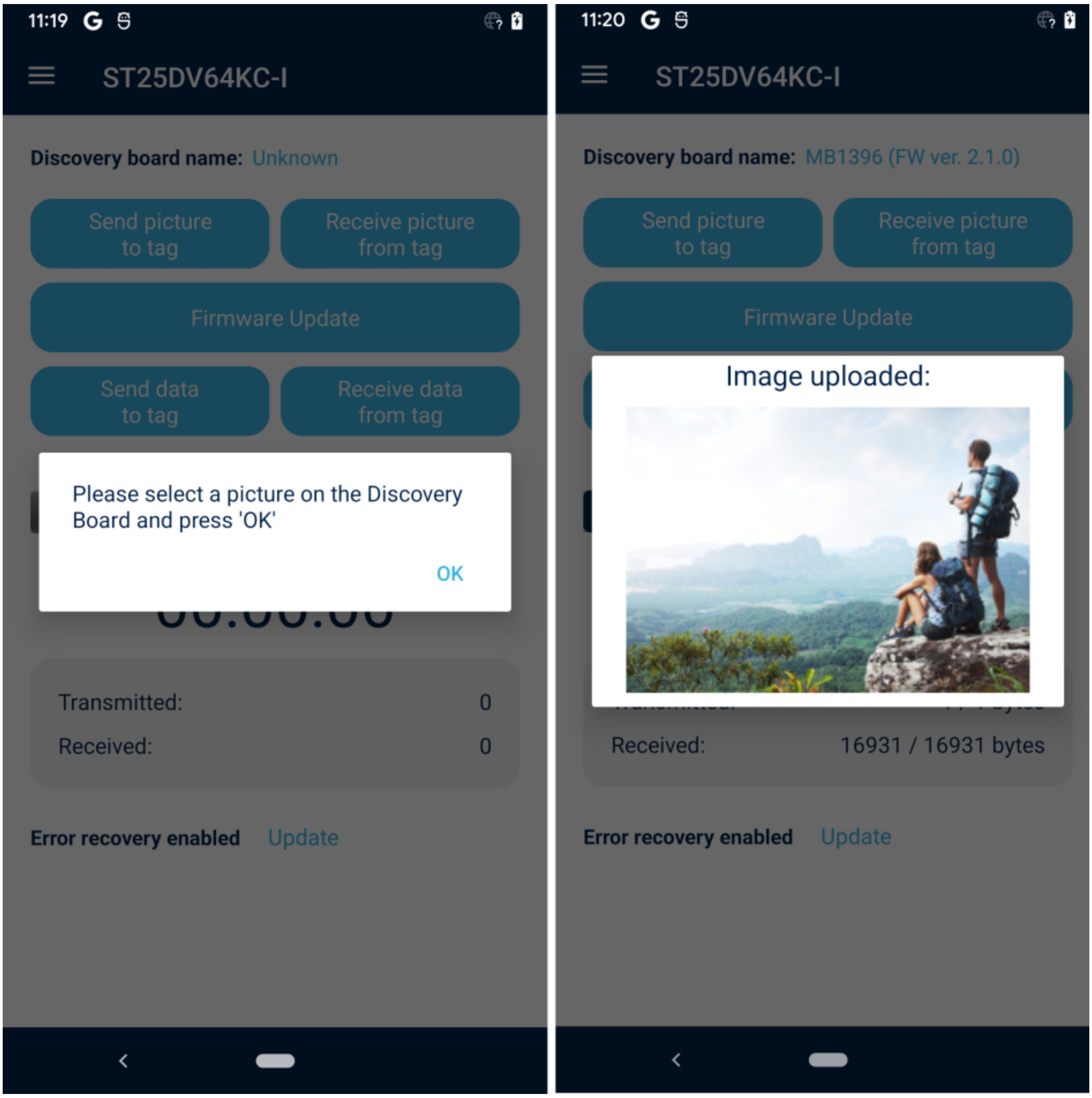
Features: same as Smart phone and picture transfers.

Picture download steps:

1. Check MB registers
2. Select “FTM demos” in the drawer menu
3. Received picture from tag button. Select a picture on the ST25 discovery kit. The transfer starts.

### Smart phone and picture upload transfers display

Figure 51. Smart phone and picture upload transfers display



Discovery board name: Unknown

Please select a picture on the Discovery Board and press 'OK'

Transmitted: 0

Received: 0

Error recovery enabled

Discovery board name: MB1396 (FW ver. 2.1.0)

Image uploaded:

Received: 16931 / 16931 bytes

Error recovery enabled

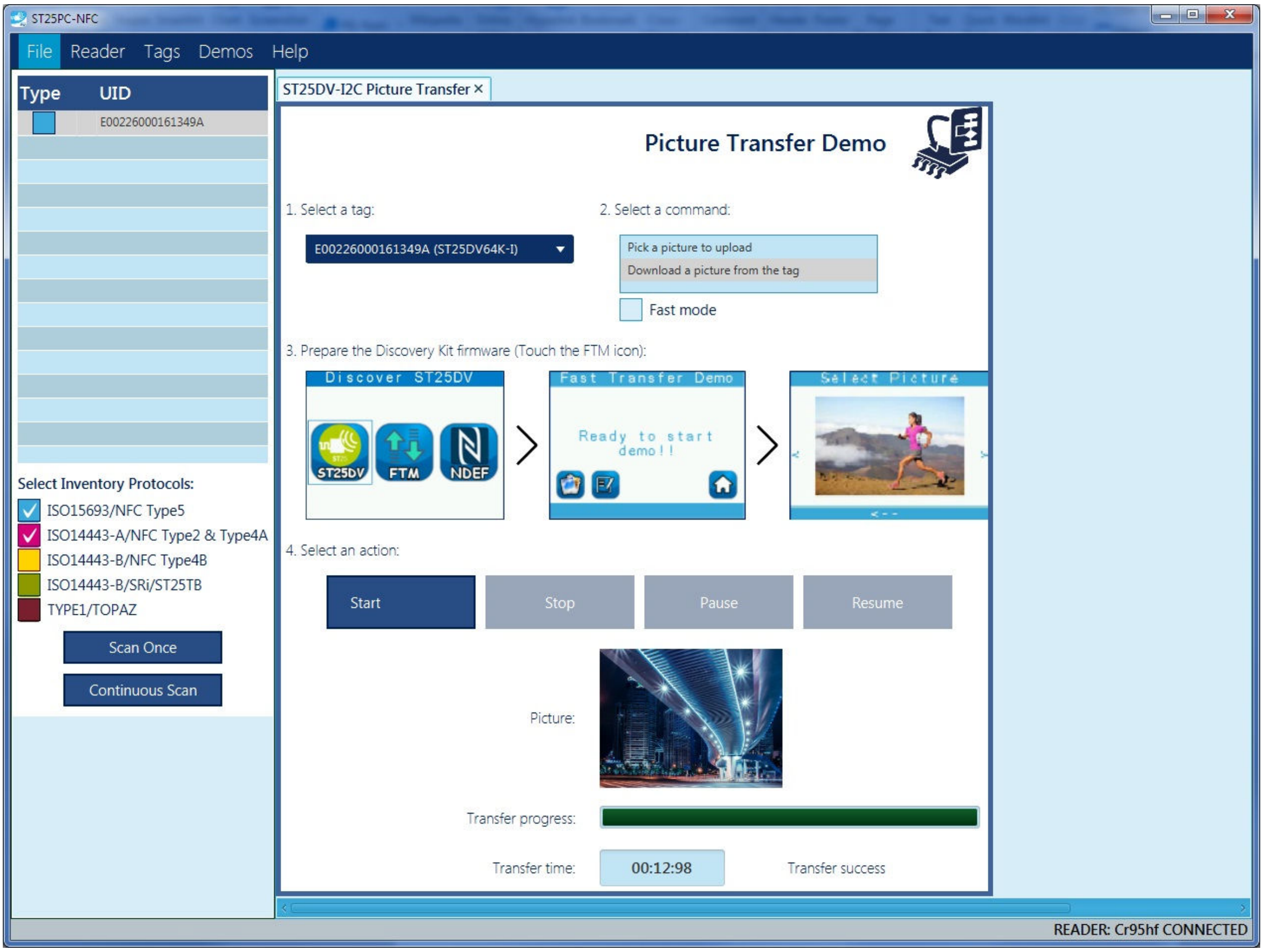
### PC software and picture upload transfer

In the ST25DV-I2C tag/ST25DV-DISCOVERY "Demos" menu (see Figure 33), select the "FTM – Picture Transfer" item. The H2R image transfer display is illustrated in Figure 52. PC software H2R image transfer display.

To initiate the transfer, choose a picture from the ST25DV64KC-DISCO board. Then click on the "Start" button to start the transfer from the board to the reader. CRC computation is done when all data is transferred, then it is sent to the tag.

The demonstration ends when the PC software receives the acknowledgment and displays the image .

**Figure 52.** **PC software H2R image transfer display**



| Type | UID |
|---|---|
| ▨ | E00226000161349A |

Select Inventory Protocols:

☑ ISO15693/NFC Type5
☑ ISO14443-A/NFC Type2 & Type4A
□ ISO14443-B/NFC Type4B
□ ISO14443-B/SRi/ST25TB
□ TYPE1/TOPAZ

1. Select a tag: E00226000161349A (ST25DV64K-I)

2. Select a command: Pick a picture to upload / Download a picture from the tag

□ Fast mode

3. Prepare the Discovery Kit firmware (Touch the FTM icon):

4. Select an action:

Picture:

Transfer progress:

Transfer time: 00:12:98 Transfer success

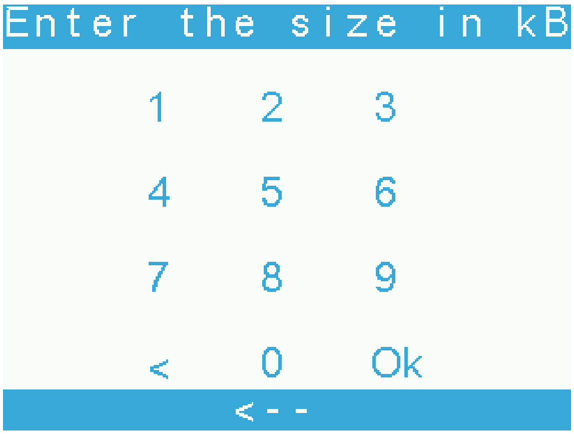
READER: Cr95hf CONNECTED

### Host to Reader data transfer

This function allows the user to send data from the STM32L476xx microcontroller device to the reader. In this demonstration, the data takes random binary values. Each time the mailbox is free the MCU writes data, and the GPO informs it when the reader consumes the mailbox message.

To start this demonstration the user needs to perform an action on firmware side. The data icon at bottom left allows the user to enter a new menu; the size of the data transfer (from the MCU to a reader) is selected from 1 to 999 kB, using the keyboard shown in Figure 53. The inputs are corrected by using the arrow on the bottom left. To cancel and return to the Fast transfer mode menu touch the arrow at the bottom of the screen. After entering the correct size, touch the "Ok" icon.

Figure 53. ST25DV64KC-DISCO select H2R data size to transfer display



Enter the size in kB

The kit is now waiting for a reader to read the message on the mailbox (see Smart phone and H2R data transfer section) to continue the demonstration.

