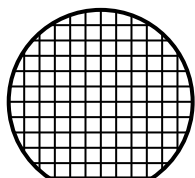


## 13.56 MHz short-range contactless memory chip with 2048-bit EEPROM and anticollision functions



-Unsawn wafer  
-Bumped and sawn wafer

### Product status link

[ST25TB02K](#)

### Features

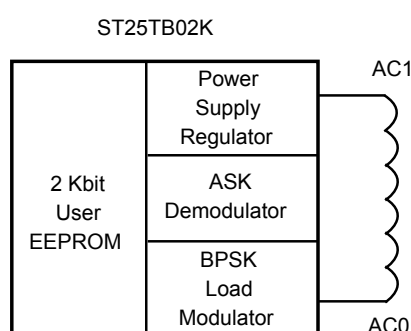
- ISO 14443-2 Type B air interface compliant
- ISO 14443-3 Type B frame format compliant
- 13.56 MHz carrier frequency
- 847 kHz subcarrier frequency
- 106 Kbit/second data transfer
- 8 bit Chip\_ID based anticollision system
- 2 count-down binary counters with automated anti-tearing protection
- 64-bit Unique Identifier
- 2048-bit EEPROM with write protect feature
- Read\_block and Write\_block (32 bits)
- Internal tuning capacitor: 68 pF
- 1 million erase/write cycles
- 40-year data retention
- Self-timed programming cycle
- 5 ms typical programming time

## 1 Description

The **ST25TB02K** is a contactless memory, powered by an externally transmitted radio wave. It contains a 2048-bit user EEPROM. The memory is organized as 64 blocks of 32 bits. The **ST25TB02K** is accessed via the 13.56 MHz carrier. Incoming data are demodulated and decoded from the received amplitude shift keying (ASK) modulation signal and outgoing data are generated by load variation using bit phase shift keying (BPSK) coding of a 847 kHz sub-carrier. The received ASK wave is 10% modulated. The data transfer rate between the **ST25TB02K** and the reader is 106 kbit/s in both reception and emission modes.

The **ST25TB02K** follows the ISO 14443 - 2 Type B recommendation for the radio-frequency power and signal interface.

**Figure 1. Logic diagram**



The **ST25TB02K** is specifically designed for short range applications that need re-usable products. The **ST25TB02K** includes an anticollision mechanism that allows it to detect and select tags present at the same time within range of the reader. The anticollision is based on a probabilistic scanning method using slot markers.

**Table 1. Signal names**

| Signal names | Description  |
|--------------|--------------|
| AC1          | Antenna coil |
| AC0          | Antenna coil |

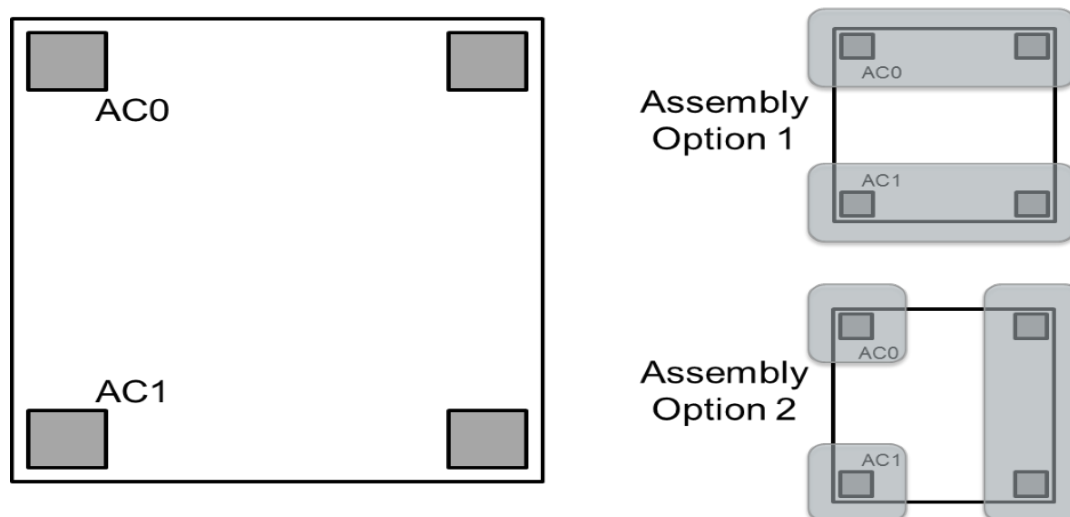
The **ST25TB02K** contact-less EEPROM can be randomly read and written in block mode (each block containing 32 bits). The instruction set includes the following nine commands:

- Read\_block
- Write\_block
- Initiate
- Pcall16
- Slot\_marker
- Select
- Completion
- Reset\_to\_inventory
- Get\_UID

The **ST25TB02K** memory is organized in three areas, as described in [Section 4 Memory mapping](#). The first area is a resettable OTP (one time programmable) area in which bits can only be switched from 1 to 0. Using a special command, it is possible to erase all bits of this area to 1. The second area provides two 32-bit binary counters which can only be decremented. The last area is the EEPROM memory. It is accessible by block of 32 bits and includes an auto-erase cycle during each Write\_block command.

Die floor plan and physical options related to the die assembly are described in [Figure 2](#).

**Figure 2. Die floor plan and assembly options**



For the option 1 of the die assembly, the CTUN (referenced in Table 2) can increase from 0.5pF to 1pF. The option 2 of the die assembly is showing a tripod which can be used for physical stability, having no impact on CTUN parameter.

---

## 2 Signal description

---

### 2.1 AC1, AC0

The pads for the Antenna Coil. AC1 and AC0 must be directly bonded to the antenna.

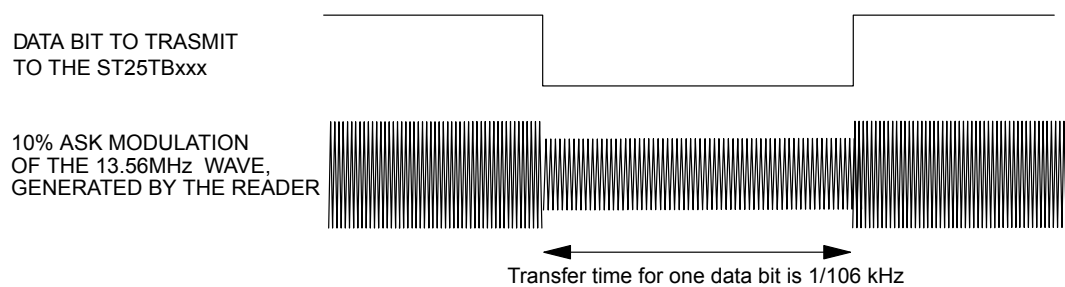
## 3 Data transfer

### 3.1 Input data transfer from reader to ST25TB02K (request frame)

The reader must generate a 13.56 MHz sinusoidal carrier frequency at its antenna, with enough energy to “remote-power” the memory. The energy received at the **ST25TB02K**’s antenna is transformed into a supply voltage by a regulator, and into data bits by the ASK demodulator. For the **ST25TB02K** to decode correctly the information it receives, the reader must 10% amplitude-modulate the 13.56 MHz wave before sending it to the **ST25TB02K**. This is represented in **Figure 3**. The data transfer rate is 106 Kbits/s.

In some figures of this datasheet the ST25TBxxx refers to **ST25TB02K**.

**Figure 3. ST25TB02K 10% ASK modulation of the received wave**

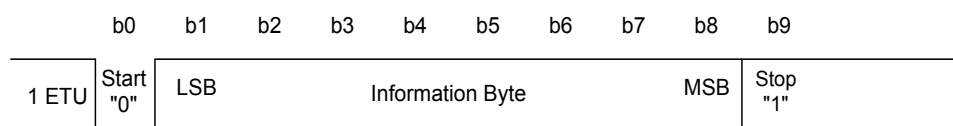


#### 3.1.1 Character transmission format for request frame

The **ST25TB02K** transmits and receives data bytes as 10-bit characters, with the least significant bit ( $b_0$ ) transmitted first, as shown in **Figure 4**. Each bit duration, an ETU (elementary time unit), is equal to 9.44  $\mu$ s (1/106 kHz).

These characters, framed by a start of frame (SOF) and an end of frame (EOF), are put together to form a command frame as shown in **Figure 10**. A frame includes an SOF, commands, addresses, data, a CRC and an EOF as defined in the ISO 14443-3 Type B Standard. If an error is detected during data transfer, the **ST25TB02K** does not execute the command, but it does not generate an error frame.

**Figure 4. ST25TB02K request frame character format**



**Table 2. Bit description**

| Bit            | Description                                      | Value                                                             |
|----------------|--------------------------------------------------|-------------------------------------------------------------------|
| $b_0$          | Start bit used to synchronize the transmission   | $b_0 = 0$                                                         |
| $b_1$ to $b_8$ | Information byte (command, address or data)      | The information byte is sent with the least significant bit first |
| $b_9$          | Stop bit used to indicate the end of a character | $b_9 = 1$                                                         |

#### 3.1.2 Request start of frame

The SOF described in **Figure 5** is composed of:

- one falling edge,

- followed by 10 ETUs at logic-0,
- followed by a single rising edge,
- followed by at least 2 ETUs (and at most 3) at logic-1.

**Figure 5. Request start of frame**

|     | b0 | b1 | b2 | b3 | b4 | b5 | b6 | b7 | b8 | b9 | b10 | b11 |
|-----|----|----|----|----|----|----|----|----|----|----|-----|-----|
| ETU | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1   | 1   |

### 3.1.3

#### Request end of frame

The EOF shown in Figure 6 is composed of:

- one falling edge,
- followed by 10 ETUs at logic-0,
- followed by a single rising edge.

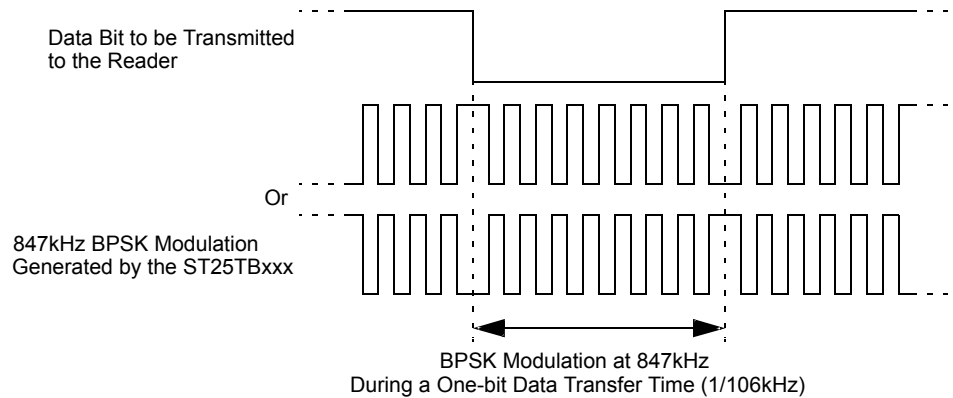
**Figure 6. Request end of frame**

|     | b0 | b1 | b2 | b3 | b4 | b5 | b6 | b7 | b8 | b9 |
|-----|----|----|----|----|----|----|----|----|----|----|
| ETU | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

## 3.2 Output data transfer from to reader ST25TB02K (answer frame)

The data bits issued by the ST25TB02K use back-scattering. Back-scattering is obtained by modifying the ST25TB02K current consumption at the antenna (load modulation). The load modulation causes a variation at the reader antenna by inductive coupling. With appropriate detector circuitry, the reader is able to pick up information from the ST25TB02K. To improve load-modulation detection, data is transmitted using a BPSK encoded, 847 kHz subcarrier frequency  $f_s$  as shown in Figure 7, and as specified in the ISO 14443-2 Type B standard.

**Figure 7. Wave transmitted using BPSK subcarrier modulation**



### 3.2.1 Character transmission format for answer frame

The character format is the same as for input data transfer (Figure 4). The transmitted frames are made up of an SOF, data, a CRC and an EOF (Figure 10). As with an input data transfer, if an error occurs, the reader does not issue an error code to the ST25TB02K, but it should be able to detect it and manage the situation. The data transfer rate is 106 Kbits/second.

