UM1795
User manual

EVAL-ST95HF firmware functionalities

Introduction

This document describes the firmware functionalities of the EVAL-ST95HF board (STSW-ST95HF001). It has the goal of helping the user to understand how this firmware works, by describing the main functionalities with simple examples.

The ST95HF is an RFID transceiver. It supports ISO/IEC 14443A, ISO/IEC 14443B, ISO/IEC 15693 and ISO/IEC 18092 in reader mode (PCD). It also supports ISO/IEC 14443A in card emulation mode (PICC).

The MCU of the EVAL-ST95HF is the STM32F103. The ST95HF, together with the microcontroller, emulates a PCD or a PICC. The MCU communicates with the ST95HF by SPI bus.

The application is divided in three parts, one for PCD functions (read or write tags), one for PICC functions (tag emulation, file transfer) and one for peer to peer functions.

The application has the capability to read and write TT1, TT2, TT3, TT4A, TT4B and vicinity card. It can also emulate TT2 and TT4A tags for the storage of NDEF messages.

Contents

List of tables

List of figures

1 Acronyms and notational conventions

1.1 Acronyms

- APB: Advanced Peripheral Bus
- AAR: Android Application record
- CAN: Controller Area Network
- GPS: Global Positioning System
- IEC: International Electrotechnical Commission
- I²C: Inter-Integrated Circuit
- ISO: International Organization for Standardization
- MCU: Micro Controller Unit
- NFC: Near Field Communication
- RF: Radio Frequency
- RFID: Radio Frequency Identification
- RISC: Reduced Instruction Set Computer
- SPI: Serial Peripheral Interface
- USB: Universal Serial Bus

1.2 Representation of numbers

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

Binary numbers

Binary numbers are represented by strings of digits 0 and 1 shown with the most significant bit (MSB) on the left, the least significant bit (LSB) on the right, and a “0b” added at the beginning.

Example: 0b11110101.

Hexadecimal numbers

Hexadecimal numbers are represented by using the numbers 0 to 9 and the characters A – F, and adding an “0x” at the beginning. The Most Significant Byte (MSB) is shown on the left and the Least Significant Byte (LSB) on the right.

Example: 0xF5.

Decimal numbers

Decimal numbers are represented as is without any trailing character.

Example: 245.

2 Overview

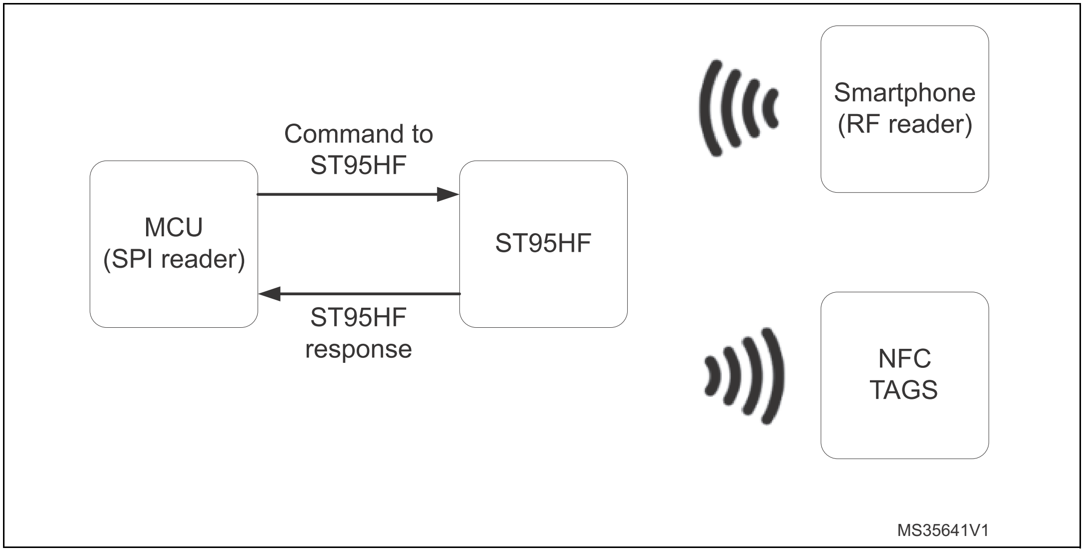
2.1 ST95HF overview

ST95HF is a transceiver for contactless application. It includes frame coding and RF modulation, thus allowing the connected MCU to send and receive NFC commands in the supported protocols.

ST95HF is a slave device, hence an host (MCU) is required to control it.

The ST95HF is connected to the MCU using an SPI communication. The ST95HF is able to act as a PCD or a PICC so it can interact with a tag or with a reader, as exemplified in *Figure 1*.

Figure 1. Functional block diagram



For more details concerning the ST95HF device, please refer to its datasheet.

2.2 STM32F103 overview

The STM32F103xx incorporates the high-performance ARM® Cortex®-M3 32-bit RISC core operating at a 72 MHz frequency, high-speed embedded memories (Flash memory up to 1 Mbyte and SRAM up to 96 Kbytes), and an extensive range of enhanced I/Os and peripherals connected to two APB buses. All devices offer two 12-bit ADCs, three general purpose 16-bit timers plus one PWM timer, as well as standard and advanced communication interfaces: up to two I2Cs and SPIs, three USARTs, an USB and a CAN.

These features make the STM32F103xx microcontrollers suitable for a wide range of applications such as motor drives, application control, medical and handheld equipment, PC and gaming peripherals, GPS platforms, industrial applications, PLCs, inverters, printers, scanners, alarm systems, video intercoms, and HVACs.

2.3 EVAL-ST95HF board

The EVAL-ST95HF is a kit which allows to evaluate the ST95HF transceiver performance.

The EVAL-ST95HF is powered through the USB bus and no external power supply is required. It includes a ST95HF, a 47x34 mm 13.56 MHz simple layer inductive etched antenna and its associated tuning components.

By default, the ST95HF communicates with the STM32F103RG 32-bit MCU via the SPI bus.

Pictures of the board are shown in *Figure 2* and *Figure 3*.