When transfer is successfully completed, the display shows the computed CRC and transfer time (see Figure 50). In case of failure, the display shows an error message.

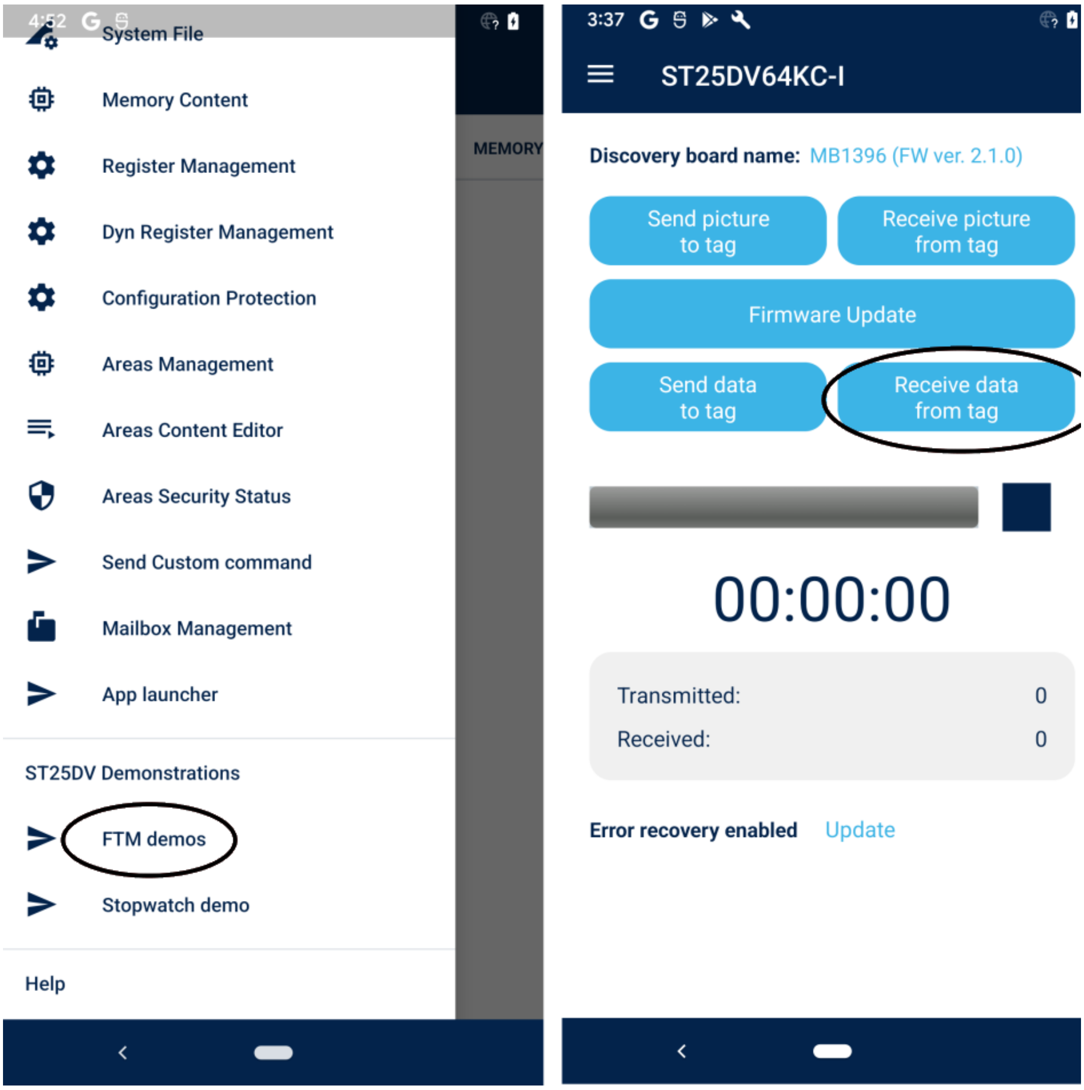
To return to the Fast transfer mode demonstration simply touch the screen.

### Smart phone and H2R data transfer

To start the demonstration after prerequisites setting checked, go to “FTM demos” menu.

After having selected the buffer size to send from ST25DV64KC-DISCO board, user can click on “Receive data from tag” button to start the transfer.