### 3.2.2 Answer start of frame

The SOF described in Figure 8 is composed of:

- one falling edge,
- followed by 10 ETUs at logic-0,
- followed by 2 ETUs at logic-1.

**Figure 8. Answer start of frame**

|     | b0 | b1 | b2 | b3 | b4 | b5 | b6 | b7 | b8 | b9 | b10 | b11 |
|-----|----|----|----|----|----|----|----|----|----|----|-----|-----|
| ETU | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1   | 1   |

### 3.2.3 Answer end of frame

The EOF shown in Figure 9 is composed of:

- one falling edge,
- followed by 10 ETUs at logic-0,
- followed by 2 ETUs at logic-1.

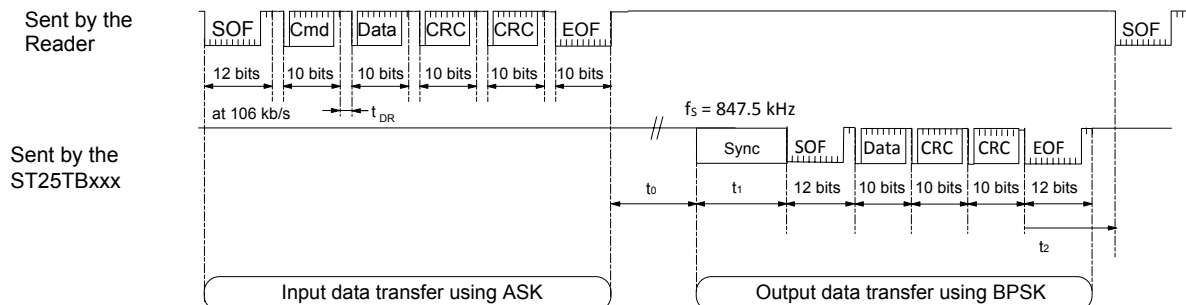
Figure 9. Answer end of frame

|     | b0 | b1 | b2 | b3 | b4 | b5 | b6 | b7 | b8 | b9 | b10 | b11 |
|-----|----|----|----|----|----|----|----|----|----|----|-----|-----|
| ETU | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1   | 1   |

## 3.3 Transmission frame

Between the request data transfer and the answer data transfer, all ASK and BPSK modulations are suspended for a minimum time of  $t_0 = 128/f_S$ . This delay allows the reader to switch from Transmission to Reception mode. It is repeated after each frame. After  $t_0$ , the 13.56 MHz carrier frequency is modulated by the ST25TB02K at 847 kHz for a period of  $t_1 = 128/f_S$  to allow the reader to synchronize. After  $t_1$ , the first phase transition generated by the ST25TB02K forms the start bit ('0') of the answer SOF. After the falling edge of the answer EOF, the reader waits a minimum time,  $t_2$ , before sending a new request frame to the ST25TB02K.

Figure 10. Example of a complete transmission frame





### 3.4 CRC

The 16-bit CRC used by the ST25TB02K is generated in compliance with the ISO14443 Type B recommendation. For further information, please see Appendix A [ISO-14443 Type B CRC calculation](#). The initial register contents are all 1s: FFFFh.

The two-byte CRC is present in every request and in every answer frame, before the EOF. The CRC is calculated on all the bytes between SOF (not included) and the CRC field.

Upon reception of a request from a reader, the ST25TB02K verifies that the CRC value is valid. If it is invalid, the ST25TB02K discards the frame and does not answer the reader.

Upon reception of an answer from the ST25TB02K, the reader should verify the validity of the CRC. In case of error, the actions to be taken are the reader designer's responsibility.

The CRC is transmitted with the least significant byte first and each byte is transmitted with the least significant bit first.

Figure 11. CRC transmission rules



## 4 Memory mapping

The ST25TB02K is organized as 64 blocks of 32 bits as shown in Table 3. All blocks are accessible by the Read\_block command. Depending on the write access, they can be updated by the Write\_block command. A Write\_block updates all the 32 bits of the block.

**Table 3. ST25TB02K memory mapping**

| Block address | MSB                 | 32-bit block |     |     |       | LSB |                    | Description |
|---------------|---------------------|--------------|-----|-----|-------|-----|--------------------|-------------|
|               | b31                 | b24 b23      | b16 | b15 | b8 b7 | b0  |                    |             |
| 0             | 32-bit Boolean area |              |     |     |       |     | Resettable OTP bit |             |
| 1             | 32-bit Boolean area |              |     |     |       |     |                    |             |
| 2             | 32-bit Boolean area |              |     |     |       |     |                    |             |
| 3             | 32-bit Boolean area |              |     |     |       |     |                    |             |
| 4             | 32-bit Boolean area |              |     |     |       |     |                    |             |
| 5             | 32-bit Boolean area |              |     |     |       |     | Count down counter |             |
| 6             | 32-bit Boolean area |              |     |     |       |     |                    |             |
| 7             | User area           |              |     |     |       |     | Lockable EEPROM    |             |
| 8             | User area           |              |     |     |       |     |                    |             |
| 9             | User area           |              |     |     |       |     |                    |             |
| 10            | User area           |              |     |     |       |     |                    |             |
| 11            | User area           |              |     |     |       |     |                    |             |
| 12            | User area           |              |     |     |       |     |                    |             |
| 13            | User area           |              |     |     |       |     |                    |             |
| 14            | User area           |              |     |     |       |     |                    |             |
| 15            | User area           |              |     |     |       |     |                    |             |
| 16            | User area           |              |     |     |       |     | EEPROM             |             |
| ...           | User area           |              |     |     |       |     |                    |             |
| 63            | User area           |              |     |     |       |     |                    |             |
| 255           | OTP_Lock_Reg        | Reserved     |     |     |       |     | System OTP bits    |             |
| UID0          | 64 bits UID area    |              |     |     |       |     | ROM                |             |
| UID1          |                     |              |     |     |       |     |                    |             |

## 4.1 EEPROM area

### 4.1.1 Block 0-4: resettable OTP area

This area contains five individual 32-bit Boolean words (see [Table 4](#) for a map of the area). A Write\_block command doesn't erase the previous contents of the block as the write cycle is not preceded by an auto-erase cycle. This feature can be used to reset selected bits from 1 to 0. All bits previously at 0 remain unchanged. When the 32 bits of a block are all at 0, the block is empty, and cannot be updated any more. See [Figure 1](#) and [Figure 2](#) for examples of the result of the Write\_block command in the resettable OTP area.

**Table 4. Resettable OTP area (addresses 0 to 4)**

| Block Address | MSB                 | 32-bit block |         |       | LSB | Description           |
|---------------|---------------------|--------------|---------|-------|-----|-----------------------|
|               | b31                 | b24 b23      | b16 b15 | b8 b7 | b0  |                       |
| 0             | 32-bit Boolean area |              |         |       |     | Resettable<br>OTP bit |
| 1             | 32-bit Boolean area |              |         |       |     |                       |
| 2             | 32-bit Boolean area |              |         |       |     |                       |
| 3             | 32-bit Boolean area |              |         |       |     |                       |
| 4             | 32-bit Boolean area |              |         |       |     |                       |

**Figure 12. Write\_block update in Standard mode (binary format)**

|                               |     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |  |  |  |  |  |    |
|-------------------------------|-----|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|--|--|--|--|--|--|----|
|                               | b31 |     |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |  |  |  |  |  | b0 |
| Previous data stored in block | 1   | ... | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |    |
| Data to be written            | 1   | ... | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |    |
| New data stored in block      | 1   | ... | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |    |

The five 32-bit blocks making up the resettable OTP area can be erased in one go by adding an auto-erase cycle to the Write\_block command. An auto-erase cycle is added each time one reload mode is activated. The reload mode is implemented through a specific update of the 32-bit binary counter located at block address 6 (see [Section 4.2 32-bit binary counters](#) for details).

**Figure 13. Write\_block update in Reload mode (binary format)**

|                               |     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |  |  |  |  |  |    |
|-------------------------------|-----|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|--|--|--|--|--|--|----|
|                               | b31 |     |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |  |  |  |  |  | b0 |
| Previous data stored in block | 1   | ... | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |    |
| Data to be written            | 1   | ... | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |    |
| New data stored in block      | 1   | ... | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |    |

ai07659

### 4.1.2 Block 7-63

The 57 blocks between addresses 7 and 63 are EEPROM blocks of 32 bits each (228 bytes in total). (See [Table 5. EEPROM area \(addresses 7 to 63\)](#) for a map of the area.) These blocks can be accessed using the Read\_block and Write\_block commands. The Write\_block command for the EEPROM area always includes an auto-erase cycle prior to the write cycle.

Blocks 7 to 15 can be write-protected. Write access is controlled by the 8 bits of the OTP\_Lock\_Reg located at block address 255 (see “[Section 4.3.1 OTP\\_Lock\\_Reg](#)” for details). Once protected, these blocks (7 to 15) cannot be unprotected.

**Table 5. EEPROM area (addresses 7 to 63)**

| Block Address | MSB       |         | 32-bit block |       | LSB |                 | Description |
|---------------|-----------|---------|--------------|-------|-----|-----------------|-------------|
|               | b31       | b24 b23 | b16 b15      | b8 b7 | b0  |                 |             |
| 7             | User area |         |              |       |     | Lockable EEPROM |             |
| 8             | User area |         |              |       |     |                 |             |
| 9             | User area |         |              |       |     |                 |             |
| 10            | User area |         |              |       |     |                 |             |
| 11            | User area |         |              |       |     |                 |             |
| 12            | User area |         |              |       |     |                 |             |
| 13            | User area |         |              |       |     |                 |             |
| 14            | User area |         |              |       |     |                 |             |
| 15            | User area |         |              |       |     |                 |             |
| 16            | User area |         |              |       |     | EEPROM          |             |
| ...           | User area |         |              |       |     |                 |             |
| 63            | User area |         |              |       |     |                 |             |

## 4.2 32-bit binary counters

The two 32-bit binary counters are located at block addresses 5 and 6. The **ST25TB02K** uses dedicated logic that only allows the update of a counter if the new value is lower than the previous one. This feature allows the application to count down by steps of 1 or more. The initial value in Counter 5 is FFFF FFEh and is FFFF FFFFh in Counter 6. When the reached value is 0000 0000h, the counter is empty and cannot be reloaded. For each counter 5 and 6, the update is done by issuing the Write\_block command. The Write\_block command writes the new 32-bit value to the counter block address. Table 6 shows examples of how the counters operate.

The counter programming cycles are protected by automated antitearing logic. This function allows the counter value to be protected in case of power down within the programming cycle. In case of power down, the counter value is not updated and the previous value continues to be stored.

**Table 6. Binary counter (addresses 5 to 6)**

| Block Address | 32-bit block        |                                 |                                 |                               | LSB            | Description        |
|---------------|---------------------|---------------------------------|---------------------------------|-------------------------------|----------------|--------------------|
|               | b <sub>31</sub>     | b <sub>24</sub> b <sub>23</sub> | b <sub>16</sub> b <sub>15</sub> | b <sub>8</sub> b <sub>7</sub> | b <sub>0</sub> |                    |
| 5             | 32-bit Boolean area |                                 |                                 |                               |                | Count down counter |
| 6             | 32-bit Boolean area |                                 |                                 |                               |                |                    |

**Figure 14. Countdown example (binary format)**

|                       | b <sub>31</sub> |     |   |   |   |   |   |   |   |   |   |   |   |   | b <sub>0</sub> |
|-----------------------|-----------------|-----|---|---|---|---|---|---|---|---|---|---|---|---|----------------|
| Initial data          | 1               | ... | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1              |
| 1-unit decrement      | 1               | ... | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0              |
| 1-unit decrement      | 1               | ... | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |                |
| 1-unit decrement      | 1               | ... | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |                |
| 8-unit decrement      | 1               | ... | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |                |
| Increment not allowed | 1               | ... | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |                |

The counter with block address 6 controls the reload mode used to reset the resettable OTP area (addresses 0 to 4). Bits b<sub>31</sub> to b<sub>21</sub> act as an 11-bit Reload counter; whenever one of these 11 bits is updated, the **ST25TB02K** detects the change and adds an Erase cycle to the Write\_block command for locations 0 to 4 (see the Block 0 - 4: resettable OTP area paragraph).

