Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

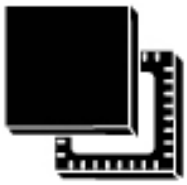

## Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

# ST95HF

## Near field communication transceiver

**Datasheet - production data**



**VFQFPN32 5x5 mm**

## Features

- Belonging to ST25 family, that includes all NFC/RF ID tag and reader products from STMicroelectronics
- Operating modes supported:
  - Reader/Writer
  - Card Emulation (ISO/IEC 14443-3 Type A)
- Hardware features
  - Dedicated internal frame controller
  - Highly integrated Analog Front End (AFE) for RF communications
  - Transmission and reception modes
  - Optimized power management
  - Tag Detection mode
  - Field Detection mode
- RF communication @13.56 MHz
  - ISO/IEC 14443 Type A and B in Reader and Card Emulation modes
  - ISO/IEC 15693 in Reader mode
  - ISO/IEC 18092 in Reader and Card Emulation modes
  - MIFARE® Classic compatible
- Communication interfaces with a Host Controller
  - Serial peripheral interface (SPI) Slave interface up to 2 Mbps
  - Up to 528-byte command/reception buffer (FIFO) depending on communication protocol
- 32-lead, 5x5 mm, very thin fine pitch quad flat (VFQFPN) ECOPACK®2 package

## Applications

Typical protocols supported:

- ISO/IEC 14443-3 Type A and B tags
- ISO/IEC 15693 tags
- ISO/IEC 18000-3M1 tags
- NFC Forum tags: Types 1, 2, 3 and 4
- ST Dual Interface EEPROM

Typical ST95HF applications include:

- Consumer electronics
- Gaming
- Healthcare
- Industrial

Typical ST95HF use cases include:

- NFC-enabled Wi-Fi pairing
- NFC-enabled Bluetooth pairing
- Data exchange

Communications with NFC/RFID tag (reader mode)

# Contents

# List of tables

# List of figures

# 1 Description

The ST95HF is an integrated transceiver IC for contactless applications.

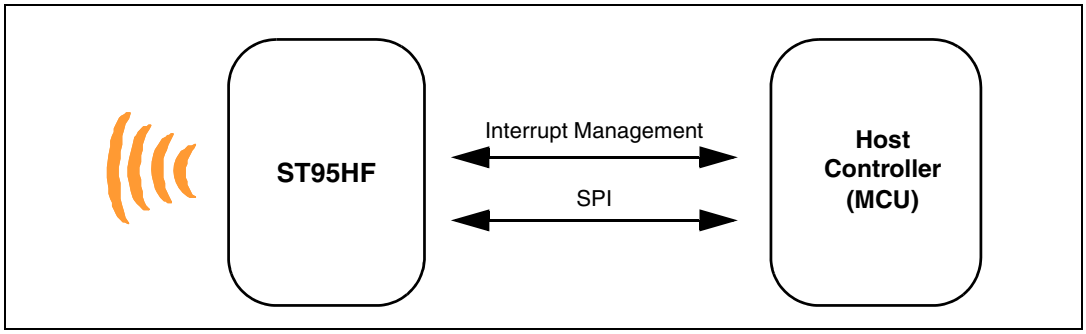
The ST95HF manages frame coding and decoding in Reader and Card Emulation modes for standard applications such as Near Field Communication (NFC), proximity and vicinity standards.

The ST95HF embeds an Analog Front End to provide the 13.56 MHz Air Interface.

The ST95HF supports ISO/IEC 14443 Type A and B communication in Reader and Card Emulation modes, ISO/IEC 15693 (single or double subcarrier in Reader mode only) and ISO/IEC 18092 communication protocols in Reader and Card Emulation modes.

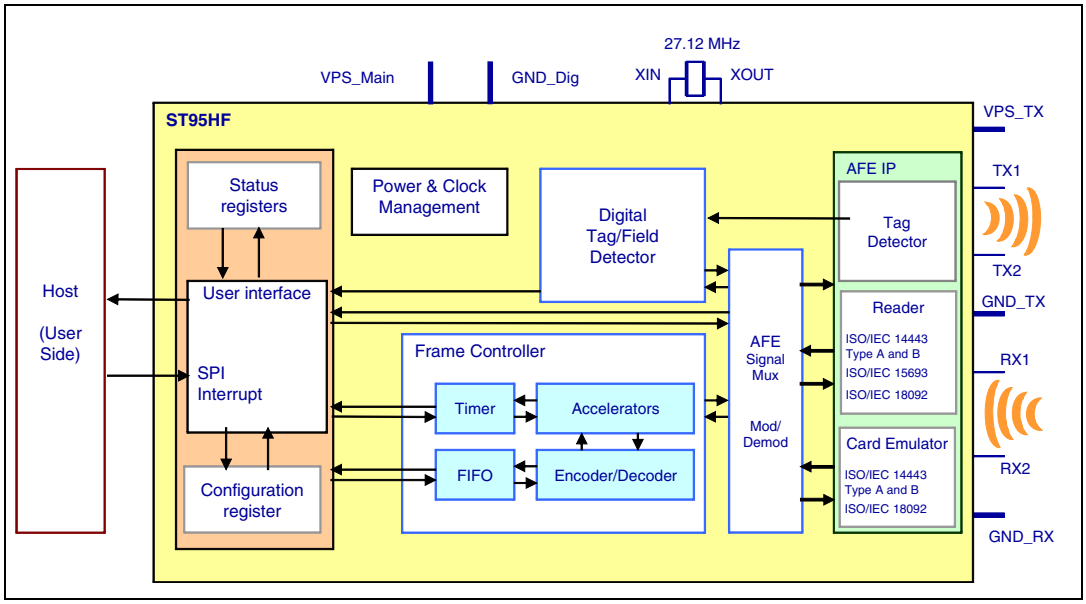
The ST95HF also supports the detection, reading and writing of NFC Forum Type 1, 2, 3 and 4 tags.