Figure 54. **Smart phone and data transfers H2R with selected size**



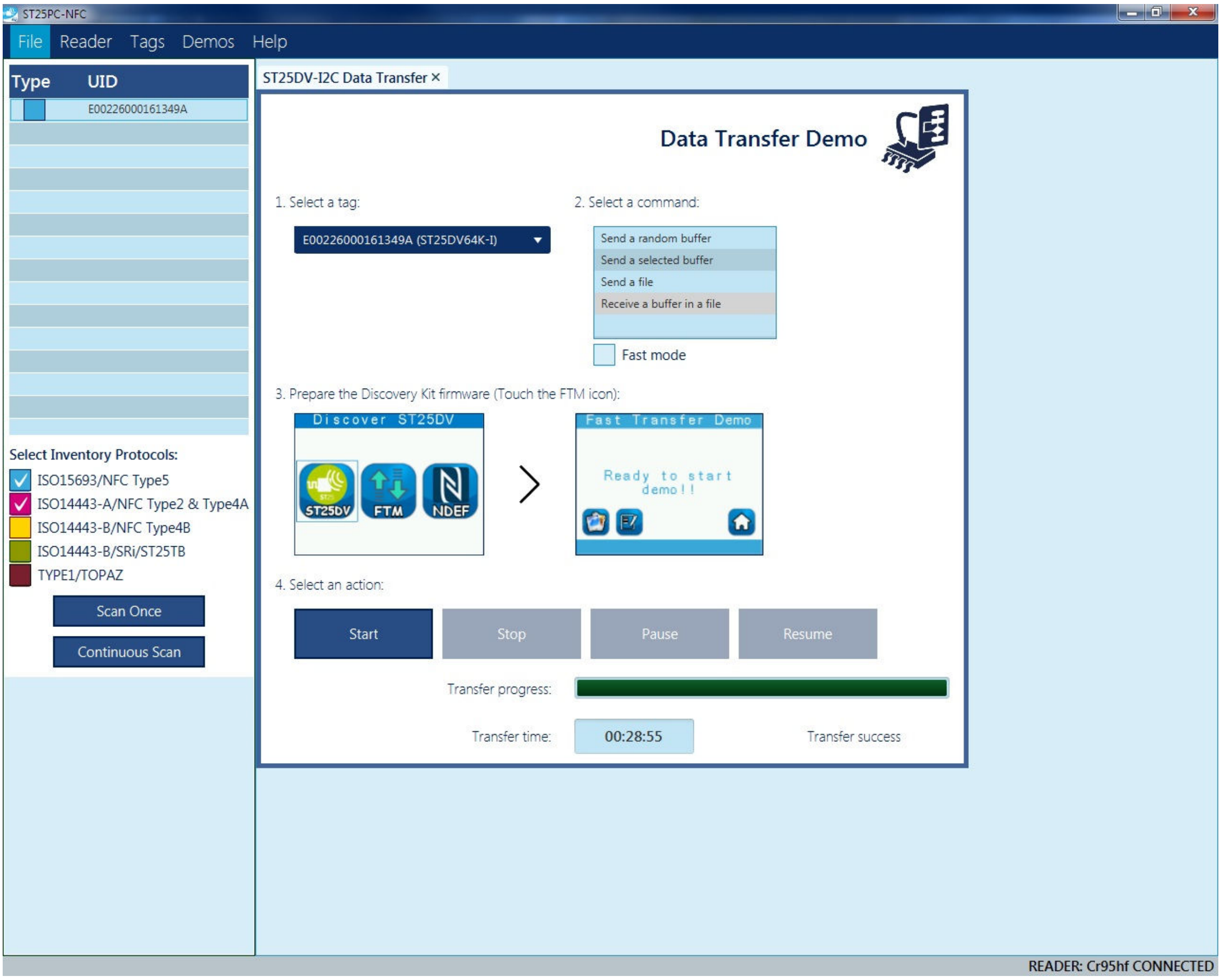
### PC software and H2R data transfer

In the ST25DV-I2C tag/ST25DV-DISCOVERY "Demos" menu (see Figure 33), select the "FTM – Picture Transfer" item.

To send data from the ST25DV64KC-DISCO board to the HF reader, select "Receive a buffer in a file" in the command menu.

Select the buffer size to be sent from the ST25DV64KC-DISCO board and click on the "Start" button to start the transfer of data from the board. CRC computation is done when all data is transferred, then it is sent to the tag.

Figure 55. PC software H2R data transfer display



| Type | UID |
|---|---|
| ■ | E00226000161349A |

Select Inventory Protocols:

- ☑ ISO15693/NFC Type5
- ☑ ISO14443-A/NFC Type2 & Type4A
- □ ISO14443-B/NFC Type4B
- □ ISO14443-B/SRi/ST25TB
- □ TYPE1/TOPAZ

1. Select a tag: E00226000161349A (ST25DV64K-I)
2. Select a command:
   - Send a random buffer
   - Send a selected buffer
   - Send a file
   - Receive a buffer in a file

   □ Fast mode
3. Prepare the Discovery Kit firmware (Touch the FTM icon):
4. Select an action:

Transfer progress:

Transfer time: 00:28:55 Transfer success

READER: Cr95hf CONNECTED

The demonstration ends when the PC software receives acknowledgment and saves the data in a file. A window pops up at the end of the transfer to select the file location.

### Stopwatch demonstration

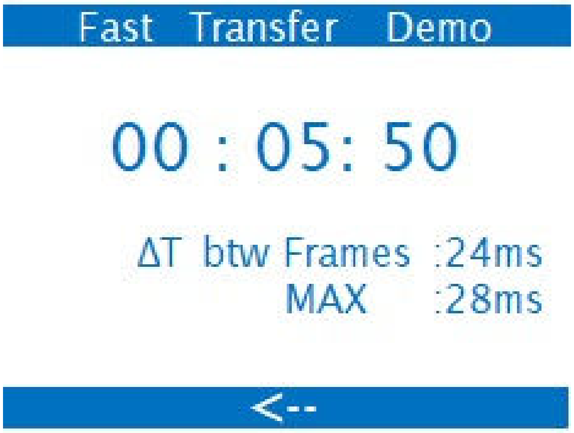
This demonstration shows how fast a smart phone updates the mailbox.

The smart phone sends, as fast as possible, stopwatch values (8-byte frames) to the MCU using the mailbox. The MCU displays these values on the LCD screen and computes the duration between two consecutive updates. The average and maximum inter-frame durations are refreshed and displayed after the reception of 20 frames.

To run the demonstration

1. On the ST25DV64KC-DISCO board, enter the Fast transfer mode menu.
2. On the smart phone: open the ST25 NFC App, tap the ST25DV-I2C tag and go to the "Stopwatch Transfers" menu and click on Start button.