The Erase cycle remains active until a Power-off or a Select command is issued.

The **ST25TB02K**'s resettable OTP area can be reloaded up to 2 047 times ( $2^{11}-1$ ).

### 4.3 System area

This area is used to modify the settings of the [ST25TB02K](#). It contains 2 registers: OTP\_Lock\_Reg and ST Reserved. See [Table 7. System area](#) for a map of this area.

A Write\_block command in this area will not erase the previous contents. Selected bits can thus be set from 1 to 0. All bits previously at 0 remain unchanged. Once all the 32 bits of a block are at 0, the block is empty and cannot be updated any more.

**Table 7. System area**

| Block Address | 32-bit block |     |     |     |     |             |    |     | Description |
|---------------|--------------|-----|-----|-----|-----|-------------|----|-----|-------------|
|               | MSB          |     |     |     |     |             |    | LSB |             |
|               | b31          | b24 | b23 | b16 | b15 | b14         | b7 | b0  |             |
| 255           | OTP_Lock_Reg |     |     |     | 0   | ST reserved |    |     | OTP         |

#### 4.3.1 OTP\_Lock\_Reg

The 8 bits,  $b_{31}$  to  $b_{24}$ , of the System area (block address 255) are used as OTP\_Lock\_Reg bits in the [ST25TB02K](#). They control the write access to the 9 EEPROM blocks with addresses 7 to 15 as follows:

- When  $b_{24}$  is at 0, blocks 7 and 8 are write-protected
- When  $b_{25}$  is at 0, block 9 is write-protected
- When  $b_{26}$  is at 0, block 10 is write-protected
- When  $b_{27}$  is at 0, block 11 is write-protected
- When  $b_{28}$  is at 0, block 12 is write-protected
- When  $b_{29}$  is at 0, block 13 is write-protected
- When  $b_{30}$  is at 0, block 14 is write-protected
- When  $b_{31}$  is at 0, block 15 is write-protected.

The OTP\_Lock\_Reg bits cannot be erased. Once write-protected, EEPROM blocks behave like ROM blocks and cannot be unprotected.

After any modification of the OTP\_Lock\_Reg bits, it is necessary to send a Select command with a valid Chip\_ID to the [ST25TB02K](#) in order to load the block write protection into the logic.

---

## 5 ST25TB02K operation

---

All commands, data and CRC are transmitted to the ST25TB02K as 10-bit characters using ASK modulation. The start bit of the 10 bits,  $b_0$ , is sent first. The command frame received by the ST25TB02K at the antenna is demodulated by the 10% ASK demodulator, and decoded by the internal logic. Prior to any operation, the ST25TB02K must have been selected by a Select command. Each frame transmitted to the ST25TB02K must start with a start of frame, followed by one or more data characters, two CRC bytes and the final end of frame. When an invalid frame is decoded by the ST25TB02K (wrong command or CRC error), the memory does not return any error code.

When a valid frame is received, the ST25TB02K may have to return data to the reader. In this case, data is returned using BPSK encoding, in the form of 10-bit characters framed by an SOF and an EOF. The transfer is ended by the ST25TB02K sending the 2 CRC bytes and the EOF.

## 6 ST25TB02K states

The ST25TB02K can be switched into different states. Depending on the current state of the ST25TB02K, its logic will only answer to specific commands. These states are mainly used during the anticollision sequence, to identify and to access the ST25TB02K in a very short time. The ST25TB02K provides 6 different states, as described in the following paragraphs and in Figure 15.

### 6.1 Power-off state

The ST25TB02K is in Power-off state when the electromagnetic field around the tag is not strong enough. In this state, the ST25TB02K does not respond to any command.

### 6.2 Ready state

When the electromagnetic field is strong enough, the ST25TB02K enters the Ready state. After Power-up, the Chip\_ID is initialized with a random value. The whole logic is reset and remains in this state until an Initiate() command is issued. Any other command will be ignored by the ST25TB02K.

### 6.3 Inventory state

The ST25TB02K switches from the Ready to the Inventory state after an Initiate() command has been issued. In Inventory state, the ST25TB02K will respond to any anticollision commands: Initiate(), Pcall16() and Slot\_marker(), and then remain in the Inventory state. It will switch to the Selected state after a Select(Chip\_ID) command is issued, if the Chip\_ID in the command matches its own. If not, it will remain in Inventory state.

### 6.4 Selected state

In Selected state, the ST25TB02K is active and responds to all Read\_block(), Write\_block() and Get\_UID() commands. When an ST25TB02K has entered the Selected state, it no longer responds to anticollision commands. So that the reader can access another tag, the ST25TB02K can be switched to the Deselected state by sending a Select(Chip\_ID) with a Chip\_ID that does not match its own, or it can be placed in Deactivated state by issuing a Completion() command. Only one ST25TB02K can be in Selected state at a time.

### 6.5 Deselected state

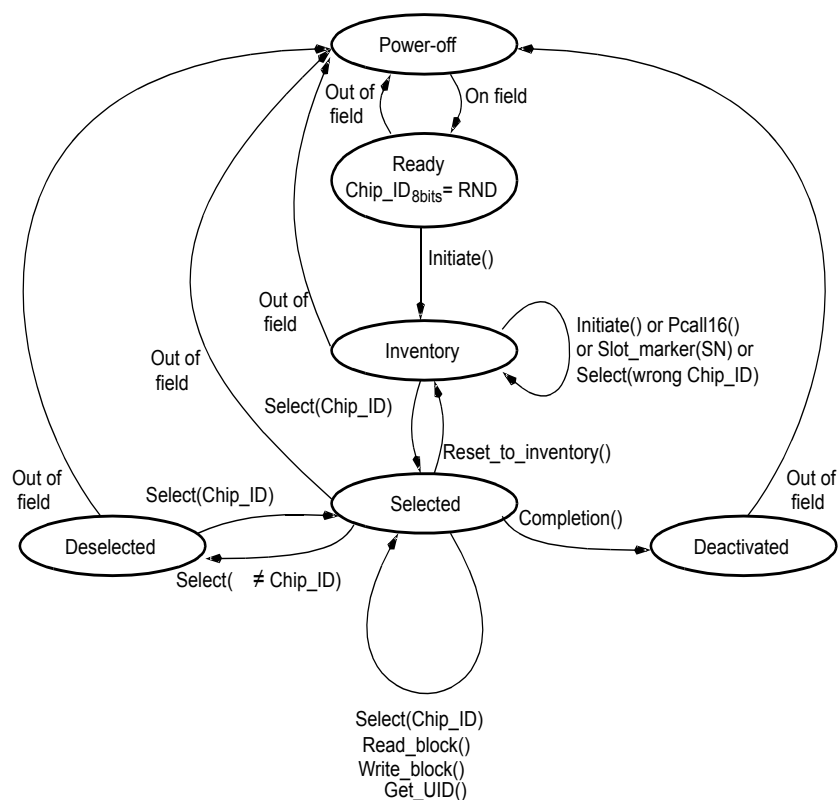
Once the ST25TB02K is in Deselected state, only a Select(Chip\_ID) command with a Chip\_ID matching its own can switch it back to Selected state. All other commands are ignored.



## 6.6 Deactivated state

When in this state, the ST25TB02K can only be turned off. All commands are ignored.

**Figure 15. State transition diagram**



## 7 Anticollision

The **ST25TB02K** provides an anticollision mechanism that searches for the **Chip\_ID** of each device that is present in the reader field range. When known, the **Chip\_ID** is used to select an **ST25TB02K** individually, and access its memory. The anticollision sequence is managed by the reader through a set of commands described in [Section 8 ST25TB02K commands](#):

- **Initiate()**
- **Pcall16()**
- **Slot\_marker()**.

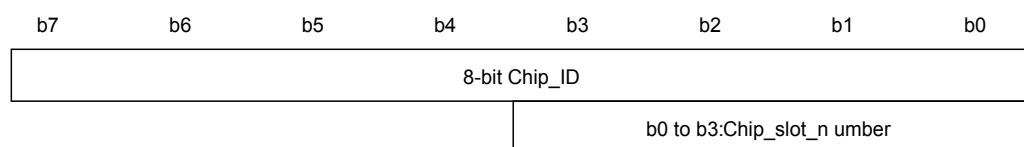
The reader is the master of the communication with one or more **ST25TB02K** device(s). It initiates the tag communication activity by issuing an **Initiate()**, **Pcall16()** or **Slot\_marker()** command to prompt the **ST25TB02K** to answer. During the anticollision sequence, it might happen that two or more **ST25TB02K** devices respond simultaneously, so causing a collision. The command set allows the reader to handle the sequence, to separate **ST25TB02K** transmissions into different time slots. Once the anticollision sequence has completed, **ST25TB02K** communication is fully under the control of the reader, allowing only one **ST25TB02K** to transmit at a time.

The Anticollision scheme is based on the definition of time slots during which the **ST25TB02K** devices are invited to answer with minimum identification data: the **Chip\_ID**. The number of slots is fixed at 16 for the **Pcall16()** command. For the **Initiate()** command, there is no slot and the **ST25TB02K** answers after the command is issued. **ST25TB02K** devices are allowed to answer only once during the anticollision sequence. Consequently, even if there are several **ST25TB02K** devices present in the reader field, there will probably be a slot in which only one **ST25TB02K** answers, allowing the reader to capture its **Chip\_ID**. Using the **Chip\_ID**, the reader can then establish a communication channel with the identified **ST25TB02K**. The purpose of the anticollision sequence is to allow the reader to select one **ST25TB02K** at a time.

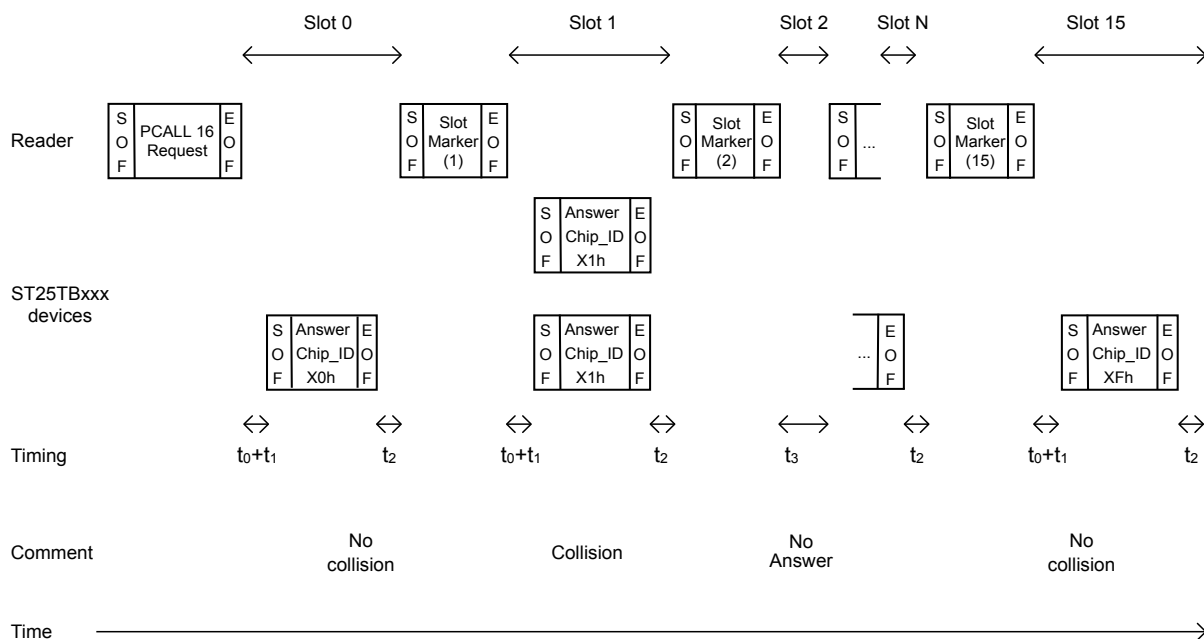
The **ST25TB02K** is given an 8-bit **Chip\_ID** value used by the reader to select only one among up to 256 tags present within its field range. The **Chip\_ID** is initialized with a random value during the Ready state, or after an **Initiate()** command in the Inventory state.