**Figure 1. ST95HF application overview**




## 1.1 Block diagram

**Figure 2. ST95HF block diagram**



## 1.2 List of terms

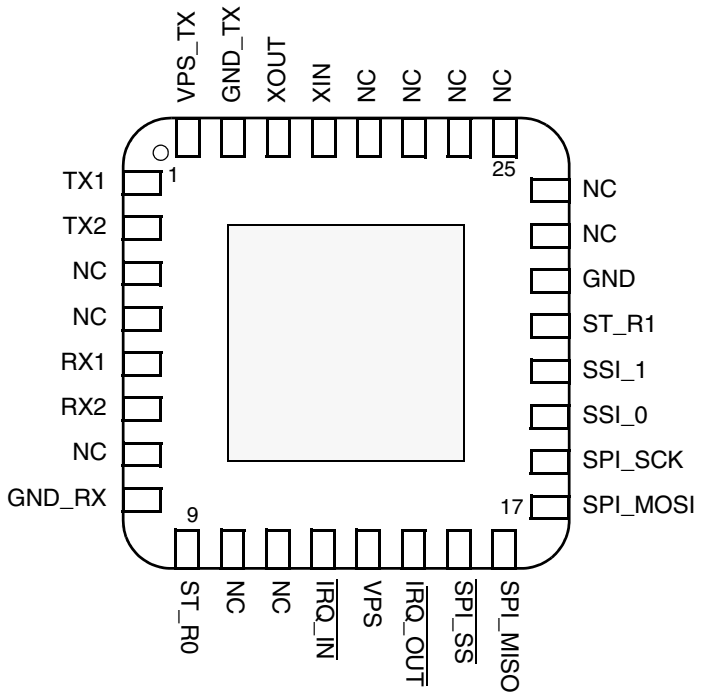
**Table 1. List of terms**

| Term | Meaning |
|---|---|
| DAC | Digital analog converter |
| GND | Ground |
| HFO | High frequency oscillator |
| LFO | Low frequency oscillator |
| MCU | Microcontroller unit |
| MIFARE(1) | Communication protocol |
| NFC | Near Field Communication |
| RFID | Radio Frequency Identification |
| RFU | Reserved for future use |
| SPI | Serial peripheral interface |
| $t_L$ | Low frequency period |
| $t_{REF}$ | Reference time |
| WFE | Wait For Event |

1. MIFARE® and MIFARE® Classic are registered trademarks of NXP BV, and are used under license.

# 2 Pin and signal descriptions

**Figure 3. ST95HF pinout description**




Shaded area represents the dissipation pad.
(Must be connected to ground.)

**Table 2. ST95HF pin descriptions**

| Pin | Pin name | Type(1) | Main function | Alternate function |
|---|---|---|---|---|
| 1 | TX1 | O | Driver output 1 | - |
| 2 | TX2 | O | Driver output 2 | - |
| 3 | NC | - | Not connected | - |
| 4 | NC | - | Not connected | - |
| 5 | RX1 | I | Receiver input 1 | - |
| 6 | RX2 | I | Receiver input 2 | - |
| 7 | NC | - | Not connected | - |
| 8 | GND_RX | P | Ground (analog) | - |
| 9 | ST_R0 | O | ST Reserved(2) | - |
| 10 | NC | - | Not connected | - |
| 11 | NC | - | Not connected | - |
| 12 | $\overline{\text{IRQ\_IN}}$ | I (3) | Interrupt input | - |
| 13 | VPS | P | Main power supply | - |

**Table 2. ST95HF pin descriptions (continued)**

| Pin | Pin name | Type(1) | Main function | Alternate function |
|---|---|---|---|---|
| 14 | $\overline{\text{IRQ\_OUT}}$ | O (4) | Interrupt output | - |
| 15 | $\overline{\text{SPI\_SS}}$ | I (5) | SPI Slave Select (active low) | - |
| 16 | SPI_MISO | O (5) | SPI Data, Slave Output | - |
| 17 | SPI_MOSI | I (5) | SPI Data, Slave Input (5) | - |
| 18 | SPI_SCK | I (6) | SPI serial clock | - |
| 19 | SSI_0 | I (5) | Select serial communication interface | - |
| 20 | SSI_1 | I (5) | Select serial communication interface | - |
| 21 | ST_R1 | I (7) | ST Reserved | - |
| 22 | GND | P | Ground (digital) | - |
| 23 | NC | - | Not connected | - |
| 24 | NC | - | Not connected | - |
| 25 | NC | - | Not connected | - |
| 26 | NC | - | Not connected | - |
| 27 | NC | - | Not connected | - |
| 28 | NC | - | Not connected | - |
| 29 | XIN | - | Crystal oscillator input | - |
| 30 | XOUT | - | Crystal oscillator output | - |
| 31 | GND_TX | P | Ground (RF drivers) | - |
| 32 | VPS_TX | P | Power supply (RF drivers) | - |

1. I: Input, O: Output, and P: Power
2. Must add a capacitor to ground (~1 nF).
3. Pad internally connected to a Very Weak Pull-up to VPS.
4. Pad internally connected to a Weak Pull-up to VPS.
5. Must not be left floating.
6. Pad internally connected to a Weak Pull-down to GND.
7. Pad input in High Impedance. Must be connected to VPS.