Figure 2. EVAL ST95HF board (front side)

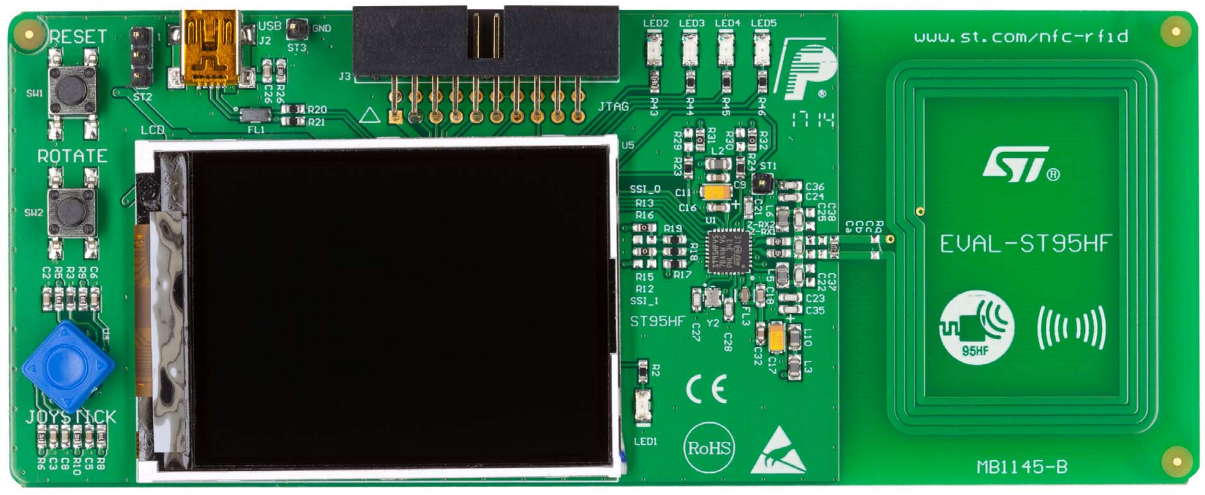
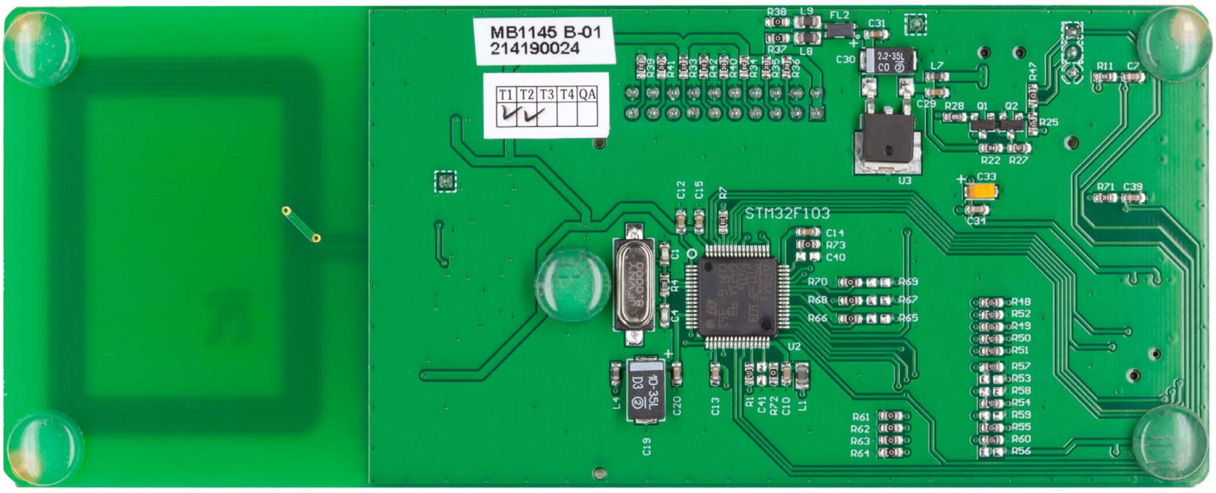


Figure 3. EVAL ST95HF board (back side)



3 Firmware description

The ST95HF component can act both as a PCD or as a PICC.

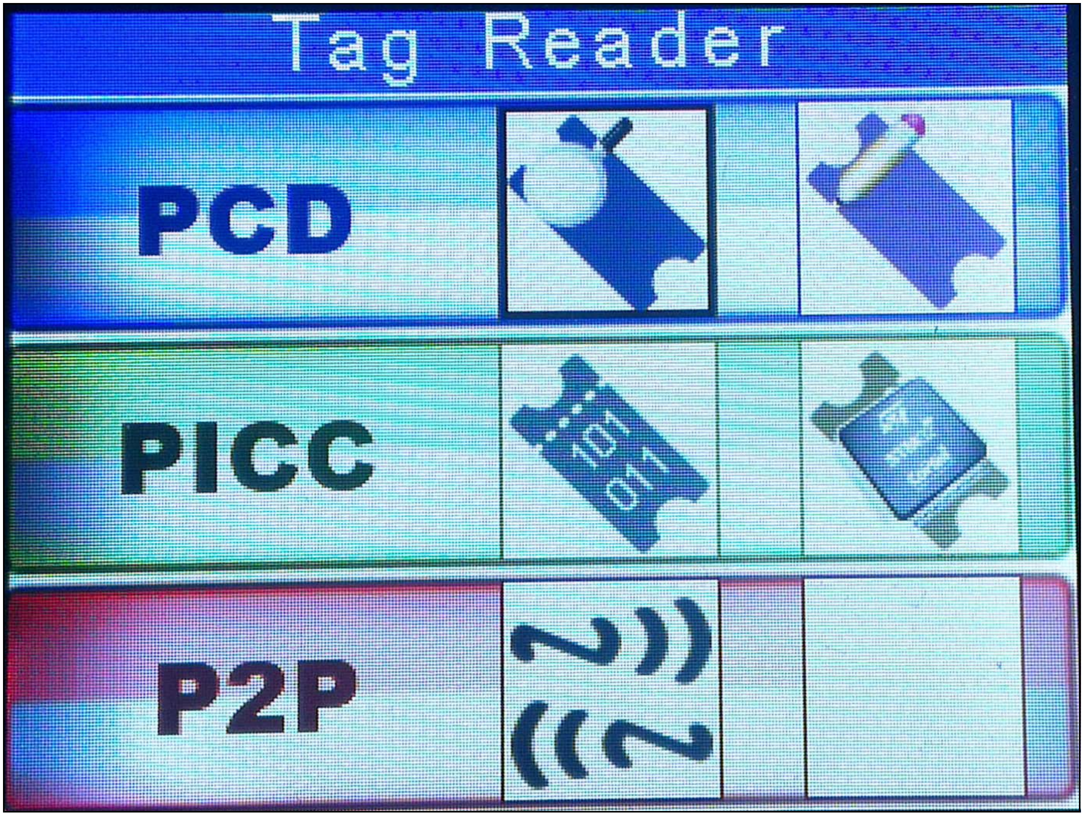
The main purpose of the firmware is to show all of these available functionalities. Accordingly, the menu is divided in three parts, one for PCD functions, one for PICC functions and the last one for Peer to Peer functions.