The four least significant bits (**b<sub>0</sub>** to **b<sub>3</sub>**) of the **Chip\_ID** are also known as the **Chip\_slot\_number**. This 4-bit value is used by the **Pcall16()** and **Slot\_marker()** commands during the anticollision sequence in the Inventory state.

**Figure 16. ST25TB02K Chip\_ID description**



Each time the **ST25TB02K** receives a **Pcall16()** command, the **Chip\_slot\_number** is given a new 4-bit random value. If the new value is 0000<sub>b</sub>, the **ST25TB02K** returns its whole 8-bit **Chip\_ID** in its answer to the **Pcall16()** command. The **Pcall16()** command is also used to define the slot number 0 of the anticollision sequence. When the **ST25TB02K** receives the **Slot\_marker(SN)** command, it compares its **Chip\_slot\_number** with the **Slot\_number** parameter (SN). If they match, the **ST25TB02K** returns its **Chip\_ID** as a response to the command. If they do not, the **ST25TB02K** does not answer. The **Slot\_marker(SN)** command is used to define all the anticollision slot numbers from 1 to 15.

**Figure 17. Description of a possible anticollision sequence**


1. The value X in the answer Chip\_ID means a random hexadecimal character from 0 to F.

## 7.1 Description of an anticollision sequence

The anticollision sequence is initiated by the Initiate() command which triggers all the **ST25TB02K** devices that are present in the reader field range, and that are in Inventory state. Only **ST25TB02K** devices in Inventory state will respond to the Pcall16() and Slot\_marker(SN) anticollision commands.

A new **ST25TB02K** introduced in the field range during the anticollision sequence will not be taken into account as it will not respond to the Pcall16() or Slot\_marker(SN) command (Ready state). To be considered during the anticollision sequence, it must have received the Initiate() command and entered the Inventory state.

Table 8 shows the elements of a standard anticollision sequence. (See Table 9 for an example.)

**Table 8. Standard anticollision sequence**

|         |         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Step 1  | Init:   | Send Initiate(). <ul style="list-style-type: none"> <li>If no answer is detected, go to step1.</li> <li>If only 1 answer is detected, select and access the <a href="#">ST25TB02K</a>. After accessing the <a href="#">ST25TB02K</a>, deselect the tag and go to step1.</li> <li>If a collision (many answers) is detected, go to step2.</li> </ul>                                                                                                                                                                                           |
| Step 2  | Slot 0  | Send Pcall16(). <ul style="list-style-type: none"> <li>If no answer or collision is detected, go to step3.</li> <li>If 1 answer is detected, store the Chip_ID, Send Select() and go to step3.</li> </ul>                                                                                                                                                                                                                                                                                                                                     |
| Step 3  | Slot 1  | Send Slot_marker(1). <ul style="list-style-type: none"> <li>If no answer or collision is detected, go to step4.</li> <li>If 1 answer is detected, store the Chip_ID, Send Select() and go to step4.</li> </ul>                                                                                                                                                                                                                                                                                                                                |
| Step 4  | Slot 2  | Send Slot_marker(2). <ul style="list-style-type: none"> <li>If no answer or collision is detected, go to step5.</li> <li>If 1 answer is detected, store the Chip_ID, Send Select() and go to step5.</li> </ul>                                                                                                                                                                                                                                                                                                                                |
| Step N  | Slop N  | Send Slot_marker(3 up to 14) ... <ul style="list-style-type: none"> <li>If no answer or collision is detected, go to stepN+1.</li> <li>If 1 answer is detected, store the Chip_ID, Send Select() and go to stepN+1.</li> </ul>                                                                                                                                                                                                                                                                                                                |
| Step 17 | Slot 15 | Send Slot_marker(15). <ul style="list-style-type: none"> <li>If no answer or collision is detected, go to step18.</li> <li>If 1 answer is detected, store the Chip_ID, Send Select() and go to step18.</li> </ul>                                                                                                                                                                                                                                                                                                                             |
| Step 18 | -       | All the slots have been generated and the Chip_ID values should be stored into the reader memory. Issue the Select(Chip_ID) command and access each identified <a href="#">ST25TB02K</a> one by one. After accessing each <a href="#">ST25TB02K</a> , switch them into Deselected or Deactivated state, depending on the application needs. <ul style="list-style-type: none"> <li>If collisions were detected between Step2 and Step17, go to Step2.</li> <li>If no collision was detected between Step2 and Step17, go to Step1.</li> </ul> |

After each Slot\_marker() command, there may be no answer, one or several answers from the [ST25TB02K](#) devices. The reader must handle all the cases and store all the Chip\_IDs, correctly decoded. At the end of the anticollision sequence, after Slot\_marker(15), the reader can start working with one [ST25TB02K](#) by issuing a Select() command containing the desired Chip\_ID. If a collision is detected, the reader has to generate a new sequence in order to identify all unidentified [ST25TB02K](#) devices in the field. The anticollision sequence can stop when all [ST25TB02K](#) devices have been identified.

[Table 9](#) gives an example of anticollision sequence, the cells containing (\*) highlight the fact that the related tags are not yet identified. When the tag is identified, in the table the (\*) changes to bold character.

**Table 9. Example of an anticollision sequence**

| Command        | Tag1    | Tag2       | Tag3       | Tag4    | Tag5    | Tag6    | Tag7    | Tag8    | Comment                                                        |
|----------------|---------|------------|------------|---------|---------|---------|---------|---------|----------------------------------------------------------------|
|                | Chip_ID | Chip_ID    | Chip_ID    | Chip_ID | Chip_ID | Chip_ID | Chip_ID | Chip_ID |                                                                |
| READY state    | 28h(*)  | 75h(*)     | 40h(*)     | 01h(*)  | 02h(*)  | FEh(*)  | A9h(*)  | 7Ch(*)  | Each tag gets a random Chip_ID                                 |
| INITIATE()     | 40h(*)  | 13h(*)     | 3Fh(*)     | 4Ah(*)  | 50h(*)  | 48h(*)  | 52h(*)  | 7Ch(*)  | Each tag get a new random Chip_ID. All tags answer: collisions |
| PCALL16()      | 45h(*)  | 12h(*)     | 30h(*)     | 43h(*)  | 55h(*)  | 43h(*)  | 53h(*)  | 73h(*)  | All CHIP_SLOT_NUMBERS get a new random value                   |
| SELECT(30h)    | (*)     | (*)        | (*)30h     | (*)     | (*)     | (*)     | (*)     | (*)     | Slot0: only one answer                                         |
| SLOT_MARKER(1) | (*)     | (*)        | <b>30h</b> | (*)     | (*)     | (*)-    | (*)-    | (*)     | Slot1: no answer                                               |
| SLOT_MARKER(2) | (*)     | 12h(*)     | -          | (*)     | (*)     | (*)     | (*)     | (*)     | Slot2: only one answer                                         |
| SELECT(12h)    | (*)     | <b>12h</b> | -          | (*)     | (*)     | (*)     | (*)     | (*)     | Tag2 is identified                                             |
| SLOT_MARKER(3) | (*)     | -          | -          | 43h(*)  | (*)     | 43h(*)  | 53h(*)  | 73h(*)  | Slot3: collision                                               |
| SLOT_MARKER(4) | (*)     | -          | -          | (*)     | (*)     | (*)     | (*)     | (*)     | Slot4: no answer                                               |

| Command        | Tag1       | Tag2    | Tag3    | Tag4       | Tag5       | Tag6       | Tag7       | Tag8       | Comment                                           |
|----------------|------------|---------|---------|------------|------------|------------|------------|------------|---------------------------------------------------|
|                | Chip_ID    | Chip_ID | Chip_ID | Chip_ID    | Chip_ID    | Chip_ID    | Chip_ID    | Chip_ID    |                                                   |
| SLOT_MARKER(5) | 45h(*)     | -       | -       | (*)        | 55h(*)     | (*)        | (*)        | (*)        | Slot5: collision                                  |
| SLOT_MARKER(6) | (*)        | -       | -       | (*)        | (*)        | (*)        | (*)        | (*)        | Slot6: no answer                                  |
| SLOT_MARKER(N) | (*)        | -       | -       | (*)        | (*)        | (*)        | (*)        | (*)        | SlotN: no answer                                  |
| SLOT_MARKER(F) | (*)        | -       | -       | (*)        | (*)        | (*)        | (*)        | (*)        | SlotF: no answer                                  |
| PCALL16()      | 40h(*)     | -       | -       | 41h(*)     | 53h(*)     | 42h(*)     | 50h(*)     | 74h(*)     | All CHIP_SLOT_ NUMBERS get a new random value     |
|                | 40h(*)     | -       | -       | (*)        | (*)        | (*)        | 50h(*)     | (*)        | Slot0: collision                                  |
| SLOT_MARKER(1) | (*)        | -       | -       | 41h(*)     | (*)        | (*)        | (*)        | (*)        | Slot1: only one answer                            |
| SELECT(41h)    | (*)        | -       | -       | <b>41h</b> | (*)        | (*)        | (*)        | (*)        | Tag4 is identified                                |
| SLOT_MARKER(2) | (*)        | -       | -       | -          | (*)        | 42h(*)     | (*)        | (*)        | Slot2: only one answer                            |
| SELECT(42h)    | (*)        | -       | -       | -          | (*)        | <b>42h</b> | (*)        | (*)        | Tag6 is identified                                |
| SLOT_MARKER(3) | (*)        | -       | -       | -          | 53h(*)     | -          | (*)        | (*)        | Slot3: only one answer                            |
| SELECT(53h)    | (*)        | -       | -       | -          | <b>53h</b> | -          | (*)        | (*)        | Tag5 is identified                                |
| SLOT_MARKER(4) | (*)        | -       | -       | -          | -          | -          | (*)        | 74h(*)     | Slot4: only one answer                            |
| SELECT(74h)    | (*)        | -       | -       | -          | -          | -          | (*)        | <b>74h</b> | Tag8 is identified                                |
| SLOT_MARKER(N) | (*)        | -       | -       | -          | -          | -          | (*)        | -          | SlotN: no answer                                  |
| PCALL16()      | 41h(*)     | -       | -       | -          | -          | -          | 50h(*)     | -          | All CHIP_SLOT_ NUMBERS get a new random value     |
|                | (*)        | -       | -       | -          | -          | -          | 50h(*)     | -          | Slot0: only one answer                            |
| SELECT(50h)    | (*)        | -       | -       | -          | -          | -          | <b>50h</b> | -          | Tag7 is identified                                |
| SLOT_MARKER(1) | 41h(*)     | -       | -       | -          | -          | -          | -          | -          | Slot1: only one answer but already found for tag4 |
| SLOT_MARKER(N) | (*)        | -       | -       | -          | -          | -          | -          | -          | SlotN: only one answer                            |
| PCALL16()      | 43h(*)     | -       | -       | -          | -          | -          | -          | -          | All CHIP_SLOT_ NUMBERS get a new random value     |
|                | (*)        | -       | -       | -          | -          | -          | -          | -          | Slot0: only one answer                            |
| SLOT_MARKER(3) | 43h(*)     | -       | -       | -          | -          | -          | -          | -          | Slot3: only one answer                            |
| SELECT(43h)    | <b>43h</b> | -       | -       | -          | -          | -          | -          | -          | Tag1 is identified                                |
| -              | (*)        | -       | -       | -          | -          | -          | -          | -          | All tags are identified                           |

## 8 ST25TB02K commands

See the paragraphs below for a detailed description of the commands available on the [ST25TB02K](#). The commands and their hexadecimal codes are summarized in [Table 10](#). A brief is given in Appendix B [ST25TB02K command brief](#).