# 3 Power management and operating modes

## 3.1 Operating modes

The ST95HF has 2 operating modes: Wait for Event (WFE) and Active. In Active mode, the ST95HF communicates actively with a tag or an external host (an MCU, for example). WFE mode includes four low consumption states: Power-up, Hibernate, Sleep/Field Detector and Tag Detector.

The ST95HF can switch from one mode to another.

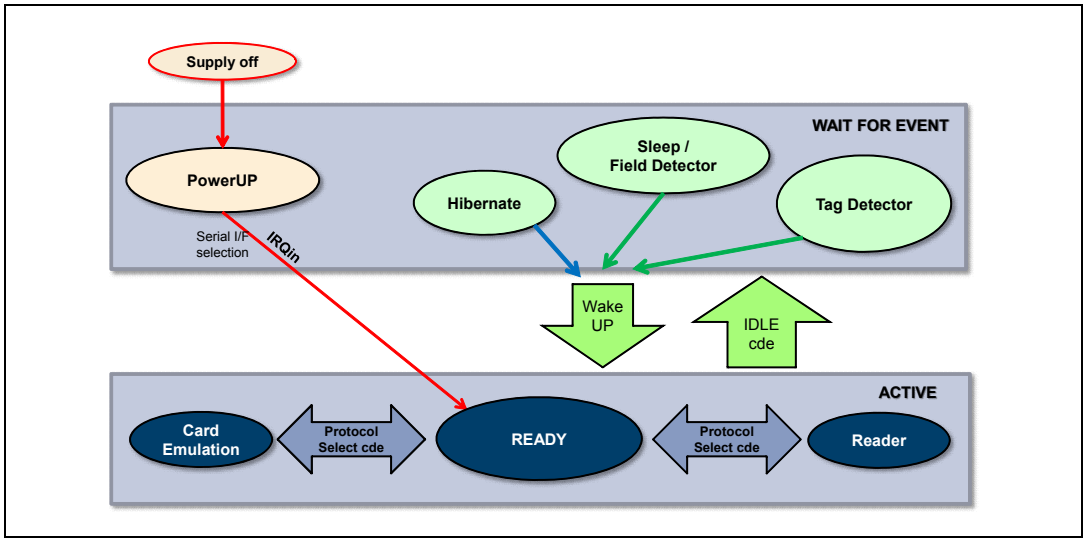
**Table 3. ST95HF operating modes and states**

| Mode | State | Description |
|---|---|---|
| **Wait For Event (WFE)** | Power-up | This mode is accessible directly after POR.<br>Low level on IRQ_IN pin (longer than 10 µs) is the only wakeup source. LFO (low-frequency oscillator) is running in this state. |
| | Hibernate | Lowest power consumption state. The ST95HF has to be woken-up in order to communicate. Low level on IRQ_IN pin (longer than 10 µs) is the only wakeup source. |
| | Sleep/Field Detector | Low power consumption state. Wakeup source is configurable:<br>– Timer<br>– IRQ_IN pin<br>– SPI_SS pin<br>– Field Detector<br>LFO (low-frequency oscillator) is running in this state. |
| | Tag Detector | Low power consumption state with tag detection. Wakeup source is configurable:<br>– Timer<br>– IRQ_IN pin<br>– SPI_SS pin<br>– Tag detector<br>LFO (low-frequency oscillator) is running in this state. |
| **Active** | Ready | In this mode, the RF is OFF and the ST95HF waits for a command (PROTOCOLSELECT, ...) from the external host via the selected serial interface (SPI). |
| | Reader | The ST95HF can communicate with a tag using the selected protocol or with an external host using the SPI interface. |
| | Card Emulation | The ST95HF can communicate as a Card or Tag with an external reader. The Card or Tag application is located in the Host and communicates with the ST95HF via the SPI interface. |

Hibernate, Tag Detector, and Sleep/Field Detector states can only be activated by a command from the external host. As soon as any of these three states are activated, the ST95HF can no longer communicate with the external host. It can only be woken up.

The behavior of the ST95HF in 'Tag Detector' state is defined by the Idle command.