Note: *In order to test the Peer to Peer mode it is necessary to have two ST95HF boards. All the other functions can be tested with an NFC phone, a tag or another ST95HF board.*

3.1 The main menu

The main menu is made up of icons allowing access to different sub menus. As can be seen from *Figure 4*, they are disposed in three lines in order to easily identify the active mode.

Figure 4. Main menu



The main functionalities associated with each operation mode are listed in *Table 1*.

Table 1. Supported operation modes and functionalities

Mode	PCD	PICC	P2P
Functionalities	Tag reader: – Tag Hunting – Tag Detect wake-up – Tag Reading – PC link mode Tag writer: – TEXT – GEO – ...	Tag Emulator: – TT2 – TT4A Card Emulator: – File transfer	Proprietary Peer to Peer: – PP2P Pong (Client) – PP2P Pong (Server) – PP2P Data transfer / IO Remote control (Client) – PP2P Data transfer / IO Remote control (Server)

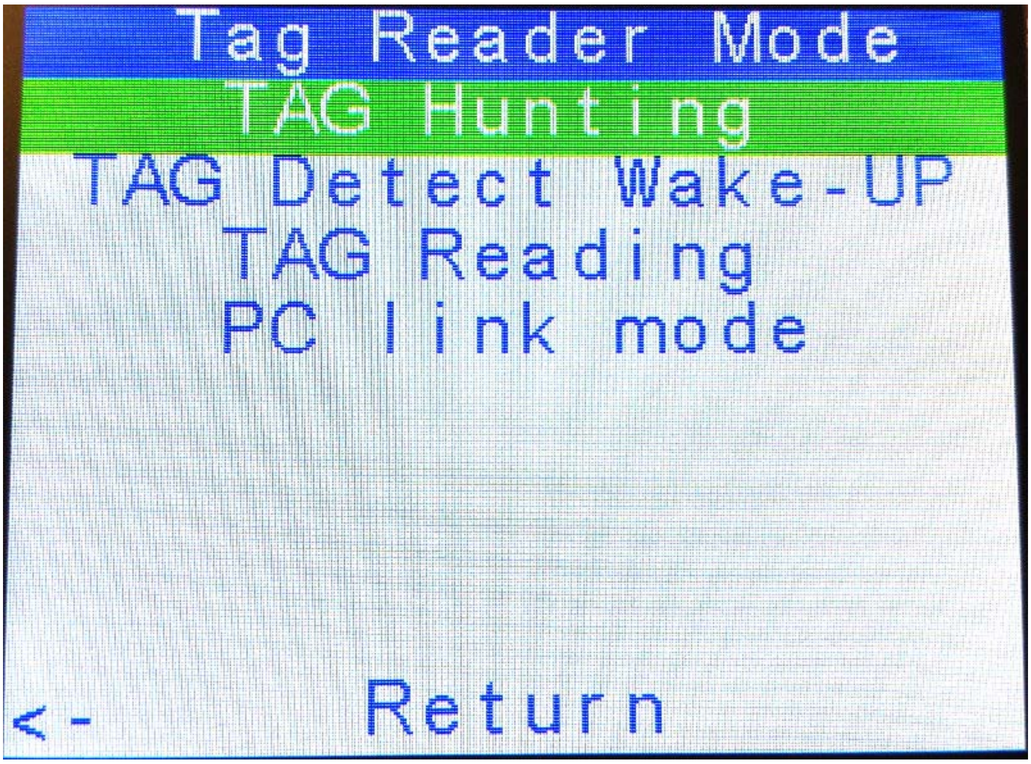
To select a sub menu, user should move the black square on the chosen icon using the joystick, and press it to select the item. Every sub menu has a return option to go back to the previous menu.

Any menu or sub menu can be rotated using the rotate button, so that it will be possible to use the board in the other direction. In this case, the joystick too will be adapted (buttons reversed). It is not possible to rotate the screen when a mode is selected, this is feasible only inside a menu.

3.2 Tag reader

In this menu there are four different modes, as can be seen from *Figure 5*.

Figure 5. Tag reader menu

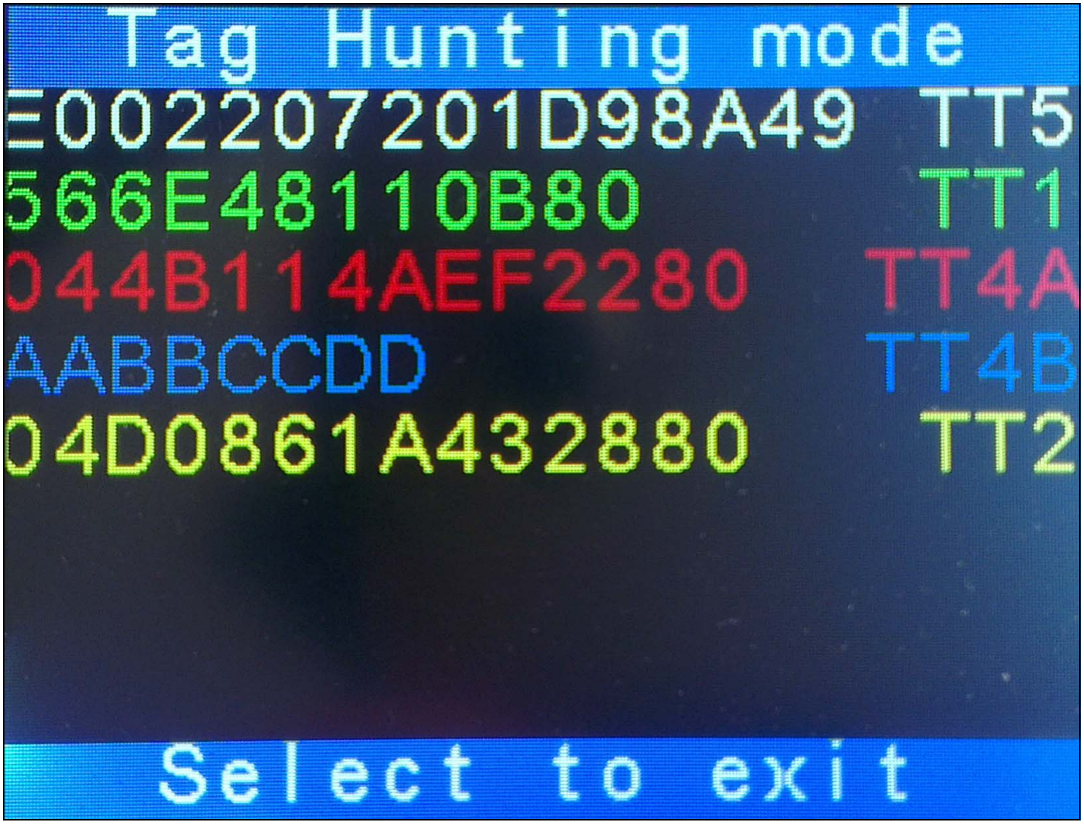


3.2.1 Tag Hunting

This mode is used to list all the found tags: when a tag is close enough to the antenna, the UID and the tag type is written on the screen (see *Figure 6*).