**Table 10. Command code**

| Hexadecimal code | Command                 |
|------------------|-------------------------|
| 06h-00h          | Initiate()              |
| 06h-04h          | Pcall16()               |
| x6h              | Slot_marker (SN)        |
| 08h              | Read_block(Addr)        |
| 09h              | Write_block(Addr, Data) |
| 0Bh              | Get_UID()               |
| 0Ch              | Reset_to_inventory      |
| 0Eh              | Select(Chip_ID)         |
| 0Fh              | Completion()            |

## 8.1 Initiate() command

Command code = 06h - 00h

Initiate() is used to initiate the anticollision sequence of the **ST25TB02K**. On receiving the Initiate() command, all **ST25TB02K** devices in Ready state switch to Inventory state, set a new 8-bit Chip\_ID random value, and return their Chip\_ID value. This command is useful when only one **ST25TB02K** in Ready state is present in the reader field range. It speeds up the Chip\_ID search process. The Chip\_slot\_number is not used during Initiate() command access.

**Figure 18. Initiate request format**

|     |          |     |                  |                  |     |
|-----|----------|-----|------------------|------------------|-----|
| SOF | Initiate |     | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|     | 06h      | 00h | 8 bits           | 8 bits           |     |

Request parameter:

- No parameter

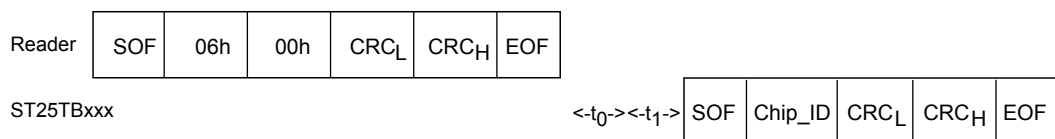
**Figure 19. Initiate response format**

|     |         |                  |                  |     |
|-----|---------|------------------|------------------|-----|
| SOF | Chip_ID | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|     | 8 bits  | 8 bits           | 8 bits           |     |

Response parameter:

- Chip\_ID of the **ST25TB02K**

**Figure 20. Initiate frame exchange between reader and ST25TB02K**



## 8.2 Pcall16() command

Command code = 06h - 04h

The **ST25TB02K** must be in Inventory state to interpret the Pcall16() command.

On receiving the Pcall16() command, the **ST25TB02K** first generates a new random Chip\_slot\_number value (in the 4 least significant bits of the Chip\_ID). Chip\_slot\_number can take on a value between 0 and 15 (1111<sub>b</sub>). The value is retained until a new Pcall16() or Initiate() command is issued, or until the **ST25TB02K** is powered off. The new Chip\_slot\_number value is then compared with the value 0000<sub>b</sub>. If they match, the **ST25TB02K** returns its Chip\_ID value. If not, the **ST25TB02K** does not send any response.

The Pcall16() command, used together with the Slot\_marker() command, allows the reader to search for all the Chip\_IDs when there are more than one **ST25TB02K** device in Inventory state present in the reader field range.

**Figure 21. Pcall16 request format**

|     |         |     |                  |                  |     |
|-----|---------|-----|------------------|------------------|-----|
| SOF | PCALL16 |     | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|     | 06h     | 04h | 8 bits           | 8 bits           |     |

Request parameter:

- No parameter

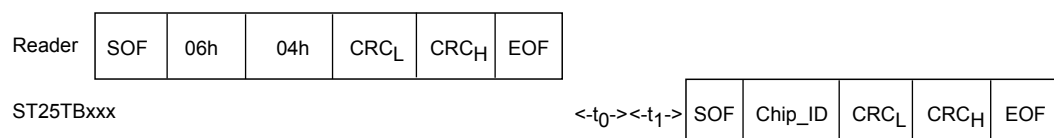
**Figure 22. Pcall16 response format**

|     |         |                  |                  |     |
|-----|---------|------------------|------------------|-----|
| SOF | Chip_ID | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|     | 8 bits  | 8 bits           | 8 bits           |     |

Response parameter:

- Chip\_ID of the **ST25TB02K**

**Figure 23. Pcall16 frame exchange between reader and ST25TB02K**





### 8.3 Slot\_marker(SN) command

Command code = x6h

The **ST25TB02K** must be in Inventory state to interpret the Slot\_marker(SN) command.

The Slot\_marker byte code is divided into two parts:

- b<sub>3</sub> to b<sub>0</sub>: 4-bit command code
- with fixed value 6.
- b<sub>7</sub> to b<sub>4</sub>: 4 bits known as the Slot\_number (SN). They assume a value between 1 and 15. The value 0 is reserved by the Pcall16() command.

On receiving the Slot\_marker() command, the **ST25TB02K** compares its Chip\_slot\_number value with the Slot\_number value given in the command code. If they match, the **ST25TB02K** returns its Chip\_ID value. If not, the **ST25TB02K** does not send any response.

The Slot\_marker() command, used together with the Pcall16() command, allows the reader to search for all the Chip\_IDs when there are more than one **ST25TB02K** device in Inventory state present in the reader field range.

**Figure 24. Slot\_marker request format**

| SOF | Slot_marker | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|-------------|------------------|------------------|-----|
|     | X6h         | 8 bits           | 8 bits           |     |

Request parameter:

- x: Slot number

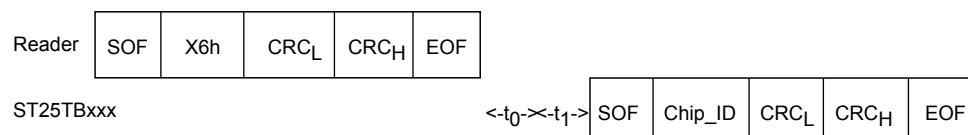
**Figure 25. Slot\_marker response format**

| SOF | Chip_ID | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|---------|------------------|------------------|-----|
|     | 8 bits  | 8 bits           | 8 bits           |     |

Response parameters:

- Chip\_ID of the **ST25TB02K**

**Figure 26. Slot\_marker frame exchange between reader and ST25TB02K**



## 8.4 Select(Chip\_ID) command

Command code = 0Eh

The Select() command allows the ST25TB02K to enter the Selected state. Until this command is issued, the ST25TB02K will not accept any other command, except for Initiate(), Pcall16() and Slot\_marker(). The Select() command returns the 8 bits of the Chip\_ID value. An ST25TB02K in Selected state, that receives a Select() command with a Chip\_ID that does not match its own is automatically switched to Deselected state.

**Figure 27. Select request format**

| SOF | Select | Chip_ID | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|--------|---------|------------------|------------------|-----|
|     | 0Eh    | 8 bits  | 8 bits           | 8 bits           |     |

Request parameter:

- 8-bit Chip\_ID stored during the anticollision sequence

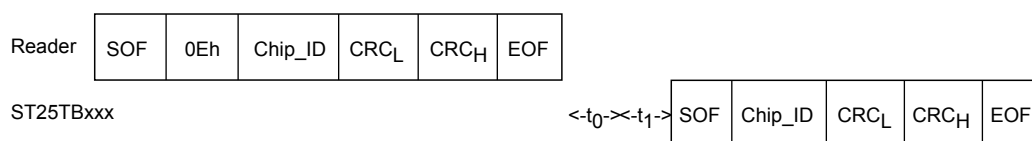
**Figure 28. Select response format**

| SOF | Chip_ID | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|---------|------------------|------------------|-----|
|     | 8 bits  | 8 bits           | 8 bits           |     |

Response parameters:

- Chip\_ID of the selected tag. Must be equal to the transmitted Chip\_ID

**Figure 29. Select frame exchange between reader and ST25TB02K**



## 8.5 Completion() command

Command code = 0Fh

On receiving the Completion() command, an ST25TB02K in Selected state switches to Deactivated state and stops decoding any new commands. The ST25TB02K is then locked in this state until a complete reset (tag out of the field range). A new ST25TB02K can thus be accessed through a Select() command without having to remove the previous one from the field. The Completion() command does not generate a response.

All ST25TB02K devices not in Selected state ignore the Completion() command.

**Figure 30. Completion request format**

| SOF | Completion | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|------------|------------------|------------------|-----|
|     | 0Fh        | 8 bits           | 8 bits           |     |

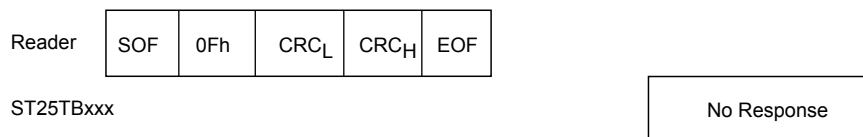
Request parameters:

- No parameter

**Figure 31. Completion response format**

|             |
|-------------|
| No Response |
|             |

**Figure 32. Completion frame exchange between reader and ST25TB02K**



## 8.6 Reset\_to\_inventory() command

Command code = 0Ch

On receiving the Reset\_to\_inventory() command, all ST25TB02K devices in Selected state revert to Inventory state. The concerned ST25TB02K devices are thus resubmitted to the anticollision sequence. This command is useful when two ST25TB02K devices with the same 8-bit Chip\_ID happen to be in Selected state at the same time. Forcing them to go through the anticollision sequence again allows the reader to generate new Pcall16() commands and so, to set new random Chip\_IDs.

The Reset\_to\_inventory() command does not generate a response.

All ST25TB02K devices that are not in Selected state ignore the Reset\_to\_inventory() command.

**Figure 33. Reset\_to\_inventory request format**

|     |                    |                  |                  |     |
|-----|--------------------|------------------|------------------|-----|
| SOF | RESET_TO_INVENTORY | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|     | 0Ch                | 8 bits           | 8 bits           |     |

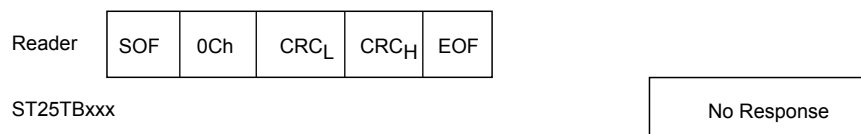
Request parameter:

- No parameter

**Figure 34. Reset\_to\_inventory response format**

|             |
|-------------|
| No Response |
|             |

**Figure 35. Reset\_to\_inventory frame exchange between reader and ST25TB02K**



## 8.7 Read\_block(Addr) command

Command code = 08h

On receiving the Read\_block command, the **ST25TB02K** reads the desired block and returns the 4 data bytes contained in the block. Data bytes are transmitted with the least significant byte first and each byte is transmitted with the least significant bit first.

The address byte gives access to the 64 blocks of the **ST25TB02K** (addresses 0 to 63). Read\_block commands issued with a block address above 63 will not be interpreted and the **ST25TB02K** will not return any response, except for the System area located at address 255.

The **ST25TB02K** must have received a Select() command and be switched to Selected state before any Read\_block() command can be accepted. All Read\_block() commands sent to the **ST25TB02K** before a Select() command is issued are ignored.

**Figure 36. Read\_block request format**

| SOF | Read_block | Address | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|------------|---------|------------------|------------------|-----|
|     | 08h        | 8 bits  | 8 bits           | 8 bits           |     |

Request parameter:

- Address: block addresses from 0 to 63, or 255

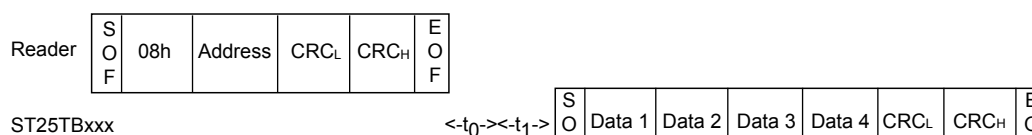
**Figure 37. Read\_block response format**

| SOF | Data 1 | Data 2 | Data 3 | Data 4 | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|--------|--------|--------|--------|------------------|------------------|-----|
|     | 8 bits | 8 bits | 8 bits | 8 bits | 8 bits           | 8 bits           |     |

Response parameters:

- Data 1: Less significant data byte
- Data 2: Data byte
- Data 3: Data byte
- Data 4: Most significant data byte

**Figure 38. Read\_block frame exchange between reader and ST25TB02K**



## 8.8 Write\_block (Addr, Data) command

Command code = 09h

On receiving the Write\_block command, the ST25TB02K writes the 4 bytes contained in the command to the addressed block, provided that the block is available and not write-protected. Data bytes are transmitted with the least significant byte first, and each byte is transmitted with the least significant bit first.