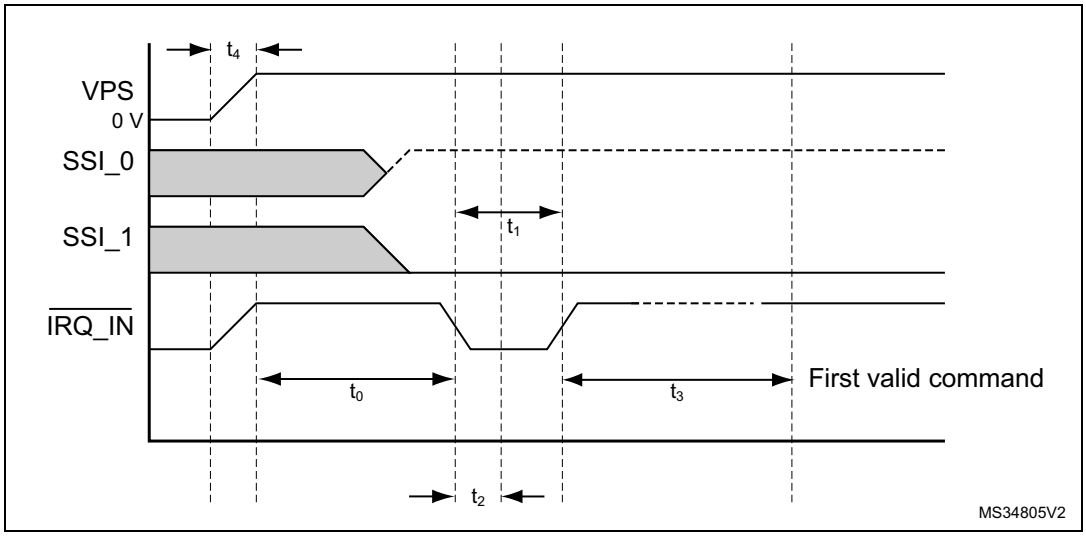
**Figure 4. ST95HF initialization and operating state change**




## 3.2 Startup sequence

After the power supply is established at power-on, the ST95HF waits for a low pulse on the pin $\overline{IRQ\_IN}$ ($t_1$) before automatically selecting the external interface (SPI) and entering Ready state after a delay ($t_3$).

**Figure 5. Power-up sequence**




1. Note for pin SSI0: - - - SPI selected
2. Pin $\overline{IRQ\_IN}$ low level < 0.2 VPS_Main.

*Note: When ST95HF leaves WFE mode (from Power-up, Hibernate, Tag Detector, or Sleep/Field Detector) following an $\overline{IRQ\_IN}$ low level pulse.*

*Figure 5* shows the power-up sequence for a ST95HF device; where,

- $t_0$ is the initial wake-up delay 100 µs (minimum)
- $t_1$ is the minimum interrupt width 10 µs (minimum)
- $t_2$ is the delay for the serial interface selection 250 ns (typical)
- $t_3$ is the HFO setup time ($t_{SU(HFO)}$) 10 ms (maximum)
- $t_4$ is the $V_{PS}$ ramp-up time from 0V to $V_{PS}$ 10 ms (max. by design validation)

*Note: VPS must be 0V before executing the start-up sequence.*

*The serial interface is selected after the following falling edge of pin $\overline{IRQ\_IN}$ when leaving from POR or Hibernate state.*

*Table 4* lists the signal configuration used to select the serial communication interface.

**Table 4. Select serial communication interface selection table**

| Pin | Serial interface |
|---|---|
| SSI_0 | SPI: 1 |
| SSI_1 | SPI: 0 |

# 4 Communication protocols

## 4.1 Serial peripheral interface (SPI)

### 4.1.1 Polling mode

In order to send commands and receive replies, the application software has to perform 3 steps.

1. Send the command to the ST95HF.
2. Poll the ST95HF until it is ready to transmit the response.
3. Read the response.

The application software should never read data from the ST95HF without being sure that the ST95HF is ready to send the response.

The maximum allowed SPI communication speed is $f_{SCK}$.

A Control byte is used to specify a communication type and direction:

- 0x00: Send command to the ST95HF
- 0x03: Poll the ST95HF
- 0x02: Read data from the ST95HF
- 0x01: Reset the ST95HF

The $\overline{\text{SPI\_SS}}$ line is used to select a device on the common SPI bus. The $\overline{\text{SPI\_SS}}$ pin is active low.

When the $\overline{\text{SPI\_SS}}$ line is inactive, all data sent by the Master device is ignored and the MISO line remains in High Impedance state.

In Slave mode, the phase and polarization are defined with CPOL = 1 and CPHA = 1 or CPOL = 0 and CPHA = 0.