Figure 56. ST25DV64KC-DISCO stopwatch start



Fast Transfer Demo

00 : 05: 50

ΔT btw Frames :24ms
MAX :28ms

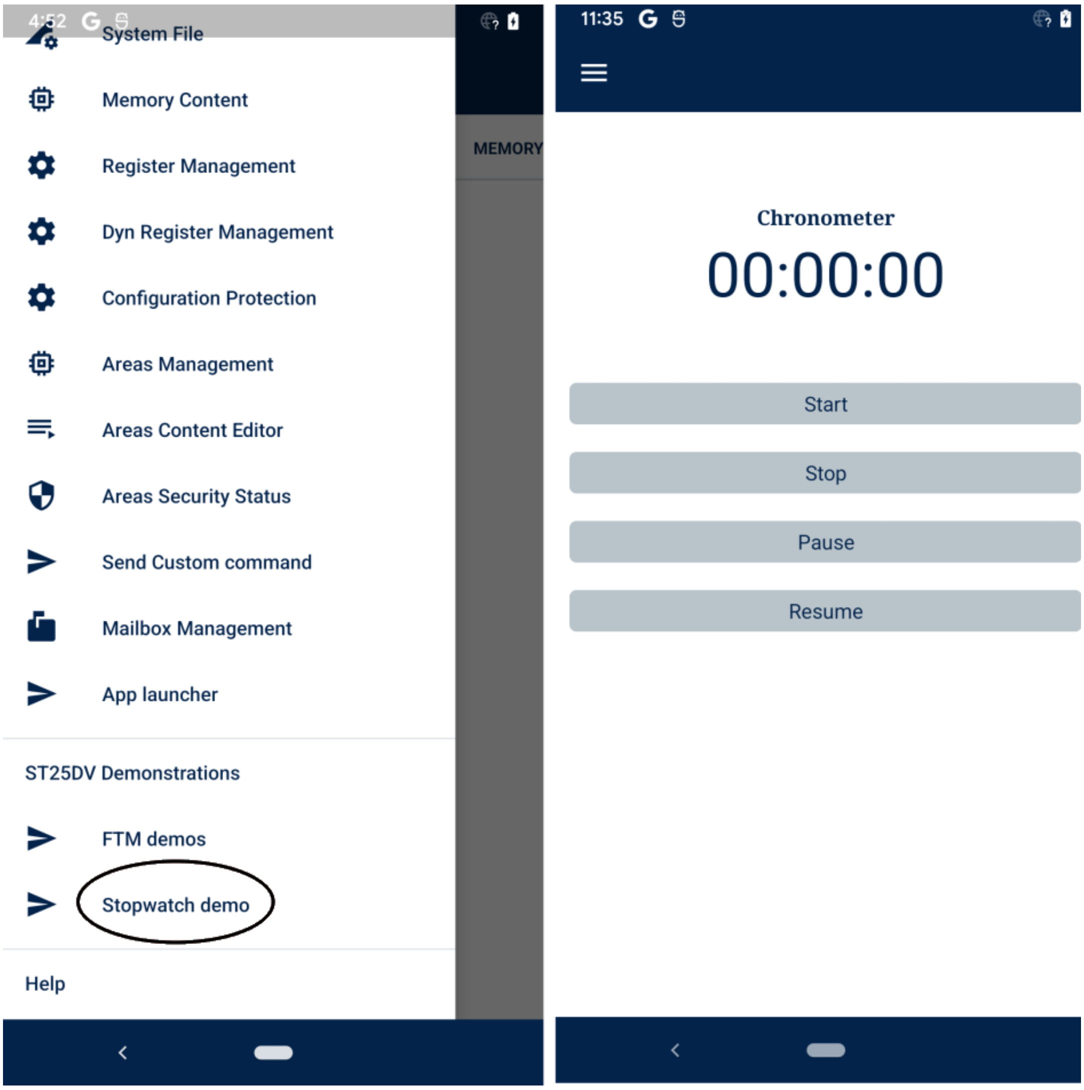
### Smart phone and stopwatch transfers

In this demonstration the smart phone sends, as fast as possible, stopwatch values (8-byte frames) to the MCU using the mailbox. The main features are

- Configuration of the “Mail Box” register [Enable / Disable MB, Watchdog)
  - Selection of static or dynamic registers
  - Read Cfg to retrieve MB status
- Start: Starts the stopwatch, the display is updated accordingly, and frames are sent to the host if a tag is in the field. The display is updated, starting from 00:00:00 (minutes:seconds:milliseconds)
- Stop: Stops the stopwatch. When restarted, the stopwatch restarts from 00:00:00.
- Pause: Pauses the stopwatch
- Resume: resumes the stopwatch after a pause

On the smart phone open the ST25 NFC application, tap the ST25DV and go to the “Stopwatch Transfers” menu and click on Start button. Pause and Resume buttons can then be used.