Note: If the same tag is re-detected by the tag hunting it will refresh the line (this feature can help to check the detection range).

Figure 6. Example of tags found

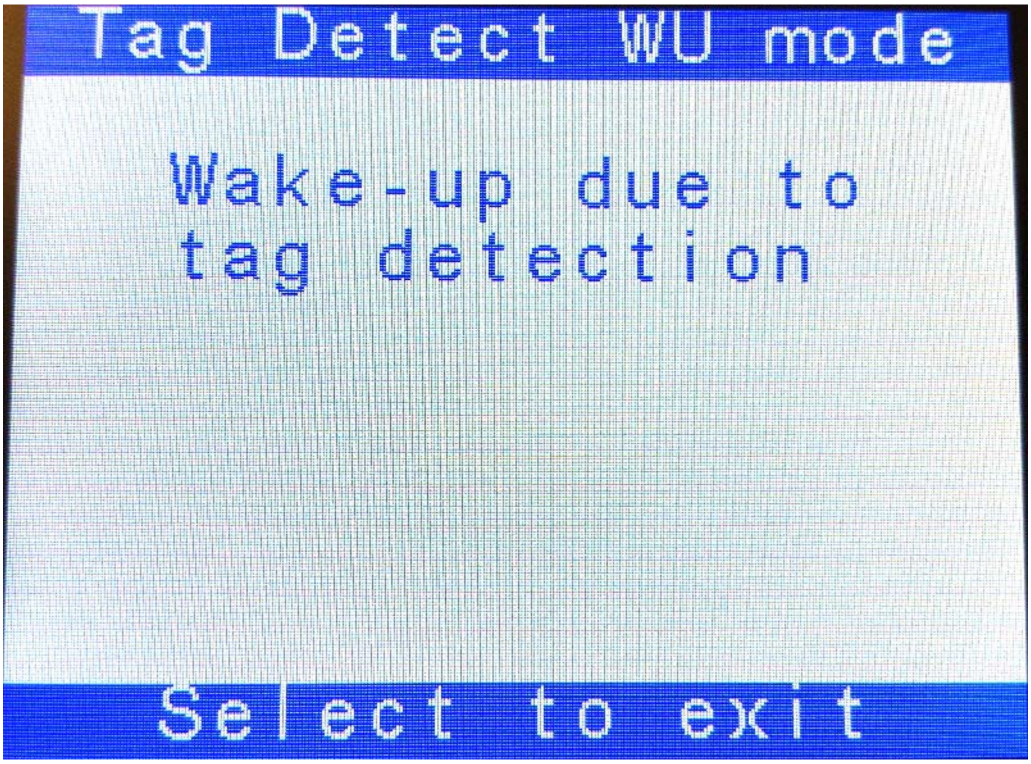


```
Tag Hunting mode
E0022207201D98A49  TT5
566E48110B80       TT1
044B114AEF2280    TT4A
AABBCCDD          TT4B
04D0861A432880     TT2
Select to exit
```

3.2.2 Tag Detect wake-up

This mode (see *Figure 7*) is used to wake-up the ST95HF component with a tag from the Low power mode. An antenna calibration will be processed before to go in idle mode (low power mode). The wake-up of the ST95HF will be done when a tag is close enough to the antenna.

Figure 7. Tag Detect wake-up



Tag Detect WU mode

Wake-up due to tag detection

Select to exit

3.2.3 Tag reading

This mode is used to read the NDEF content of a tag. When a tag is close enough to the antenna, its content is decoded and printed on the screen (see *Figure 8*).

The reader is able to decode following NDEF messages:

- Empty
- URI
- SMS
- Email
- Geo position
- Text
- VCARD.

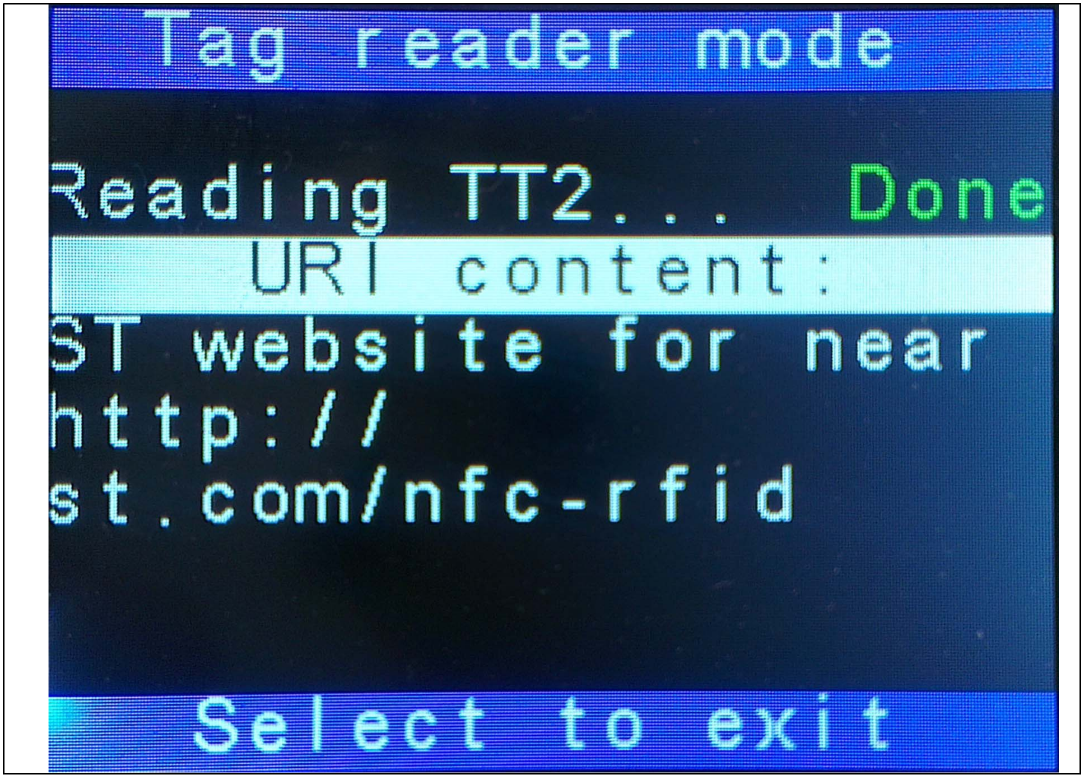
The message can be formatted inside a smart poster in order to add an additional Text field to the message.