**Figure 6. Sending command to ST95HF**

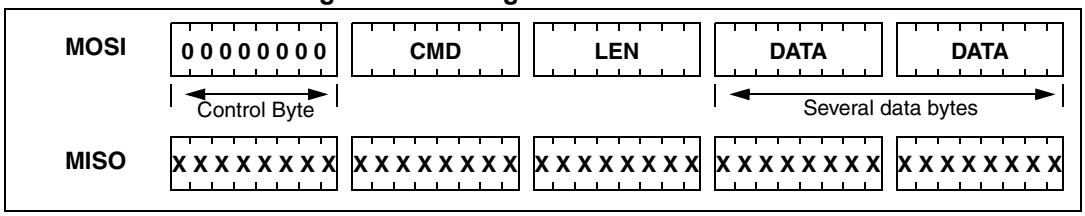



**Figure 7. Polling the ST95HF until it is ready**

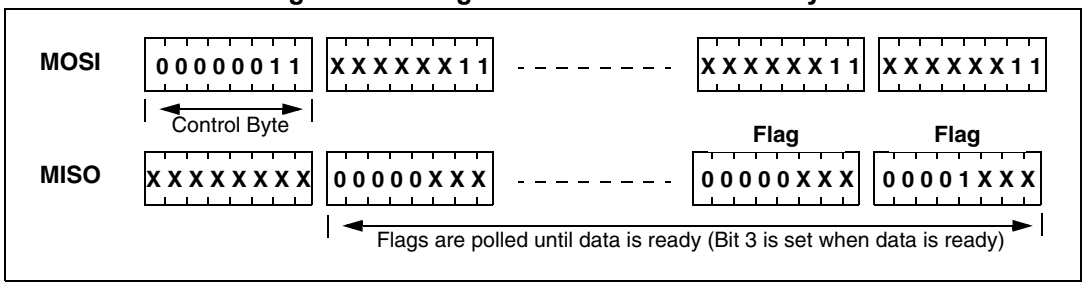
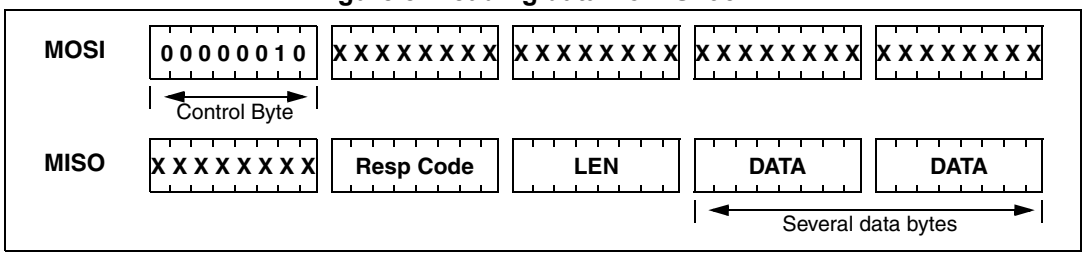


**Table 5. Interpretation of flags**

| Bit | Meaning (application point of view) |
|---|---|
| [7:4] | Not significant |
| 3 | Data can be read from the ST95HF when set. |
| 2 | Data can be sent to the ST95HF when set. |
| [1:0] | Not significant |

**Figure 8. Reading data from ST95HF**



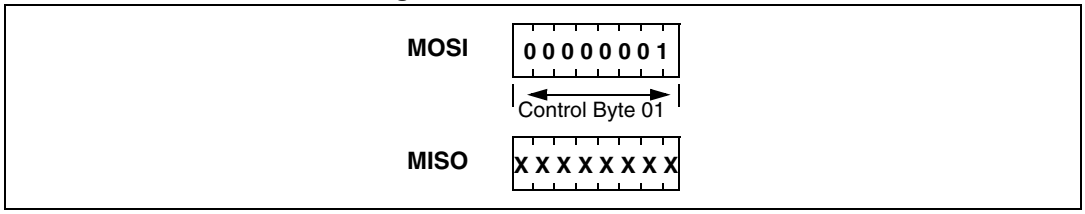

Data must be sampled at the rising edge of the SCK signal.

‘Sending’, ‘Polling’ and ‘Reading’ commands must be separated by a high level of the SPI_SS line. For example, when the application needs to wait for data from the ST95HF, it asserts the SPI_SS line low and issues a ‘Polling’ command. Keeping the SPI_SS line low, the Host can read the Flags Waiting bit which indicates that the ST95HF can be read. Then, the application has to assert the SPI_SS line high to finish the polling command. The Host asserts the SPI_SS line low and issues a ‘Reading’ command to read data. When all data is read, the application asserts the SPI_SS line high.

The application is not obliged to keep reading Flags using the Polling command until the ST95HF is ready in one command. It can issue as many 'Polling' commands as necessary. For example, the application asserts SPI_SS low, issues 'Polling' commands and reads Flags. If the ST95HF is not ready, the application can assert SPI_SS high and continue its algorithm (measuring temperature, communication with something else). Then, the application can assert SPI_SS low again and again issue 'Polling' commands, and so on, as many times as necessary, until the ST95HF is ready.

Note that at the beginning of communication, the application does not need to check flags to start transmission. The ST95HF is assumed to be ready to receive a command from the application.

**Figure 9. Reset the ST95HF**




To reset the ST95HF using the SPI, the application sends the SPI Reset command (Control Byte 01, see *Figure 9*) which starts the internal controller reset process and puts the ST95HF into Power-up state. The ST95HF will wake up when pin IRQ_IN goes low. The ST95HF reset process only starts when the SPI_SS pin returns to high level.

**Caution: SPI communication is MSB first.**

### 4.1.2 Interrupt mode

When the ST95HF is configure to use the SPI serial interface, pin $\overline{\text{IRQ\_OUT}}$ is used to give additional information to user. When the ST95HF is ready to send back a reply, it sends an Interrupt Request by setting a low level on pin $\overline{\text{IRQ\_OUT}}$, which remains low until the host reads the data.

The application can use the Interrupt mode to skip the polling stage.

**Caution: SPI communication is MSB first.**

## 4.2 Error codes

**Table 6. Possible error codes and their meaning**

| Code | Name | Meaning |
|---|---|---|
| 0x80 | EFrameRecvOK | Frame correctly received (additionally see CRC/Parity information) |
| 0x85 | EUserStop | Stopped by user (used only in Card mode) |
| 0x86 | ECommError | Hardware communication error |
| 0x87 | EFrameWaitTOut | Frame wait time out (no valid reception) |
| 0x88 | EInvalidSof | Invalid SOF |
| 0x89 | EBufOverflow | Too many bytes received and data still arriving |
| 0x8A | EFramingError | if start bit = 1 or stop bit = 0 |
| 0x8B | EEgtError | EGT time out |
| 0x8C | EInvalidLen | Valid for ISO/IEC 18092, if Length <3 |
| 0x8D | ECrcError | CRC error, Valid only for ISO/IEC 18092 |
| 0x8E | ERecvLost | When reception is lost without EOF received (or subcarrier was lost) |
| 0x8F | ENoField | When Listen command detects the absence of external field |
| 0x90 | EUnintByte | Residual bits in last byte. Useful for ACK/NAK reception of ISO/IEC 14443 Type A. |