Figure 57. Smart phone and stopwatch transfers



### PC software and stopwatch demonstration

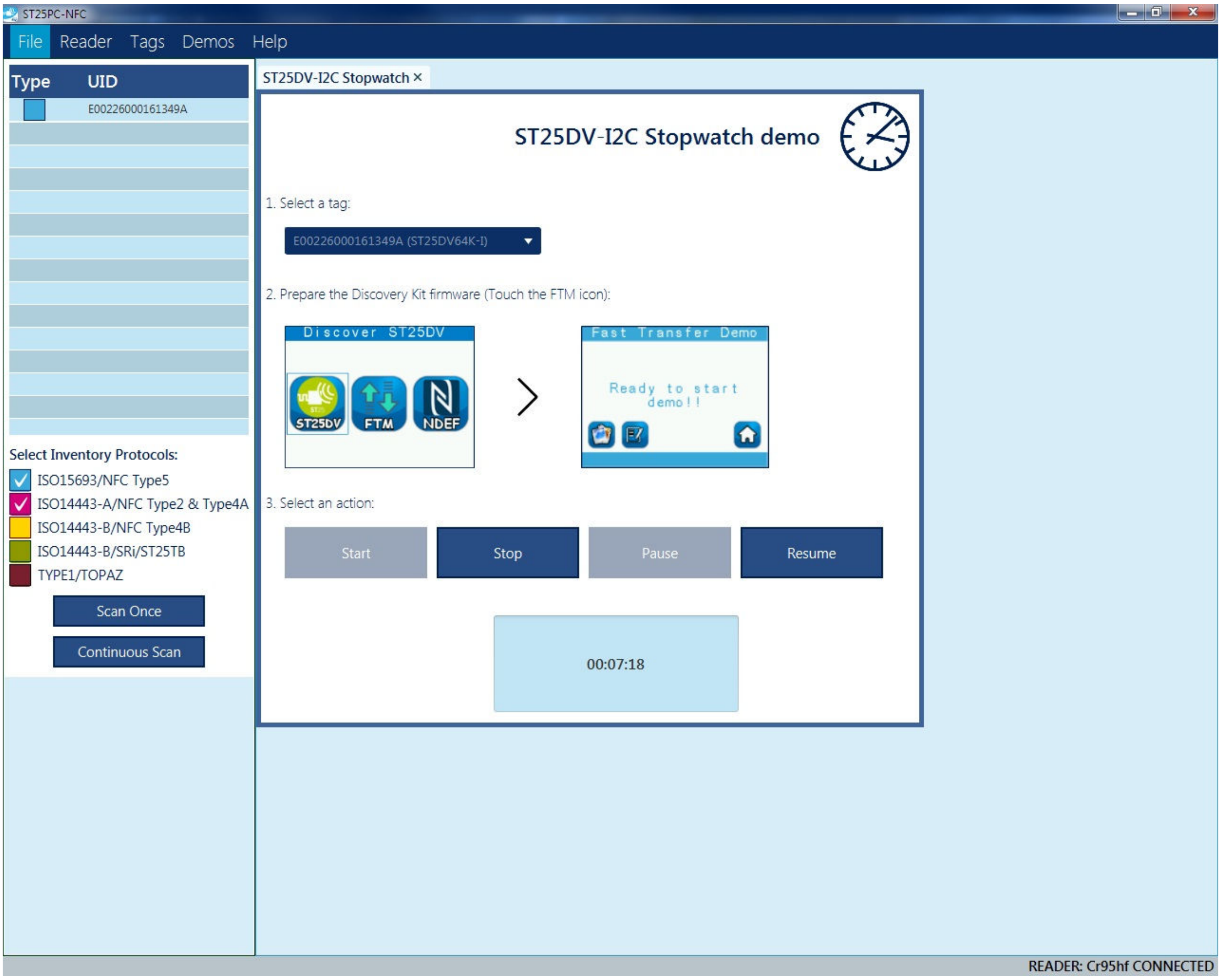
In the ST25DV-I2C tag/ST25DV-DISCOVERY "Demos" menu (see Figure 33), select the "FTM – Picture Transfer" item.

Click on the "Start" button to begin sending time data to the board.

Click on "Pause" to temporarily stop the timer. The "Resume" button starts the timer where it left off.

Click on "Stop" to definitely stop the timer.

Figure 58. ST25PC-NFC software and Stopwatch demonstration



| Type | UID |
|---|---|
| 🟦 | E00226000161349A |

Select Inventory Protocols:

☑ ISO15693/NFC Type5
☑ ISO14443-A/NFC Type2 & Type4A
□ ISO14443-B/NFC Type4B
□ ISO14443-B/SRi/ST25TB
□ TYPE1/TOPAZ

ST25DV-I2C Stopwatch demo

1. Select a tag:

E00226000161349A (ST25DV64K-I)

2. Prepare the Discovery Kit firmware (Touch the FTM icon):

3. Select an action:

00:07:18

READER: Cr95hf CONNECTED

# Appendix A FTM protocol details

## A.1 Introduction

The new FTM protocol has been introduced to bring the following benefits:

- Protocol is integrated into the ST25SDK so that every application, whether on Android™, iOS™, or PC, can use it. The previous protocol was fully managed by the applications.
- Simplify the use of the Fast transfer mode: a single API function takes care of the transmission, whatever the data size. The users do not need to split the data and neither do they need to acknowledge it anymore.
- Short overhead in order to optimize the data rate.
- Robust in case of errors or if the tag is moved out and back in the RF field. The transmission is be able to restart and to complete without error.

The NFC reader (for example a smart phone) is always the initiator of all communications.

The new protocol is available on MB1286 “ST25DV-Discovery board” and on MB1396 “ST25DV64KC-DISCO board”.

## A.2 Operating modes

The protocol is able to operate in two modes:

- A mode with error recovery: This mode is robust in case of transmission error or packet lost.
- A mode without error recovery. This mode gives the priority to the transfer speed but is not able to recover in case of error.

For both modes, the data to transfer must be split to fit in the ST25DV-I2C Mailbox (which can contain up to 256 bytes). «Packet» is called the amount of data that fits into the ST25DV mailbox.

Depending of the size of the data to send, there are one or more «packets» to transmit.