Errors can be printed if there is a transmission error or an error during the decoding of a message.

Note: VCARD contains too much information to be printed on the screen, so only a message "VCARD detected" is shown, however all the decoding is done inside the program.

Note: The same tag is not read two times in a row.

Figure 8. Reading tags

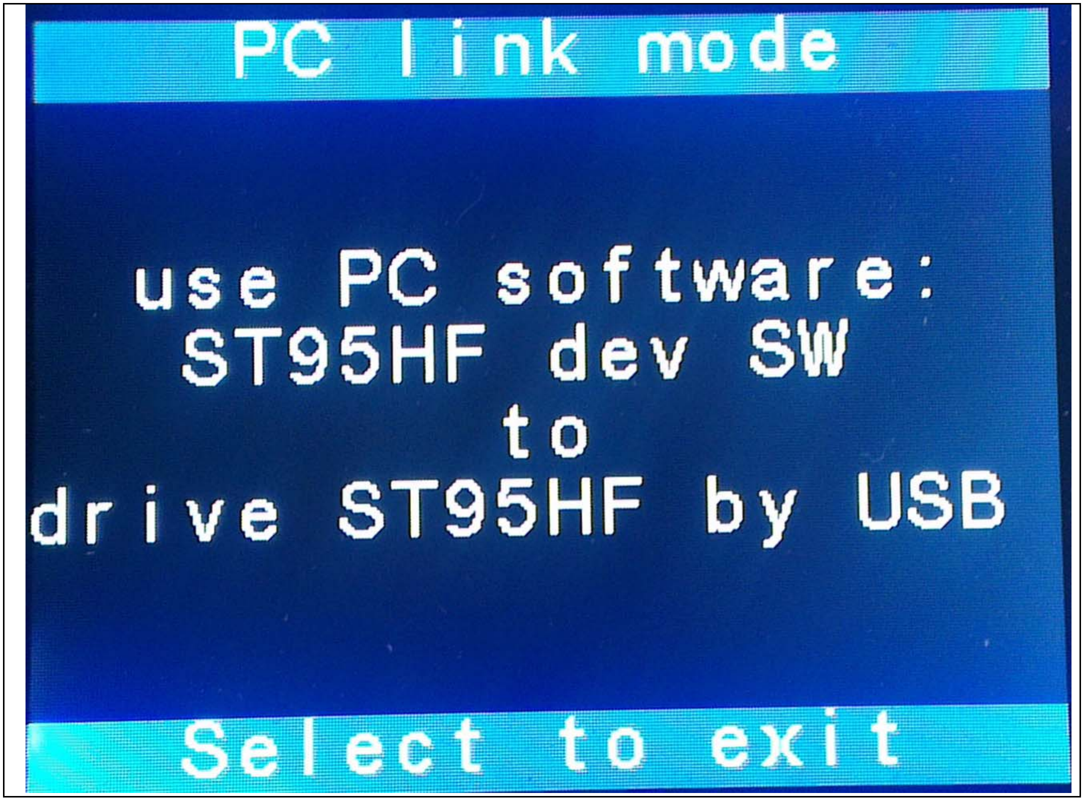


```
Tag reader mode

Reading TT2...   Done
URI content:
ST website for near
http://
st.com/nfc-rfid

Select to exit
```

3.2.4 PC link mode

This mode allows the EVAL-ST95HF to work like the CR95HF board and to be controlled using the CR95HF_Development_Software (see *Figure 9*). The computer interacts with the board using the USB cable.

Figure 9. Board linked to a PC



PC link mode

use PC software:
ST95HF dev SW
to
drive ST95HF by USB

Select to exit

3.3 Tag writer

This mode is used to write an NDEF message to a tag, the first step here is to select the preformatted message to write. The right column corresponds to the size needed to write the message (an example is shown in *Figure 10*).

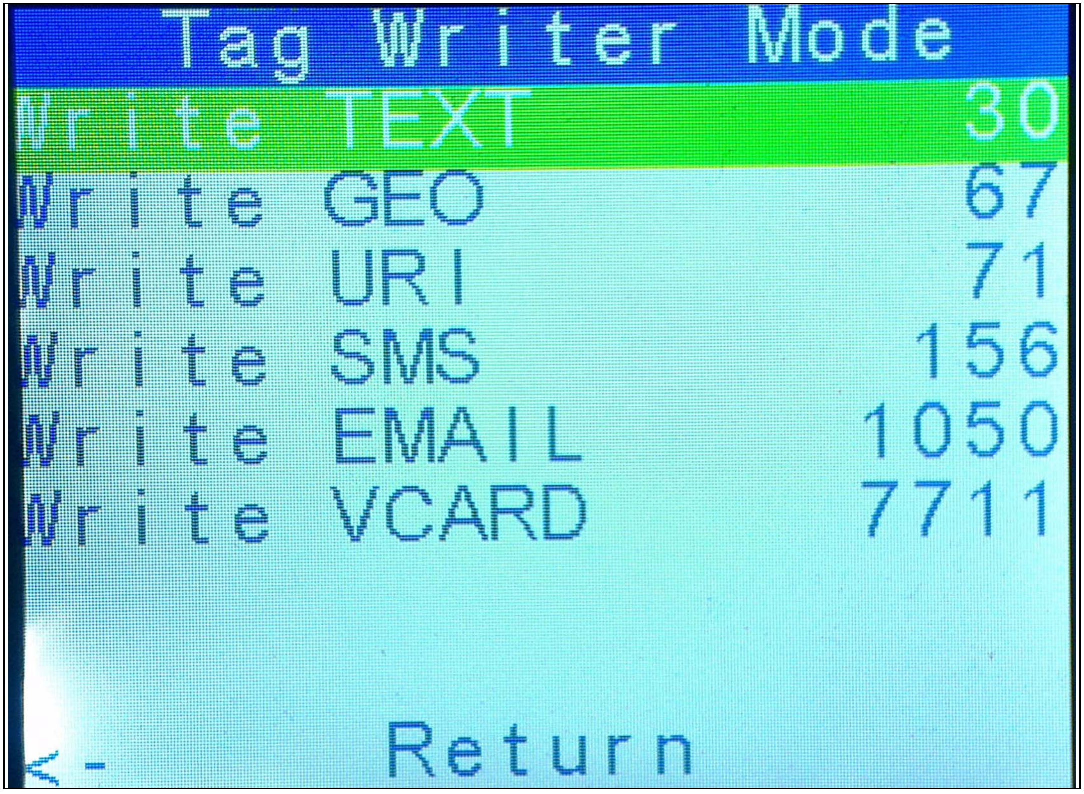
Then the message can be written by putting the tag close to the antenna. Error messages can be written to the screen in the following cases:

- Not enough memory inside the tag: the message cannot be written because the tag inside the field is too small.
- Tag locked: the read or write access is not allowed.
- Transmission error: when the tag is removed before the end of write operation a transmission error is returned.