## 4.3 Support of long frames

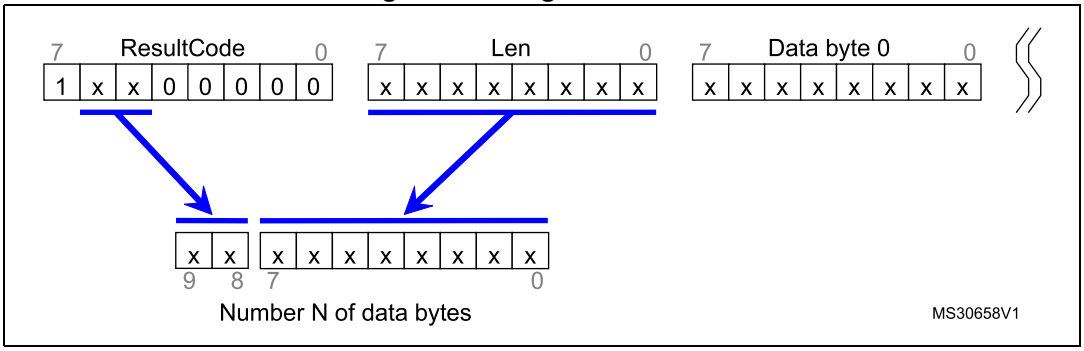
In Reader mode it is possible to receive up to 528 bytes of frame data from VICC and Type-B cards and up to 256 bytes of frame data from Type-A cards. In this case, the device sends a reply to the external MCU in the following format:

```
<ResultCode> + <Len> + <N bytes of data>
```

**Table 7. Format of ResultCode**

| Bit | Meaning | |
|---|---|---|
| 7 | Always 1 | |
| 6 | Bit 9 of Length | See examples and explanation below |
| 5 | Bit 8 of Length | |
| 4 | If set, there are residual bits in the last byte. Applicable only for Type-A protocol. | |
| 3:0 | Always 0 | |

**Figure 10. Long frame format**




The number of databytes is 10-bit long.

**Table 8. Examples of ResultCode: Len pairs**

| ResultCode | Len | Length of data |
|---|---|---|
| 0x80 | 0x00 | 0 |
| 0x80 | 0x01 | 1 |
| 0x80 | 0xFF | 255 |
| 0xA0 | 0x00 | 256 |
| 0xA0 | 0x01 | 257 |
| 0xA0 | 0xFF | 511 |
| 0xC0 | 0x00 | 512 |
| 0xC0 | 0x01 | 513 |

# 5 Commands

## 5.1 Command format

- The frame from the Host to the ST95HF has the following format:
  `<CMD><Len><Data>`
- The frame from the ST95HF to Host has the following format:
  `<RespCode><Len><Data>`

These two formats are available in SPI mode.

Fields <Cmd>, <RespCode> and <Len> are always 1 byte long. <Data> can be from 0 to 253 bytes.

*Note: The ECHO command is an exception as it has only one byte (0x55).*

The following symbols correspond to:

>>> Frame sent by the Host to ST95HF

<<< Frame sent by the ST95HF to the Host

## 5.2 List of commands

*Table 9* summarizes the available commands.

**Table 9. List of ST95HF commands**

| Code | Command | Description |
|---|---|---|
| 0x01 | IDN | Requests short information about the ST95HF and its revision. |
| 0x02 | PROTOCOLSELECT | Selects the RF communication protocol and specifies certain protocol-related parameters. |
| 0x03 | POLLFIELD | Returns the current value of the FieldDet flag (used in Card Emulation mode). |
| 0x04 | SendRecv | Sends data using the previously selected protocol and receives the tag response. |
| 0x05 | LISTEN | Listens for data using previously selected protocol (used in Card Emulation mode). |
| 0x06 | SEND | Sends data using previously selected protocol (used in Card Emulation mode). |
| 0x07 | IDLE | Switches the ST95HF into a low consumption Wait for Event (WFE) mode (Power-up, Hibernate, Sleep/Field or Tag Detection), specifies the authorized wake-up sources and waits for an event to exit to Ready state. |
| 0x08 | RDREG | Reads Wake-up event register or the Analog Register Configuration (ACC_A or ARC_B) register. |

Table 9. List of ST95HF commands (continued)

| Code | Command | Description |
|---|---|---|
| 0x09 | WRREG | Writes Analog Register Configuration (ACC_A or ARC_B)) register or writes index of ACC_A or ARC_B register address.<br>Writes the Timer Window (TimerW) value dedicated to ISO/IEC 14443 Type A tags.<br>Writes the AutoDetect Filter enable register dedicated to ISO/IEC 18092 tags. |
| 0x0B | SubFreqRes | Returns the most recent subcarrier frequency detected for ISO/IEC 18092 communication. |
| 0x0D | ACFILTER | Enables or disables the anti-collision filter for ISO/IEC 14443 Type A protocol. |
| 0x55 | ECHO | ST95HF performs a serial interface ECHO command (reply data 0x55 or stops the Listening state when a listen command has been sent without error). |
| Other codes | | ST Reserved |

## 5.3 IDN command (0x01) description

The IDN command (0x01) gives brief information about the ST95HF and its revision.