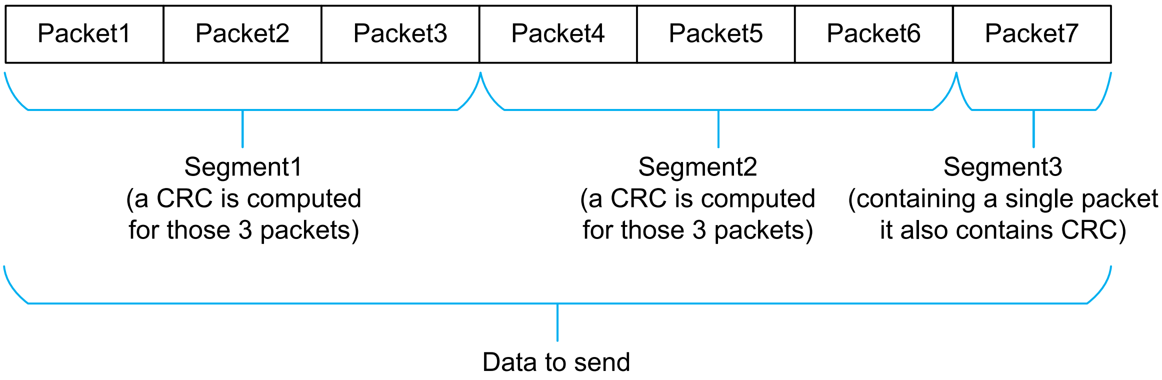
When sending big data files (like images or firmware data), it is interesting to detect transmission errors early and to be able to resent only the failing packets. The user could use a CRC for the whole file but in case of error, the user has to resend the whole file. The user could use a CRC to acknowledge each packet but this has a big impact on performance. Instead a CRC can be used to acknowledge a group of packets. This group of packets is called a “segment”.

A CRC is computed on all the data of the segment. If the CRC is correct, all the packets of the segment are considered as valid. Otherwise, an error is sent to the transmitter and all the packets of the current segment have to be resent.

Depending of the use case, the application has the possibility to change the number of packets per segment. In the ST25SDK, the default setting is 20 packets per segment (= 4900 Bytes per segment).

Example of transmission with three packets per segment:

Figure 59. **Example of transmission with three packets per segment**



MS54572V1

## A.3 Low-level protocol

A low-level protocol has been defined to:

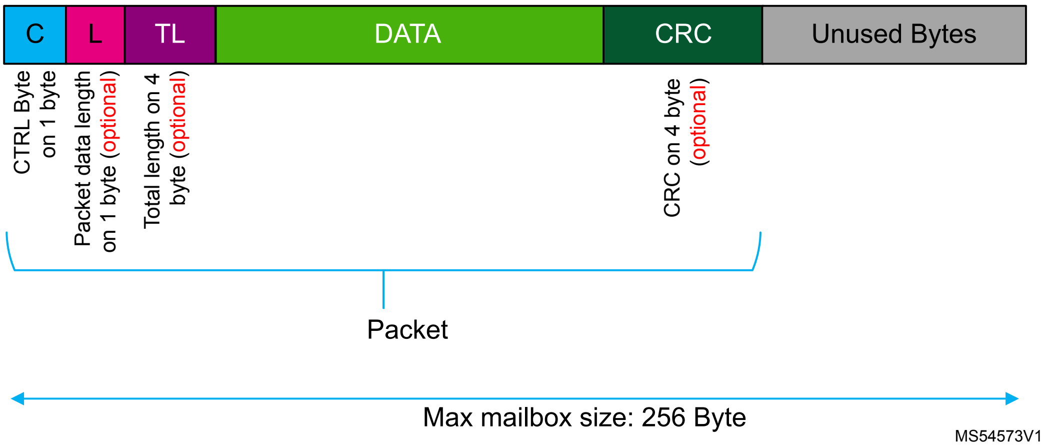
- Split the data to be sent into packets (a small header is added to indicate the content of the packet)
- Ensure their transmission
- Possibly: check the validity of the data received and acknowledge it

This protocol allows user to send data of any size, both ways.

Each packet starts with a “Control Byte” or a “Status Byte”:

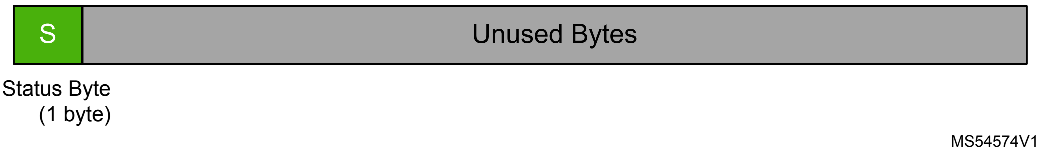
1. A “Control Byte” is used to indicate the bytes present in the packet (packet data length, total, data length, CRC presence and so on).

**Figure 60. Structure of a packet**


C
L
TL
DATA
CRC
Unused Bytes
CTRL Byte on 1 byte
Packet data length on 1 byte (optional)
Total length on 4 byte (optional)
CRC on 4 byte (optional)
Packet
Max mailbox size: 256 Byte
MS54573V1

2. A “Status Byte” is a single Byte sent in response to one or more received packets to indicate if the transmission was successful or not.

**Figure 61. Structure of a status packet**



The following table describes the content structure of the control byte.

**Table 3. Control byte structure definition**

| Bit number | Description |
|---|---|
| Bit 7 | This bit must be set to 0 for a control byte (it is used to distinguish control bytes and status bytes). |
| Bit 6 | Bit indicating if a packet data length field (on one byte) is present. It indicates the length of the data present in this packet (including the CRC bytes, if any). If no packet data length field is present, the receiver considers that the packet has the max possible size. |
| Bit 5 | Segment End. When this bit is set, a CRC is present in this packet. The CRC is calculated on all the data present between “Segment Start” and “Segment End”. An acknowledge must be sent by the receiver. |
| Bit 4 | Segment Start |

| Bit number | Description |
|---|---|
| Bit 3 - 2 | • Bits used to indicate data split on several packets. They have the following meaning:<br>– b00: Data fitting in a single packet.<br>– b01: First packet of a serie of packets. In that case, a total length field (on 4 bytes) is present. It indicates the length of the data that has been split on several packets.<br>– b10: Middle packet (there can be more than one).<br>– b11: End packet finishing the transfer of the data. |
| Bit 1 | Parity bit toggling for every new segment. It allows to detect the re-transmission of the same segment |
| Bit 0 | Bit indicating if the packet belongs to a segment |

The following table describes the content structure of the status byte.