Note: The same tag cannot be written two times in a row.

Note: The firmware is able to format vicinity card if the tag does not contain any NDEF data but it will not try to format other tag types.

Figure 10. Tag writer mode



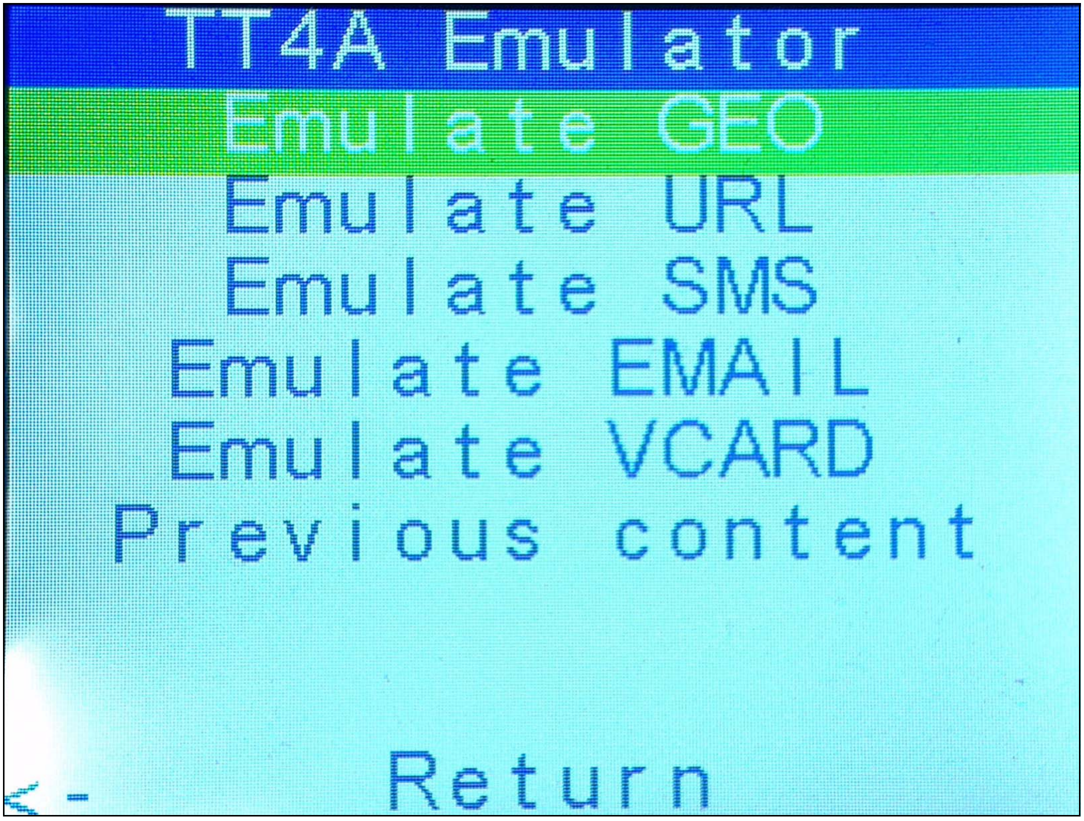
```
Tag Writer Mode
Write TEXT        30
Write GEO         67
Write URI         71
Write SMS        156
Write EMAIL     1050
Write VCARD     7711

<-      Return
```

3.4 Tag emulator

In this menu (see *Figure 11*) it's possible to select the kind of tag (TT2 or TT4A) to be emulated and then default content can be chosen.

The memory available is 492 bytes for the TT2, and 7998 bytes for the TT4A. These values can be changed inside the source code, it's worth noting that some smartphones do not support TT2 sizes over 1024 bytes.

Figure 11. Emulator mode



Once the content is chosen, the emulation starts, and a PCD (e.g. a smartphone) can read the content as if it were a classic tag. It is also possible to read the content using another EVAL-ST95HF in reader mode.

User can write content with the smartphone. When the emulation exits, the content of the tag is written inside the MCU flash, allowing it to be restored using the “Previous content” choice.

Note: If the board is powered off or reset during the emulation, the current content is lost because the content will be written to the flash only when the emulation is stopped.

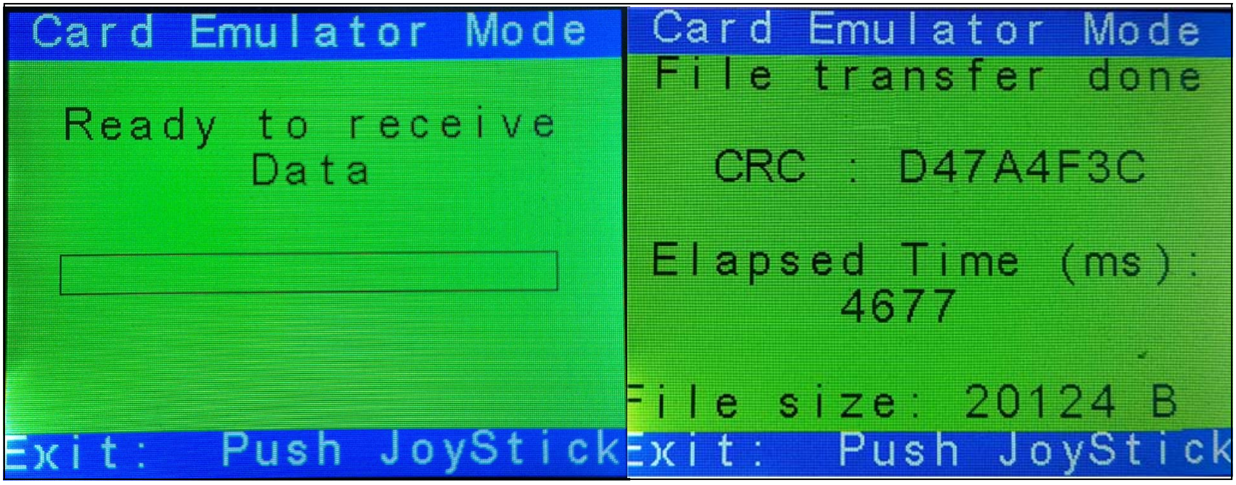
3.5 Card Emulator

3.5.1 File transfer

This mode is used to transfer a file from an NFC phone to the EVAL-ST95HF board (see *Figure 12*).

In order to do that the phone must have the correct application which supports this proprietary transfer. When the board is configured in card emulator mode it is seen as a classic TT4A tag containing an AAR to the correct application.