Table 10. IDN command description

| Direction | Data | Comments | Example |
|---|---|---|---|
| Host to ST95HF | 0x01 | Command code | >>>0x0100 |
| | 0x00 | Length of data | |
| ST95HF to Host | 0x00 | Result code | <<<0x000F4E4643204653324A41535434002ACE<br><br>In this example,<br><<<0x4E4643204653324A4153543400 : 'NFC FS2JAST4', #4 (Last Character of NFC FS2JAST4 means ROM code revision 4.)<br>0x2ACE: CRC of ROM (real CRC may differ from this example) |
| | <Len> | Length of data | |
| | <Device ID> | Data in ASCII format (13 bytes) | |
| | <ROM CRC> | CRC calculated for ROM content (2 bytes) | |

It takes approximately 6 ms to calculate the CRC for the entire ROM. The application must allow sufficient time for waiting for a response for this command.

## 5.4 Protocol Select command (0x02) description

This command selects the RF communication protocol and prepares the ST95HF for communication with a reader or contactless tag.

**Table 11. ProtocolSelect command description**

| Direction | Data | Comments | Example |
|---|---|---|---|
| Host to ST95HF | 0x02 | Command code | See *Table 12: List of <Parameters> values for the ProtocolSelect command for different protocols (Reader) on page 22*.<br><br>See *Table 13: List of <Parameters> values for different protocols (Card Emulation) on page 25*. |
| | <Len> | Length of data | |
| | <Protocol> | Protocol codes (Reader):<br>00: Field OFF<br>01: ISO/IEC 15693<br>02: ISO/IEC 14443-A / NFC Forum Tag Type 1, Type 2, Type 4A<br>03: ISO/IEC 14443-B / NFC Forum Tag Type 4B<br>04: ISO/IEC 18092 / NFC Forum Tag Type 3<br>---<br>Protocol codes (Card Emulation):<br>12: ISO/IEC 14443-A<br>13: ST Reserved (ISO/IEC 14443-B under qualification)<br>14: ST Reserved (ISO/IEC 18092 under qualification) | |
| | <Parameters> | Each protocol has a different set of parameters. See *Table 12*. | |
| ST95HF to Host | 0x00 | Result code | <<<0x0000<br>Protocol is successfully selected |
| | 0x00 | Length of data | |
| ST95HF to Host | 0x82 | Error code | <<<0x8200<br>Invalid command length |
| | 0x00 | Length of data | |
| ST95HF to Host | 0x83 | Error code | <<<0x8300<br>Invalid protocol |
| | 0x00 | Length of data | |

Note that there is no ‘Field ON’ command. When the application selects an RF communication protocol, the field automatically switches ON if the Reader state is selected.

When the application selects a protocol, the ST95HF performs all necessary settings: it will choose the appropriate reception and transmission chains, switch ON or OFF the RF field and connect the antenna accordingly.

Different protocols have different sets of parameters. Values for the <Parameters> field are listed in *Table 12*.

**Table 12. List of <Parameters> values for the PROTOCOLSELECT command for different protocols (Reader)**

| Protocol | Code | Parameters | | | Examples of commands |
|---|---|---|---|---|---|
| | | Byte | Bit | Function | |
| Field OFF | 0x00 | 0 | 7:0 | RFU | >>>0x02020000 |
| ISO/IEC 15693 | 0x01 | 0 | 7:6 | RFU | H 100 S: >>>0x02 02 01 01<br>H 100 D: >>>0x02 02 01 03<br>H 10 S: >>>0x02 02 01 05<br>H 10 D: >>>0x02 02 01 07<br>L 100 S: >>>0x02 02 01 21<br>L 100 D: >>>0x02 02 01 23<br>L 10 S: >>>0x02 02 01 25<br>L 10 D: >>>0x02 02 01 27<br><br>In these examples, the CRC is automatically appended. |
| | | | 5:4 | 00: 26 Kbps (H)<br>01: 52 Kbps<br>10: 6 Kbps (L)<br>11: RFU | |
| | | | 3 | 0: Respect 312-µs delay<br>1: Wait for SOF (1) | |
| | | | 2 | 0: 100% modulation (100)<br>1: 10% modulation (10) | |
| | | | 1 | 0: Single subcarrier (S)<br>1: Dual subcarrier (D) | |
| | | | 0 | Append CRC if set to '1'. (1) | |
| ISO/IEC 14443 Type A<br><br>NFC Forum Tag Type 1 (Topaz)<br><br>NFC Forum Tag Type 2<br><br>NFC Forum Tag Type 4A | 0x02 | 0 | 7:6 | Transmission data rate<br>00: 106 Kbps<br>01: 212 Kbps (2)<br>10: 424 Kbps<br>11: RFU | >>>0x02020200: ISO/IEC 14443 Type A tag, 106 Kbps transmission and reception rates, Time interval 86/90<br><br>Note that REQA, WUPA, Select20 and Select70 commands use a fixed interval of 86/90 µs between a request and its reply. Other commands use a variable interval with fixed granularity.<br>Refer to the ISO/IEC 14443 standard for more details. |
| | | | 5:4 | Reception data rate<br>00: 106 Kbps<br>01: 212 Kbps (2)<br>10: 424 Kbps<br>11: RFU | |
| | | | 3 | RFU | |
| | | | 2:0 | RFU | |
| | | 1 | 7:0 | PP | These 5 bytes are optional. The default PP:MM:DD value is 0 (corresponds to FDT 86/90µs) . For other values, FDT = (2^PP)*(MM+1)*(DD+128) *32/13.56 µs |
| | | 2 | 7:0 | MM | |
| | | 3 | 7:0 | DD (optional to PP:MM) | |
| | | 4 | 7:0 | ST Reserved (Optional) | - |
| | | 5 | 7:0 | ST Reserved (Optional) | - |