The address byte gives access to the 64 blocks of the ST25TB02K (addresses 0 to 63). Write\_block commands issued with a block address above 63 will not be interpreted and the ST25TB02K will not return any response, except for the System area located at address 255.

The result of the Write\_block command is submitted to the addressed block. See the following tables for a complete description of the Write\_block command:

- Table 4. Resettable OTP area (addresses 0 to 4)
- Table 6. Binary counter (addresses 5 to 6)
- Table 5. EEPROM area (addresses 7 to 63)

The Write\_block command does not give rise to a response from the ST25TB02K. The reader must check after the programming time,  $t_W$ , that the data was correctly programmed. The ST25TB02K must have received a Select() command and be switched to Selected state before any Write\_block command can be accepted. All Write\_block commands sent to the ST25TB02K before a Select() command is issued, are ignored.

**Figure 39. Write\_block request format**

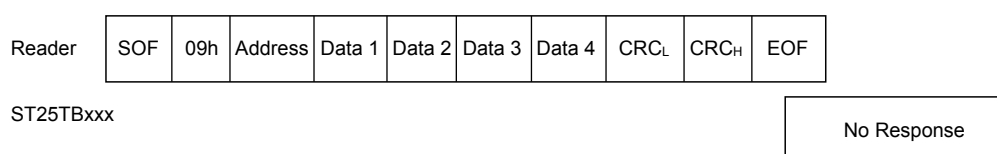
| SOF | Write_block | Address | Data 1 | Data 2 | Data 3 | Data 4 | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|-------------|---------|--------|--------|--------|--------|------------------|------------------|-----|
|     | 09h         | 8 bits  | 8 bits | 8 bits | 8 bits | 8 bits | 8 bits           | 8 bits           |     |

- Request parameters:
- Address: block addresses from 0 to 63, or 255
- Data 1: Less significant data byte
- Data 2: Data byte
- Data 3: Data byte
- Data 4: Most significant data byte.

**Figure 40. Write\_block response format**

|             |
|-------------|
| No Response |
|             |

**Figure 41. Write\_block frame exchange between reader and ST25TB02K**



## 8.9 Get\_UID() command

Command code = 0Bh

On receiving the Get\_UID command, the **ST25TB02K** returns its 8 UID bytes. UID bytes are transmitted with the least significant byte first, and each byte is transmitted with the least significant bit first.

The **ST25TB02K** must have received a Select() command and be switched to Selected state before any Get\_UID() command can be accepted. All Get\_UID() commands sent to the **ST25TB02K** before a Select() command is issued, are ignored.

**Figure 42. Get\_UID request format**

| SOF | Get_IUD | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|---------|------------------|------------------|-----|
|     | 0Bh     | 8 bits           | 8 bits           |     |

Request parameter:

- No parameter

**Figure 43. Get\_UID response format**

| SOF | UID 0  | UID 1  | UID 2  | UID 3  | UID 4  | UID 5  | UID 6  | UID 7  | CRC <sub>L</sub> | CRC <sub>H</sub> | EOF |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|------------------|------------------|-----|
|     | 8 bits | 8 bits | 8 bits | 8 bits | 8 bits | 8 bits | 8 bits | 8 bits | 8 bits           | 8 bits           |     |

Response parameters:

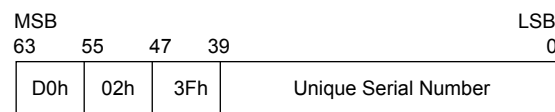
- UID 0: Less significant UID byte
- UID 1 to UID 6: UID bytes
- UID 7: Most significant UID byte.

### Unique identifier (UID)

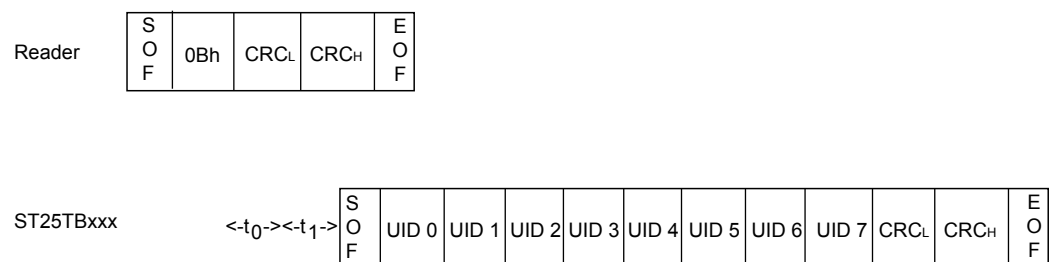
Members of the **ST25TB02K** family are uniquely identified by a 64-bit unique identifier (UID). This is used for addressing each **ST25TB02K** device uniquely after the anticollision loop. The UID complies with ISO/IEC 15963 and ISO/IEC 7816-6. It is a read-only code, and comprises (as summarized in [Figure 44. 64-bit unique identifier of the ST25TB02K](#)):

- an 8-bit prefix, with the most significant bits set to D0h
- an 8-bit IC manufacturer code (ISO/IEC 7816-6/AM1) set to 02h (for STMicroelectronics)
- a 8-bit product ref code set to 3Fh for **ST25TB02K**
- a 40-bit unique serial number

**Figure 44. 64-bit unique identifier of the ST25TB02K**



**Figure 45. Get\_UID frame exchange between reader and ST25TB02K**





## 8.10 Power-on state

After power-on, the **ST25TB02K** is in the following state:

- It is in the low-power state.
- It is in Ready state.
- It shows highest impedance with respect to the reader antenna field.
- It will not respond to any command except Initiate().

## 9 Maximum ratings

Stressing the device above the ratings listed in the absolute maximum ratings table may cause permanent damage to the device. These are stress ratings only and operation of the device at these or any other conditions above those indicated in the operating sections of this specification is not implied. Exposure to absolute maximum ratings conditions for extended periods may affect device reliability. Refer also to the STMicroelectronics SURE Program and other relevant quality documents.

**Table 11. Absolute maximum ratings**

| Symbol                   | Parameter                                                             |                                     | Min. | Max.             | Unit   |
|--------------------------|-----------------------------------------------------------------------|-------------------------------------|------|------------------|--------|
| $T_{STG}$ , $t_{STG}$    | Storage conditions                                                    | Sawn wafer                          | 15   | 25               | °C     |
|                          |                                                                       | (kept in its original packing form) | -    | 9 <sup>(1)</sup> | months |
|                          |                                                                       | Unsawn wafer                        | 19   | 25               | °C     |
|                          |                                                                       | (kept in its antistatic bag)        | -    | 23               | months |
| $I_{CC}$                 | Supply current on AC0 / AC1                                           | -                                   | -    | 40               | mA     |
| $V_{MAX}$ <sup>(2)</sup> | RF input voltage amplitude between AC0 and AC1, GND pad left floating | -                                   | -    | 10               | V      |
| $V_{ESD}$                | Electrostatic discharge voltage                                       | Human Body Model <sup>(3)</sup>     | -    | 2000             | V      |

1. Counted from ST shipment date.

2. Based on characterization, not tested in production.

3. Positive and negative pulses applied on different combinations of pin connections, according to AEC-Q100-002 (compliant with ANSI/ESDA/JEDEC JS-001-2012,  $C1=100$  pF,  $R1=1500$   $\Omega$ ,  $R2=500$   $\Omega$ ).

## 10 RF electrical parameters

**Table 12. Operating conditions**

| Symbol         | Parameter                     | Min. | Max. | Unit |
|----------------|-------------------------------|------|------|------|
| T <sub>A</sub> | Ambient operating temperature | -40  | 85   | °C   |

**Table 13. Electrical characteristics**

| Symbol           | Parameter                                     | Condition                        | Min  | Typ | Max  | Unit |
|------------------|-----------------------------------------------|----------------------------------|------|-----|------|------|
| H_ISO            | Operating field according to ISO              | T <sub>A</sub> = 0 °C to 50 °C   | 1500 | -   | 7500 | mA/m |
| H_extended       | Operating field in extended temperature range | T <sub>A</sub> = -40 °C to 85 °C | 1500 | -   | 7500 |      |
| V <sub>RET</sub> | Back-scattering induced voltage               | ISO 10373-6                      | 20   | -   | -    | mV   |
| C <sub>TUN</sub> | Internal tuning capacitor                     | 13.56 MHz <sup>(1)</sup>         | 62   | 68  | 74   | pF   |

1. The tuning capacitance value is evaluated by characterization with equipment at chip power on reset and ambient temperature. This value is to be used as reference for antenna design. Min and max value are deduced from correlation at ambient temperature with industrial tester limits.

**Note:** For inlay implementation, the antenna design applied for SRI2K can be re-used as-is for **ST25TB02K**: typical 68pF value for the **ST25TB02K** is equivalent to what was specified in the SRI2K data-sheet as 64pF. This change is related to a different measurement methodology between SRI2K and **ST25TB02K**.