Figure 12. File transfer



Card Emulator Mode

Ready to receive
Data

Exit: Push JoyStick

Card Emulator Mode
File transfer done

CRC : D47A4F3C

Elapsed Time (ms):
4677

File size: 20124 B
Exit: Push JoyStick

The progress bar will be refreshed during the transfer and at the end of the transfer a CRC will be calculated in order to check if the file is not corrupted. The elapsed time is written and the total size of the file sent.

Doing this we can evaluate the speed to transfer a file using ST95HF.

Note: The result can change between two tries even with the same file and same smartphone as smartphone data bandwidth is unpredictable (multi task OS).

Note: The application on the EVAL-ST95HF will not try to interpret the data sent.

3.6 Proprietary Peer to Peer

This is a proprietary peer to peer demonstration between two EVAL-ST95HF boards. It uses NFC-DEP functions for the Pong demo and ISO-DEP functions for the Prop P2P demo, but all data transferred are formatted using a proprietary way.

3.6.1 Initialization

When the menu Peer to Peer is selected a choice between four different initializations is proposed (see *Figure 13*):

Prop P2P Pong Config Client

This menu is used to force the application to act as the client. It will wait infinitely for a server (a second EVAL-ST95HF board set in Prop P2P server).

Prop P2P Pong Config Server

This menu is used to force the application to act as the server. It will wait infinitely for a client (a second EVAL-ST95HF board set in Prop P2P client).

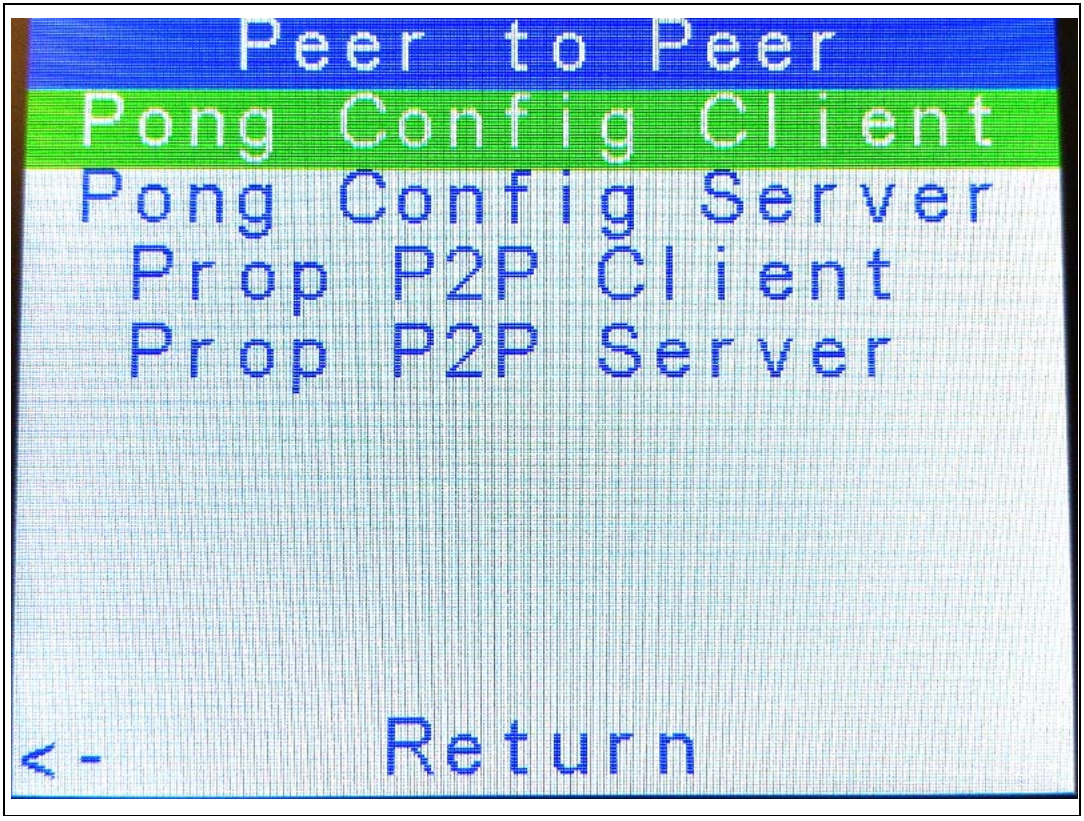
Prop P2P Client

This menu is used to force the application to act as the client. It will wait infinitely for a server. The server can be a second EVAL-ST95HF board set in Prop P2P server. This application allows to make some Data transfer or remote control (such as Data transfer, IO control, etc...).

Prop P2P Server

This menu is used to force the application to act as the server. It will wait infinitely for a client (A second EVAL-ST95HF board set in Prop P2P client). This application allows to make some Data transfer or remote control (Data transfer, IO control, etc...).

Figure 13. P2P mode

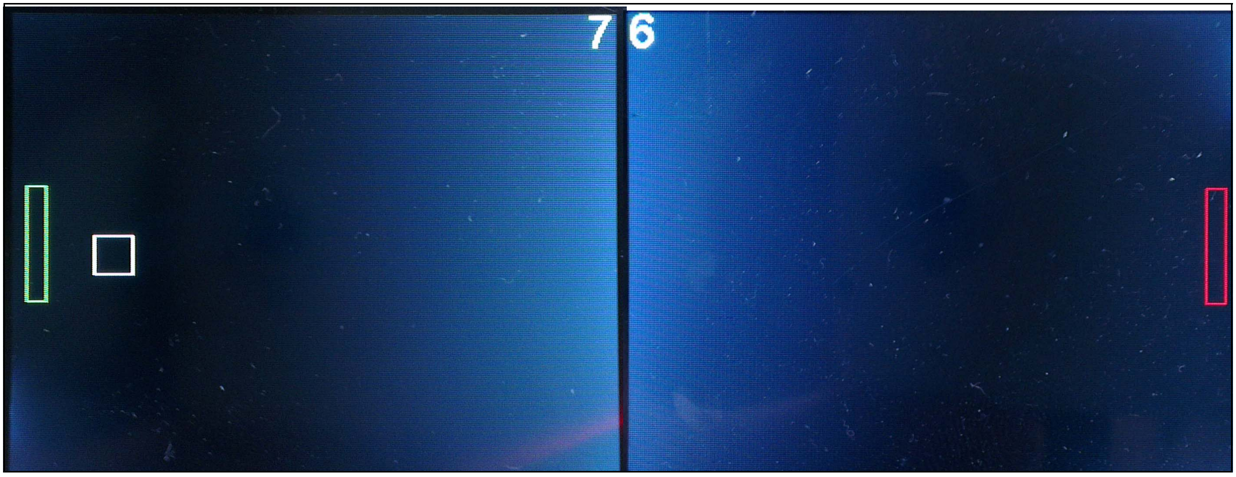


3.6.2 Prop P2P Pong

Description

As can be seen from *Figure 14*, the application is a simple tennis game in order to show the peer to peer communication. When the initialization step is done, one board is defined as the initiator (Green racket) and the other as the target (Red racket).