**Table 12. List of <Parameters> values for the PROTOCOLSELECT command for different protocols (Reader) (continued)**

| Protocol | Code | Parameters | | | Examples of commands |
|---|---|---|---|---|---|
| | | Byte | Bit | Function | |
| ISO/IEC 14443 Type B<br><br>NFC Forum Tag Type 4B | 0x03 | 0 | 7:6 | Transmission data rate<br>00: 106 Kbps<br>01: 212 Kbps<br>10: 424 Kbps<br>11: 848 Kbps | >>>0x02020301:<br>ISO/IEC 14443 Type B tag with CRC appended |
| | | | 5:4 | Reception data rate<br>00: 106 Kbps<br>01: 212 Kbps<br>10: 424 Kbps<br>11: 848 Kbps | |
| | | | 3:1 | RFU | |
| | | | 0 | Append CRC if set to '1'. (1) | |
| | | 1 | 7:0 | PP | These 9 bytes are optional. Default value of PP:MM:DD is 0 and corresponds to FWT ~302µs.<br>FWT = (2^PP)*(MM+1)*(DD+128)* 32/13.56 µs |
| | | 2 | 7:0 | MM | |
| | | 3 | 7:0 | DD (optional to PP:MM) | |
| | | 5:4 | 7:0 | TTTT (Optional) | TR0 = TTTT/FC (LSB first), default 1023 = 0x3FF |
| | | 6 | 7:0 | YY (Optional) | PCD Min TR1 (Min_TR1 = 8 * XX / $f_S$), default = 0 |
| | | 7 | 7:0 | ZZ (Optional) | PCD Max TR1 (Max_TR1 = 8 * ZZ / $f_S$), default = 26 = 0x1A |
| | | 8 | 7:0 | ST Reserved (Optional) | - |
| | | 9 | 7:0 | ST Reserved (Optional) | - |

**Table 12. List of <Parameters> values for the PROTOCOLSELECT command for different protocols (Reader) (continued)**

| Protocol | Code | Parameters | | | Examples of commands |
|---|---|---|---|---|---|
| | | Byte | Bit | Function | |
| ISO/IEC 18092<br><br>NFC Forum Tag Type 3 | 0x04 | 0 | 7:6 | Transmission data rate<br>00: RFU<br>01: 212 Kbps<br>10: 424 Kbps<br>11: RFU | >>>0x02020451:<br>ISO/IEC18092 tag, 212 Kbps transmission and reception rates with CRC appended.<br><br>Parameter 'Slot counter' is not mandatory. If it is not present, it is assumed that SlotCounter = 0x00 (1 slot)<br><br>For device detection commands, byte 1 bit 4 must be set to '0'. In this case, the FWT is 2.4 ms for the 1st slot and 1.2 ms more for each following slot, if slot counter is specified.<br><br>If slot counter = 0x10, the ST95HF does not respect reply timings, but polls incoming data and searches a valid response during ~8.4 ms. |
| | | | 5:4 | Reception data rate<br>00: RFU<br>01: 212 Kbps<br>10: 424 Kbps<br>11: RFU | |
| | | | 3:1 | RFU | |
| | | | 0 | Append CRC if set to '1'. (1) | |
| | | 1 | 7:5 | RFU | |
| | | | 4 | 0: FWT = 2.4 ms<br>1: FWT is specified by PP:MM bits | |
| | | | 3:0 | Slot counter<br>0: 1 slot<br>1: 2 slots<br>…<br>F: 16 slots | |
| | | 2 | 7:0 | PP | These 3 bytes are optional. Default value PP:MM:DD: is 0 and corresponds to RWT ~302µs.<br>RWT = (2^PP)*(MM+1)*(DD+128)*32/13.56µs |
| | | 3 | 7:0 | MM | |
| | | 4 | 7:0 | DD (optional to PP:MM) | |

1. It is recommended to set this bit to '1'.
2. Not characterized.

**Table 13. List of <Parameters> values for different protocols (Card Emulation)**

| Protocol (Card) | Code | Parameters | | | Examples of commands Comments |
|---|---|---|---|---|---|
| | | Byte | Bit | Function | |
| ISO/IEC 14443 Type A (1) | 0x12 | 0 | 7:6 | Transmission data rate<br>00: 106 Kbps<br>01: 212 Kbps (2)<br>10: 424 Kbps (2)<br>11: RFU | >>>0x02021208<br><<<0x0000<br>Card Emulation for ISO/IEC 14443 Type A, Data rate is 106 Kbps for both up- and down-links. |
| | | | 5:4 | Reception data rate<br>00: 106 Kbps<br>01: 212 Kbps (2)<br>10: 424 Kbps (2)<br>11: RFU | |
| | | | 3 | 0: Return an error, if no RF field<br>1: Wait for RF field | |
| | | | 2 | RFU | |
| | | | 1 | 0: HFO<br>1: ClkRec | |
| | | | 0 | RFU | |

1. Topaz is not supported in Card Emulation mode.
2. Not qualified for this version.