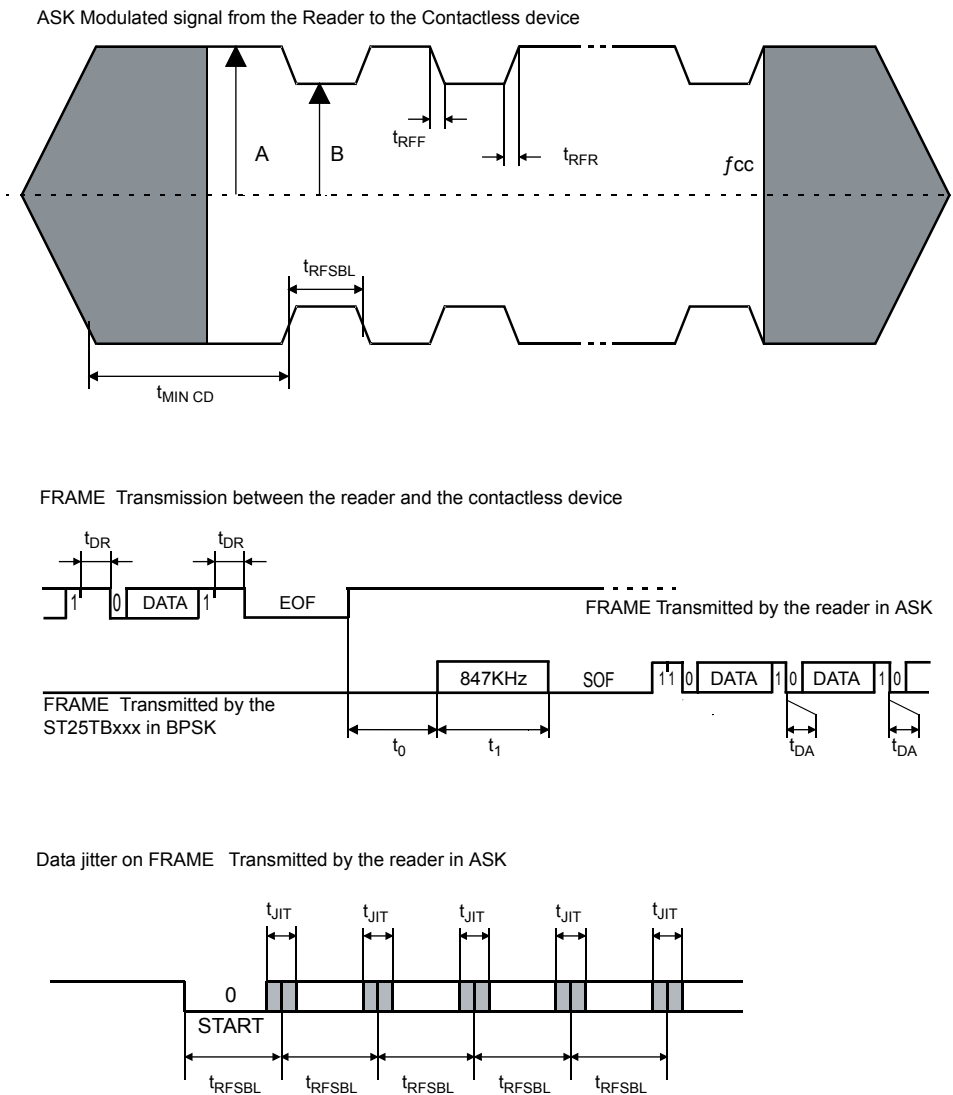
**Table 14. RF characteristics**

| Symbol                              | Parameter                                         | Condition                                      | Min    | Typ   | Max    | Unit |
|-------------------------------------|---------------------------------------------------|------------------------------------------------|--------|-------|--------|------|
| f <sub>CC</sub>                     | RFcarrier frequency                               | -                                              | 13.553 | -     | 13.567 | MHz  |
| MI <sub>CARRIER</sub>               | Carrier modulation index                          | MI=(A-B)/(A+B)                                 | 8      | 11    | 14     | %    |
| t <sub>RFR</sub> , t <sub>RFF</sub> | 10% Rise and Fall times                           | -                                              | 0.1    | -     | 1.25   | µs   |
| t <sub>RFSBL</sub>                  | Minimum pulse width for Start bit                 | ETU = 128/f <sub>CC</sub>                      | -      | 9.44  | -      | µs   |
| t <sub>JIT</sub>                    | ASK modulation data jitter                        | Coupler to <b>ST25TB02K</b>                    | -2     | -     | +2     | µs   |
| t <sub>MIN CD</sub>                 | Minimum timefrom carrier generation to first data | -                                              | 5      | -     | -      | ms   |
| f <sub>S</sub>                      | Subcarrier frequency                              | f <sub>CC</sub> /16                            | -      | 847.5 | -      | kHz  |
| t <sub>0</sub>                      | Antenna reversal delay                            | -                                              | -      | 159   | -      | µs   |
| t <sub>1</sub>                      | Synchronization delay                             | -                                              | -      | 151   | -      | µs   |
| t <sub>2</sub>                      | Answer to new request delay                       | 14 ETU                                         | 132    | -     | -      | µs   |
| t <sub>DR</sub>                     | Time between request characters                   | Coupler to <b>ST25TB02K</b>                    | 0      | -     | 57     | µs   |
| t <sub>DA</sub>                     | Time between answer characters                    | <b>ST25TB02K</b> to coupler                    | -      | 0     | -      | µs   |
| t <sub>W</sub>                      | Programming time for write                        | With no auto-erase cycle (OTP)                 | -      | -     | 3      | ms   |
|                                     |                                                   | With auto-erase cycle (EEPROM)                 | -      | -     | 5      | ms   |
|                                     |                                                   | Binarycounter decrement with tearing condition | -      | -     | 7      | ms   |

Note: All timing measurements were performed on a reference antenna with the following characteristics:

- External size: 76 mm x 46 mm
- Number of turns: 4
- Width of conductor: 0.9 mm
- Space between 2 conductors: 0.9 mm
- Tuning Frequency: 13.58 MHz

**Figure 46. ST25TB02K synchronous timing, transmit and receive**



## 11 Ordering information

**Table 15. Ordering information scheme**

|                          |      |   |   |  |     |    |   |   |   |   |
|--------------------------|------|---|---|--|-----|----|---|---|---|---|
| Example:                 | ST25 | T | B |  | 02K | -A | C | 6 | G | 6 |
| <b>Device type</b>       |      |   |   |  |     |    |   |   |   |   |
| ST25 = RF memory         |      |   |   |  |     |    |   |   |   |   |
| <b>Product type</b>      |      |   |   |  |     |    |   |   |   |   |
| T = Tags + RFID          |      |   |   |  |     |    |   |   |   |   |
| <b>Protocol</b>          |      |   |   |  |     |    |   |   |   |   |
| B = ISO14443-B           |      |   |   |  |     |    |   |   |   |   |
| <b>Memory density</b>    |      |   |   |  |     |    |   |   |   |   |
| 02K (binary)             |      |   |   |  |     |    |   |   |   |   |
| <b>Interface</b>         |      |   |   |  |     |    |   |   |   |   |
| A = None                 |      |   |   |  |     |    |   |   |   |   |
| <b>Features</b>          |      |   |   |  |     |    |   |   |   |   |
| C = Counter as option    |      |   |   |  |     |    |   |   |   |   |
| <b>Device grade</b>      |      |   |   |  |     |    |   |   |   |   |
| 6 = - 40 to 85 °C        |      |   |   |  |     |    |   |   |   |   |
| <b>Package/Packaging</b> |      |   |   |  |     |    |   |   |   |   |
| G = Bumped 120 µm        |      |   |   |  |     |    |   |   |   |   |
| U = Unseen 725 µm        |      |   |   |  |     |    |   |   |   |   |
| <b>Capacitor value</b>   |      |   |   |  |     |    |   |   |   |   |
| 6 = 68 pF                |      |   |   |  |     |    |   |   |   |   |

**Note:** *Devices are shipped from the factory with the memory content bits erased to 1.*

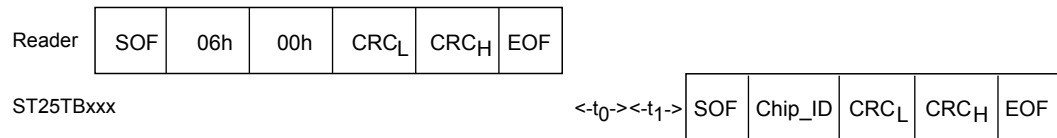
Parts marked as "ES", "E" or accompanied by an Engineering Sample notification letter, are not yet qualified and therefore not yet ready to be used in production and any consequences deriving from such usage will not be at ST charge. In no event, ST will be liable for any customer usage of these engineering samples in production. ST Quality has to be contacted prior to any decision to use these Engineering samples to run qualification activity.