Table 4. **Status byte structure**

| Bit number | Description |
|---|---|
| Bit 7 | This bit must be set to 1 for a status byte (to avoid confusion with control bytes) |
| Bit 6 - 4 | Reserved |
| Bit 3 - 0 | Status of the last packet(s) received:<br>• 00: SEGMENT_OK<br>• 01: SEGMENT_ERROR (Ex: CRC error, packet lost…etc) |

So a status byte with the value 0x80 corresponds to SEGMENT_OK and 0x81 corresponds to SEGMENT_ERROR.

## A.4 ST Light Protocol

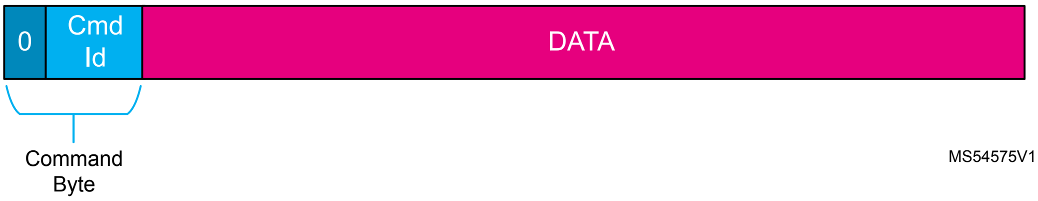
On top of previous low level protocol, a light protocol is used to indicate the content of the sent data. This light protocol is used to send commands that can be understood by the receiver.

It is not mandatory to use this light protocol. Users can use their own protocol if they prefer.

In this protocol, the data sent are called “Commands”.

A command starts with a byte called “cmdId” and can be followed by some data:

Figure 62. **Structure of a command**



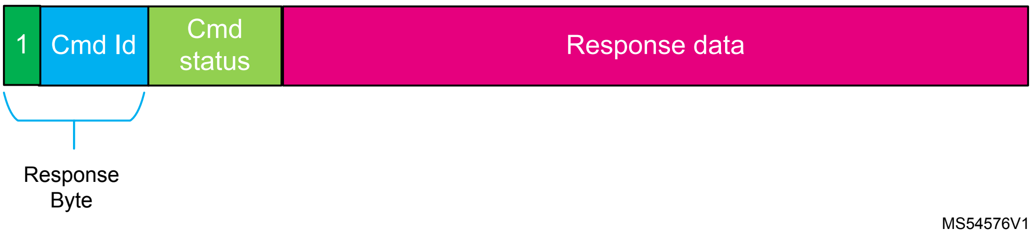
**Warning:** *The most significant bit of the command_id must be set to 0.*

Response format:

- The response must start with a response byte that is (0x80 | cmdId). It allows the user to check that the response really matches the command sent. The most significant bit is set to 1 to indicate that it is a response and not a command.

- Then there is a command status:
  - CMD_OK (=0x81) in case of success
  - CMD_ERROR (=0x82) in case of error
  - CMD_INTERNAL_ERROR (=0x83) in case of internal error (not due to the protocol).
  - CMD_UNKNOWN (= 0x84)
  - CMD_NOT_ALLOWED (= 0x85): Command not allowed
- Then there might be some "Response data"

Figure 63. **Structure of a command response**



Note: *Beware of the possible confusion between "status bytes" used for acknowledging (SEGMENT_OK = 0x80, SEGMENT_ERROR = 0x81) and Command Status (CMD_OK = 0x81, CMD_ERROR = 0x82).*

The following commands are currently used:

Table 5. **FTM commands list**

| Cmd Id | Command name | Data | Response data |
|---|---|---|---|
| 0x00 | FTM_GET_BOARD_INFO | Data must contain 2 bytes (0x00, 0x00) | The response data contains 4 bytes: one indicating the board name and 3 indicating the firmware version |
| 0x01 | FTM_SEND_PICTURE | Data must contain the picture | Null |
| 0x02 | FTM_READ_PICTURE | Null | The response data contains the picture read |
| 0x03 | FTM_STOPWATCH | Data must contain the stopwatch data | Null |
| 0x04 | FTM_FW_UPGRADE | Data must contain the firmware to send | Null |
| 0x05 | FTM_SEND_DATA | Data must contain the data to send | Null |
| 0x06 | FTM_READ_DATA | Null | The response data contains the data read |
| 0x07 | FTM_SEND_PASSWORD | Data must contain the password to send to grant the Firmware upgrade | Null |
| 0x0F | ECHO<br>(used only for debug) | The first byte indicates if the STM32 must send a response with or without error recovery (0x00 = without error recovery, 0x01 = with error recovery).<br>The other bytes can be chosen by the issuer | The response data contains the same bytes has in the command data |

## Revision history

Table 6. **Document revision history**

| Date | Revision | Description of changes |
|---|---|---|
| 03-Fev-22 | 1 | Initial release. |
| 14-Jun-2024 | 2 | Updated:<br>• Figure 4. ST25DV64KC-DISCO main menu display<br>• Figure 10. ST25DV64KC-DISCO energy harvesting measurement display<br>• MyApp record demonstration in Section 4.2.7: NFC NDEF demonstration menu<br>– Figure 24. ST25DV64KC-DISCO MyApp demonstration screen<br>• Section 4.2.8: Fast transfer mode demonstration menu<br>– Figure 39. ST25DV64KC-DISCO firmware upgrade: password OK<br>– Figure 41. ST25DV64KC-DISCO software transfer progress status display<br>– Figure 42. ST25DV64KC-DISCO firmware upgrade: transfer done<br>– Figure 50. ST25DV64KC-DISCO image upload successful |

## Contents

# List of tables

# List of figures