Figure 14. Tennis game



The racket can be moved using the up and down buttons of each board. The goal is to avoid the ball (the white square) hitting the border of the screen. Every time a point is scored, the player's score is increased. When the score exceeds 9, it restarts from 0.

The score can be displayed at the top or at the bottom of the screen by pressing the rotate button from any board. The information is transmitted to the other board using the peer to peer communication.

The speed of the game can be reduced by pressing the left key of the initiator and increased by pressing the right key (by default it is set to the maximum speed).

If the transmission is broken by spacing the two boards it will suspend the game and will restart in the same state when the communication will be reestablished.

How it works

The server manage all the game, including moving the ball with bounces on walls or on rackets. It works like if the game had a 640x240 screen for the storage of the ball position. It also deals with the score calculation.

The server is continuously sending data to the client containing the following information:

Score2	Xmsb	Xlsb	Y	ScreenRotated

The Score2 byte contains the score for the client. It is send in order to be printed by the client on its screen.

The Xmsb and Xlsb bytes are used to store the horizontal position of the ball. The client will print the ball on its screen only if the coordinate is between 320 and 640 (on its part of the global screen).

The Y byte contains the vertical coordinate of the ball. It is used also in order to print the ball at the good position.

The ScreenRotated field is used to know if the server's rotate button has been pressed or not. It toggles every time the button is pressed, so the other board can know if it has to update its own screen.

The client answers to this frame by sending back two values::

Racket 2 Y	ScreenRotated2

The Racket 2 Y stores the position of the client’s racket. In fact the server needs it to compute bounces and to manage scores.

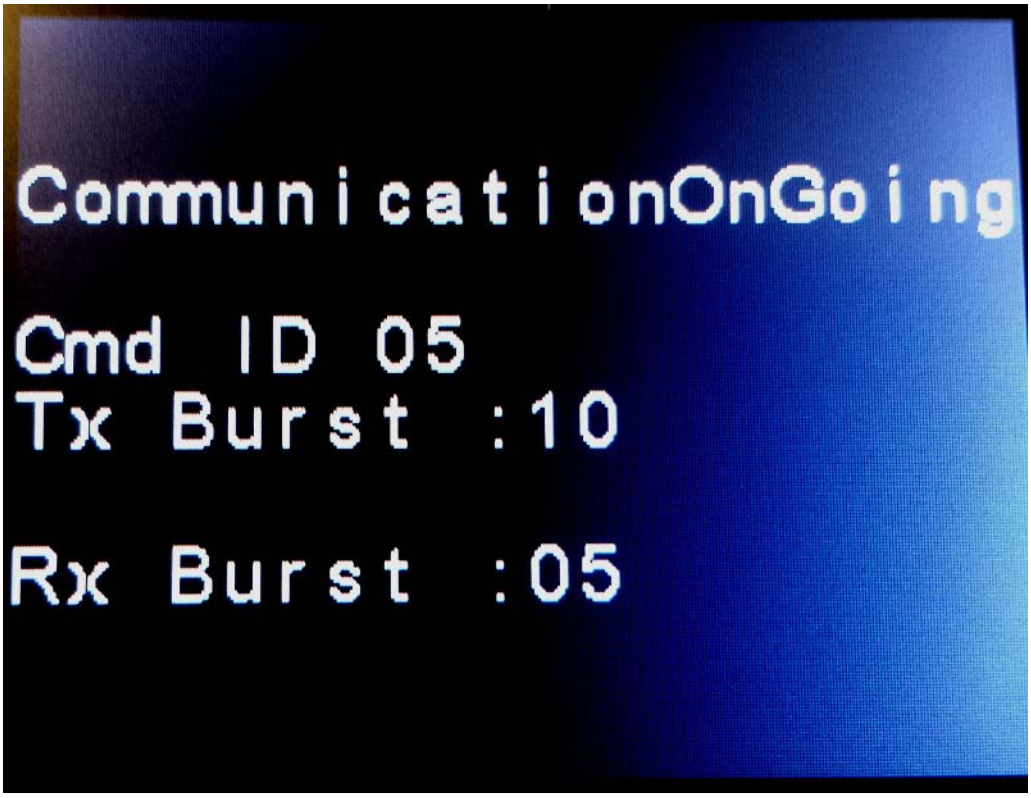
The ScreenRotated2 works like the ScreenRotated parameter but this time it is for the client’s rotate button.

3.6.3 Prop P2P Data transfer

Description

This application demonstrates how to send commands through the peer to peer communication. When the initialization step is done and communication is established, one board is set as server and the other as client, the message "CommunicationOnGoing" is displayed on both screens (see *Figure 15*).

Figure 15. Example of P2P communication



CommunicationOnGoing

Cmd ID 05

Tx Burst :10

Rx Burst :05

Then user can start sending commands by pressing each directional key of both server and client. Commands details is described below.

Commands sent by the Server

- *Up key*

  Server sends the burst config to the Client, the Client displays the number of bursts (10 bursts) and the quantity of data by burst (56 bytes) on screen.
- *Down key*

  Server sends the command to toggle the Client led 5.
- *Right key*

  Server sends 10 Bursts, the Client displays the number of received Bursts on screen.

Commands sent by the Client

- *Up key*

  Client sends command ID 05 (start/stop camera), a message "CAM STARTED" is displayed on the Server screen.
- *Down key*

  Client sends 10 Tx Burst and displays the number of transmitted Bursts on screen.
- *Right key*

  Client sends command ID 08 (simulate FW upload). Server displays "FIRMWARE UPGRADE" on screen.
- *Left key*

  Client sends command ID 07 (simulate wifi pairing). Server displays "WIFI PAIRING DONE" on screen.

4 Revision history

Table 2. Document revision history

Date	Revision	Changes
25-Jun-2014	1	Initial release.
20-Jul-2015	2	Updated *Table 1: Supported operation modes and functionalities*, *Figure 5: Tag reader menu* and *Figure 13: P2P mode*. Updated *Section 3.2: Tag reader*, *Section 3.6: Proprietary Peer to Peer*, *Section 3.6.1: Initialization* and *Section 3.6.2: Prop P2P Pong*. Added *Section 3.2.2: Tag Detect wake-up*, *Section 3.6.3: Prop P2P Data transfer* and *Figure 15: Example of P2P communication*.

Rev 2

IMPORTANT NOTICE – PLEASE READ CAREFULLY