## Appendix A ISO-14443 Type B CRC calculation

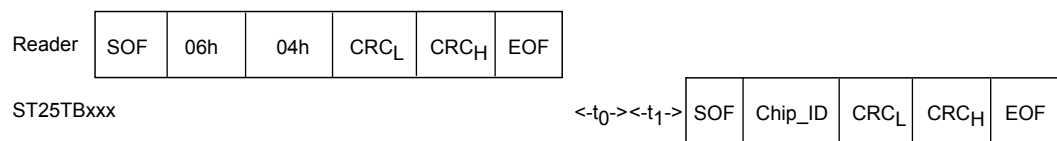
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#define BYTE unsigned char
#define USHORT unsigned short
unsigned short UpdateCrc(BYTE ch, USHORT *lpwCrc)
{
  ch = (ch^(BYTE)((*lpwCrc) & 0x00FF));
  ch = (ch^(ch<<4));
  *lpwCrc = (*lpwCrc >> 8)^((USHORT)ch <<
  8)^((USHORT)ch<<3)^((USHORT)ch>>4);
  return(*lpwCrc);
}
void ComputeCrc(char *Data, int Length, BYTE *TransmitFirst, BYTE
*TransmitSecond)
{
  BYTE chBlock; USHORT wCrc;
  wCrc = 0xFFFF; // ISO 3309
  do
  {
    chBlock = *Data++;
    UpdateCrc(chBlock, &wCrc);
  } while (--Length);
  wCrc = ~wCrc; // ISO 3309
  *TransmitFirst = (BYTE) (wCrc & 0xFF);
  *TransmitSecond = (BYTE) ((wCrc >> 8) & 0xFF);
  return;
}
int main(void)
{
  BYTE BuffCRC_B[10] = {0x0A, 0x12, 0x34, 0x56}, First, Second, i;
  printf("Crc-16 G(x) = x^16 + x^12 + x^5 + 1");
  printf("CRC_B of [ ");
  for(i=0; i<4; i++)
    printf("%02X ", BuffCRC_B[i]);
  ComputeCrc(BuffCRC_B, 4, &First, &Second);
  printf("] Transmitted: %02X then %02X.", First, Second);
  return(0);
}
```

## Appendix B ST25TB02K command brief

**Figure 47. Initiate frame exchange between reader and ST25TB02K**



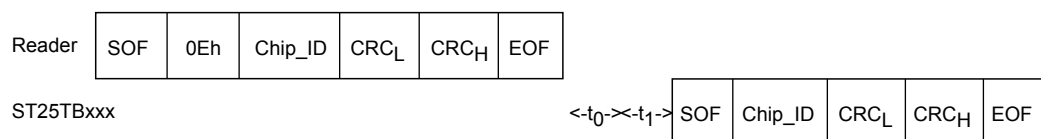
**Figure 48. Pcall16 frame exchange between reader and ST25TB02K**



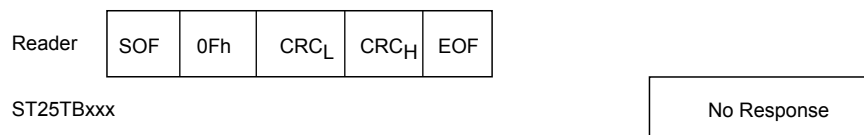
**Figure 49. Slot\_marker frame exchange between reader and ST25TB02K**

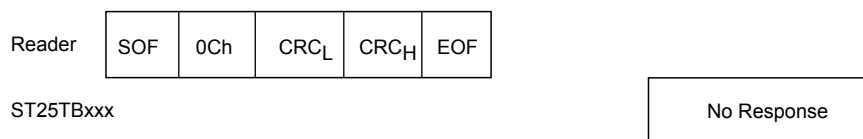
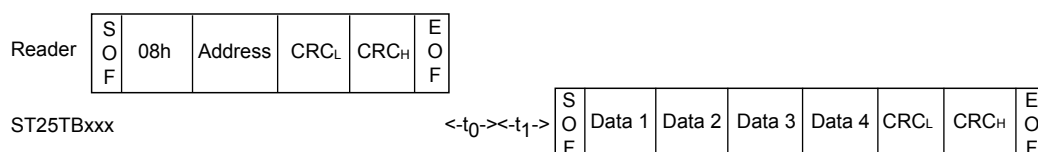
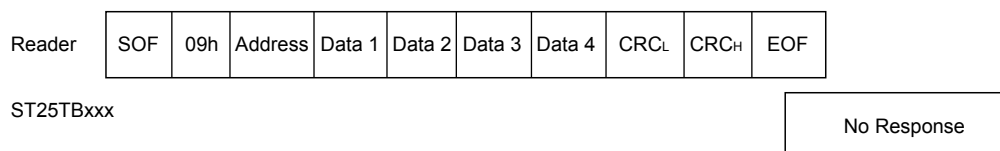
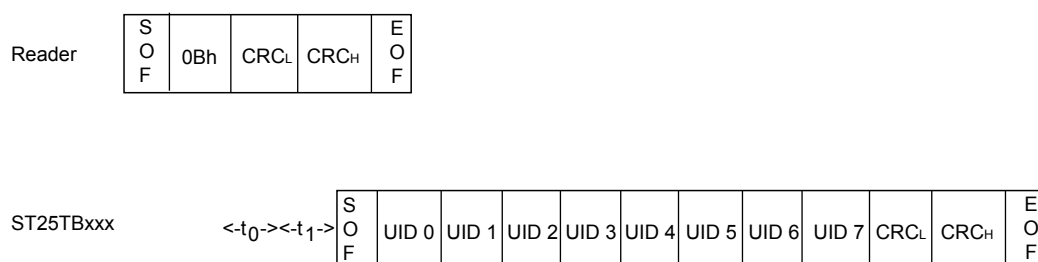


**Figure 50. Select frame exchange between reader and ST25TB02K**



**Figure 51. Completion frame exchange between reader and ST25TB02K**



**Figure 52. Reset\_to\_inventory frame exchange between reader and ST25TB02K**

**Figure 53. Read\_block frame exchange between reader and ST25TB02K**

**Figure 54. Write\_block frame exchange between reader and ST25TB02K**

**Figure 55. Get\_UID frame exchange between reader and ST25TB02K**




## Revision history

**Table 16. Document revision history**

| Date        | Version | Changes                                                                                                                                                                                                                                                                                                                                            |
|-------------|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 01-Feb-2016 | 1       | Initial release                                                                                                                                                                                                                                                                                                                                    |
| 10-Mar-2016 | 2       | Updated Figure 28 and Figure 41.                                                                                                                                                                                                                                                                                                                   |
| 18-Apr-2016 | 3       | Updated Table 3: ST25TB02K memory mapping.<br>Changed confidentiality level from ST restricted to public.                                                                                                                                                                                                                                          |
| 21-Sep-2016 | 4       | Updated: <ul style="list-style-type: none"> <li>Figure 44: 64-bit unique identifier of the ST25TB02K</li> <li>Figure 46: ST25TB02K synchronous timing, transmit and receive</li> <li>Table 11. Absolute maximum ratings,</li> <li>Table 15. Ordering information scheme (bumped and sawn wafer)</li> <li>Section 8.9: Get_UID() command</li> </ul> |
| 18-Oct-2016 | 5       | Updated Features in cover page                                                                                                                                                                                                                                                                                                                     |
| 10-Oct-2018 | 6       | Updated <a href="#">Section 4.1 EEPROM area</a>                                                                                                                                                                                                                                                                                                    |
| 07-Feb-2023 | 7       | Updated note 1 in <a href="#">Table 13. Electrical characteristics</a>                                                                                                                                                                                                                                                                             |

## Contents

|          |                                                              |           |
|----------|--------------------------------------------------------------|-----------|
| <b>1</b> | <b>Description</b>                                           | <b>2</b>  |
| <b>2</b> | <b>Signal description</b>                                    | <b>4</b>  |
| 2.1      | AC1, AC0                                                     | 4         |
| <b>3</b> | <b>Data transfer</b>                                         | <b>5</b>  |
| 3.1      | Input data transfer from reader to ST25TB02K (request frame) | 5         |
| 3.1.1    | Character transmission format for request frame              | 5         |
| 3.1.2    | Request start of frame                                       | 5         |
| 3.1.3    | Request end of frame                                         | 6         |
| 3.2      | Output data transfer from to reader ST25TB02K (answer frame) | 7         |
| 3.2.1    | Character transmission format for answer frame               | 7         |
| 3.2.2    | Answer start of frame                                        | 7         |
| 3.2.3    | Answer end of frame                                          | 8         |
| 3.3      | Trasmission frame                                            | 8         |
| 3.4      | CRC                                                          | 9         |
| <b>4</b> | <b>Memory mapping</b>                                        | <b>10</b> |
| 4.1      | EEPROM area                                                  | 11        |
| 4.1.1    | Block 0-4: resettable OTP area                               | 11        |
| 4.1.2    | Block 7-63                                                   | 12        |
| 4.2      | 32-bit binary counters                                       | 13        |
| 4.3      | System area                                                  | 14        |
| 4.3.1    | OTP_Lock_Reg                                                 | 14        |
| <b>5</b> | <b>ST25TB02K operation</b>                                   | <b>15</b> |
| <b>6</b> | <b>ST25TB02K states</b>                                      | <b>16</b> |
| 6.1      | Power-off state                                              | 16        |
| 6.2      | Ready state                                                  | 16        |
| 6.3      | Inventory state                                              | 16        |
| 6.4      | Selected state                                               | 16        |
| 6.5      | Deselected state                                             | 16        |
| 6.6      | Deactivated state                                            | 17        |
| <b>7</b> | <b>Anticollision</b>                                         | <b>18</b> |
| 7.1      | Description of an anticollision sequence                     | 19        |
| <b>8</b> | <b>ST25TB02K commands</b>                                    | <b>22</b> |
| 8.1      | Initiate() command                                           | 23        |
| 8.2      | Pcall16() command                                            | 24        |

|                   |                                               |           |
|-------------------|-----------------------------------------------|-----------|
| 8.3               | Slot_marker(SN) command .....                 | 25        |
| 8.4               | Select(Chip_ID) command .....                 | 26        |
| 8.5               | Completion() command .....                    | 27        |
| 8.6               | Reset_to_inventory() command .....            | 28        |
| 8.7               | Read_block(Addr) command .....                | 29        |
| 8.8               | Write_block (Addr, Data) command .....        | 30        |
| 8.9               | Get_UID() command .....                       | 31        |
| 8.10              | Power-on state .....                          | 33        |
| <b>9</b>          | <b>Maximum ratings .....</b>                  | <b>34</b> |
| <b>10</b>         | <b>RF electrical parameters .....</b>         | <b>35</b> |
| <b>11</b>         | <b>Ordering information .....</b>             | <b>37</b> |
| <b>Appendix A</b> | <b>ISO-14443 Type B CRC calculation .....</b> | <b>38</b> |
| <b>Appendix B</b> | <b>ST25TB02K command brief .....</b>          | <b>39</b> |
|                   | <b>Revision history .....</b>                 | <b>41</b> |

## List of tables

|                  |                                                  |    |
|------------------|--------------------------------------------------|----|
| <b>Table 1.</b>  | Signal names . . . . .                           | 2  |
| <b>Table 2.</b>  | Bit description . . . . .                        | 5  |
| <b>Table 3.</b>  | ST25TB02K memory mapping. . . . .                | 10 |
| <b>Table 4.</b>  | Resettable OTP area (addresses 0 to 4) . . . . . | 11 |
| <b>Table 5.</b>  | EEPROM area (addresses 7 to 63) . . . . .        | 12 |
| <b>Table 6.</b>  | Binary counter (addresses 5 to 6). . . . .       | 13 |
| <b>Table 7.</b>  | System area . . . . .                            | 14 |
| <b>Table 8.</b>  | Standard anticollision sequence. . . . .         | 20 |
| <b>Table 9.</b>  | Example of an anticollision sequence . . . . .   | 20 |
| <b>Table 10.</b> | Command code . . . . .                           | 22 |
| <b>Table 11.</b> | Absolute maximum ratings . . . . .               | 34 |
| <b>Table 12.</b> | Operating conditions . . . . .                   | 35 |
| <b>Table 13.</b> | Electrical characteristics . . . . .             | 35 |
| <b>Table 14.</b> | RF characteristics . . . . .                     | 35 |
| <b>Table 15.</b> | Ordering information scheme. . . . .             | 37 |
| <b>Table 16.</b> | Document revision history . . . . .              | 41 |

## List of figures

|            |                                                                          |    |
|------------|--------------------------------------------------------------------------|----|
| Figure 1.  | Logic diagram. . . . .                                                   | 2  |
| Figure 2.  | Die floor plan and assembly options . . . . .                            | 3  |
| Figure 3.  | ST25TB02K 10% ASK modulation of the received wave . . . . .              | 5  |
| Figure 4.  | ST25TB02K request frame character format. . . . .                        | 5  |
| Figure 5.  | Request start of frame . . . . .                                         | 6  |
| Figure 6.  | Request end of frame . . . . .                                           | 6  |
| Figure 7.  | Wave transmitted using BPSK subcarrier modulation. . . . .               | 7  |
| Figure 8.  | Answer start of frame . . . . .                                          | 7  |
| Figure 9.  | Answer end of frame . . . . .                                            | 8  |
| Figure 10. | Example of a complete transmission frame. . . . .                        | 8  |
| Figure 11. | CRC transmission rules . . . . .                                         | 9  |
| Figure 12. | Write_block update in Standard mode (binary format) . . . . .            | 11 |
| Figure 13. | Write_block update in Reload mode (binary format). . . . .               | 11 |
| Figure 14. | Countdown example (binary format) . . . . .                              | 13 |
| Figure 15. | State transition diagram . . . . .                                       | 17 |
| Figure 16. | ST25TB02K Chip_ID description. . . . .                                   | 18 |
| Figure 17. | Description of a possible anticollision sequence . . . . .               | 19 |
| Figure 18. | Initiate request format . . . . .                                        | 23 |
| Figure 19. | Initiate response format . . . . .                                       | 23 |
| Figure 20. | Initiate frame exchange between reader and ST25TB02K . . . . .           | 23 |
| Figure 21. | Pcall16 request format. . . . .                                          | 24 |
| Figure 22. | Pcall16 response format . . . . .                                        | 24 |
| Figure 23. | Pcall16 frame exchange between reader and ST25TB02K . . . . .            | 24 |
| Figure 24. | Slot_marker request format . . . . .                                     | 25 |
| Figure 25. | Slot_marker response format . . . . .                                    | 25 |
| Figure 26. | Slot_marker frame exchange between reader and ST25TB02K . . . . .        | 25 |
| Figure 27. | Select request format . . . . .                                          | 26 |
| Figure 28. | Select response format . . . . .                                         | 26 |
| Figure 29. | Select frame exchange between reader and ST25TB02K. . . . .              | 26 |
| Figure 30. | Completion request format. . . . .                                       | 27 |
| Figure 31. | Completion response format. . . . .                                      | 27 |
| Figure 32. | Completion frame exchange between reader and ST25TB02K . . . . .         | 27 |
| Figure 33. | Reset_to_inventory request format . . . . .                              | 28 |
| Figure 34. | Reset_to_inventory response format . . . . .                             | 28 |
| Figure 35. | Reset_to_inventory frame exchange between reader and ST25TB02K . . . . . | 28 |
| Figure 36. | Read_block request format . . . . .                                      | 29 |
| Figure 37. | Read_block response format . . . . .                                     | 29 |
| Figure 38. | Read_block frame exchange between reader and ST25TB02K. . . . .          | 29 |
| Figure 39. | Write_block request format. . . . .                                      | 30 |
| Figure 40. | Write_block response format . . . . .                                    | 30 |
| Figure 41. | Write_block frame exchange between reader and ST25TB02K. . . . .         | 30 |
| Figure 42. | Get_UID request format. . . . .                                          | 31 |
| Figure 43. | Get_UID response format . . . . .                                        | 31 |
| Figure 44. | 64-bit unique identifier of the ST25TB02K . . . . .                      | 32 |
| Figure 45. | Get_UID frame exchange between reader and ST25TB02K . . . . .            | 32 |
| Figure 46. | ST25TB02K synchronous timing, transmit and receive. . . . .              | 36 |
| Figure 47. | Initiate frame exchange between reader and ST25TB02K . . . . .           | 39 |
| Figure 48. | Pcall16 frame exchange between reader and ST25TB02K . . . . .            | 39 |
| Figure 49. | Slot_marker frame exchange between reader and ST25TB02K . . . . .        | 39 |
| Figure 50. | Select frame exchange between reader and ST25TB02K. . . . .              | 39 |
| Figure 51. | Completion frame exchange between reader and ST25TB02K . . . . .         | 39 |
| Figure 52. | Reset_to_inventory frame exchange between reader and ST25TB02K . . . . . | 40 |
| Figure 53. | Read_block frame exchange between reader and ST25TB02K. . . . .          | 40 |

---

|                   |                                                                   |    |
|-------------------|-------------------------------------------------------------------|----|
| <b>Figure 54.</b> | Write_block frame exchange between reader and ST25TB02K . . . . . | 40 |
| <b>Figure 55.</b> | Get_UID frame exchange between reader and ST25TB02K . . . . .     | 40 |

**IMPORTANT NOTICE – READ CAREFULLY**

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgment.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to [www.st.com/trademarks](http://www.st.com/trademarks). All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2023 STMicroelectronics – All rights reserved