Safety Guidelines

This manual contains notices which you should observe to ensure your own personal safety, as well as to protect the product and connected equipment. These notices are highlighted in the manual by a warning triangle and are marked as follows according to the level of danger:

Danger
indicates that death, severe personal injury or substantial property damage will result if proper precautions are not taken.

Warning
indicates that death, severe personal injury or substantial property damage can result if proper precautions are not taken.

Caution
indicates that minor personal injury or property damage can result if proper precautions are not taken.

Note
draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.
# Table of contents

1 Introduction ................................................................................................................. 3  
1.1 Abbreviations ........................................................................................................ 4  
2 Host Software Modules................................................................................................. 5  
2.1 Introduction ............................................................................................................ 5  
2.2 Using the Host Software Modules ........................................................................... 6  
3 Communication Read/Write Device-Host ..................................................................... 7  
3.1 Introduction ............................................................................................................. 7  
3.2 Ordinary Protocol .................................................................................................. 8  
3.3 Extended Protocol ................................................................................................ 8  
3.4 Transfer Timeout Intervals ..................................................................................... 9  
3.5 Command Set ........................................................................................................ 10  
3.6 Status Byte ........................................................................................................... 12  
3.7 Command Description for Operating Mode ........................................................... 13  
3.7.1 GetSnr .............................................................................................................. 13  
3.7.2 GetSnr_Adv ..................................................................................................... 14  
3.7.3 SelectSnr ......................................................................................................... 15  
3.7.4 SelectLast ........................................................................................................ 15  
3.7.5 HaltSelected ................................................................................................... 16  
3.7.6 ReadPage ......................................................................................................... 17  
3.7.7 ReadBlock ....................................................................................................... 18  
3.7.8 WritePage ....................................................................................................... 19  
3.7.9 WriteBlock ...................................................................................................... 20  
3.7.10 TagAuthent ................................................................................................... 21  
3.7.11 MutualAuthent .............................................................................................. 22  
3.7.12 GetSnr_LT .................................................................................................... 23  
3.7.13 GetSnrReset_LT .......................................................................................... 24  
3.7.14 HaltSelected_LT ......................................................................................... 25  
3.7.15 ReadPage_LT .............................................................................................. 25  
3.7.16 ReadPageInv_LT ......................................................................................... 26  
3.7.17 WritePage_LT .............................................................................................. 27  
3.7.18 ReadPublicB_LT ......................................................................................... 28  
3.7.19 Read_F1 ...................................................................................................... 29  
3.7.20 PollTags ....................................................................................................... 30  
3.7.21 PollKbTags ................................................................................................... 32  
3.7.22 GetVersion .................................................................................................... 33  
3.7.23 Reset ............................................................................................................ 34  
3.7.24 HFReset ....................................................................................................... 34  
3.7.25 StopCommand .............................................................................................. 35  
3.7.26 ReadInput ...................................................................................................... 36  
3.7.27 SetOutput ..................................................................................................... 37  
3.7.28 ConfigPorts ................................................................................................. 38  
3.7.29 ReadPorts .................................................................................................... 39  
3.7.30 WritePorts ................................................................................................... 40  
3.7.31 ReadEEData ................................................................................................. 41  
3.7.32 WriteEEData ............................................................................................... 41  
3.7.33 SetProxTrmTime ......................................................................................... 42  
3.7.34 SetModuleAdr ............................................................................................. 43  
3.7.35 SetHFMode .................................................................................................. 44  
3.7.36 StartFFT ...................................................................................................... 45  
3.7.37 ReadBCD .................................................................................................... 46  
3.7.38 SetBCD ....................................................................................................... 47  
3.7.39 SetBCDOffset ............................................................................................. 48  
3.7.40 ReadLRStatus ............................................................................................. 48
## Table of contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.7.41 SetPowerDown</td>
<td>49</td>
</tr>
<tr>
<td>3.7.42 ReadAllPage</td>
<td>50</td>
</tr>
<tr>
<td>3.7.43 GetDSP_version</td>
<td>51</td>
</tr>
<tr>
<td>3.7.44 KeyInitMode</td>
<td>52</td>
</tr>
<tr>
<td>3.8 Command Description for KeyInit Mode</td>
<td>53</td>
</tr>
<tr>
<td>3.8.1 Reset</td>
<td>53</td>
</tr>
<tr>
<td>3.8.2 WriteSerNum</td>
<td>54</td>
</tr>
<tr>
<td>3.8.3 ReadEEPROM</td>
<td>55</td>
</tr>
<tr>
<td>3.8.4 WriteEEPROM</td>
<td>56</td>
</tr>
<tr>
<td>3.8.5 ReadControl</td>
<td>57</td>
</tr>
<tr>
<td>3.8.6 WriteControl</td>
<td>58</td>
</tr>
<tr>
<td>3.8.7 ReadSecret_LT</td>
<td>59</td>
</tr>
<tr>
<td>3.8.8 WriteSecret_LT</td>
<td>60</td>
</tr>
<tr>
<td>3.8.9 ReadControl_LT</td>
<td>61</td>
</tr>
<tr>
<td>3.8.10 WriteControl_LT</td>
<td>62</td>
</tr>
<tr>
<td>3.9 Examples to Access MDS F4xx Transponders</td>
<td>63</td>
</tr>
<tr>
<td>3.9.1 SIM: Anticollision Cycle</td>
<td>63</td>
</tr>
<tr>
<td>3.9.2 Proximity/SIM: Read Plain</td>
<td>64</td>
</tr>
<tr>
<td>3.9.3 Proximity/SIM: Write Plain</td>
<td>64</td>
</tr>
<tr>
<td>3.9.4 Proximity/SIM: Read Crypto</td>
<td>64</td>
</tr>
<tr>
<td>3.9.5 Proximity/SIM: Write Crypto</td>
<td>64</td>
</tr>
<tr>
<td>3.10 Examples to Access MDS F2xx Transponders</td>
<td>65</td>
</tr>
<tr>
<td>3.10.1 Proximity/SIM: Read</td>
<td>65</td>
</tr>
<tr>
<td>3.10.2 Proximity/SIM: Write</td>
<td>65</td>
</tr>
<tr>
<td>4 Transponders</td>
<td>66</td>
</tr>
<tr>
<td>4.1 MDS F4xx Transponders</td>
<td>66</td>
</tr>
<tr>
<td>4.1.1 Memory Organization</td>
<td>66</td>
</tr>
<tr>
<td>4.1.2 Anticollision</td>
<td>67</td>
</tr>
<tr>
<td>4.1.3 Operation-Modes and Configuration</td>
<td>68</td>
</tr>
<tr>
<td>4.1.4 Configuration of Delivered MDS F4xx Transponders</td>
<td>71</td>
</tr>
<tr>
<td>4.2 MDS F2xx Transponders</td>
<td>72</td>
</tr>
<tr>
<td>4.2.1 Memory Organization</td>
<td>72</td>
</tr>
<tr>
<td>4.2.2 Operation-Modes and Configuration</td>
<td>73</td>
</tr>
<tr>
<td>4.2.3 Configuration of Delivered MDS F2xx Transponders</td>
<td>75</td>
</tr>
<tr>
<td>4.3 MDS F1xx Transponder</td>
<td>75</td>
</tr>
<tr>
<td>5 Personalization</td>
<td>76</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>76</td>
</tr>
<tr>
<td>5.2 Personalization Concept</td>
<td>76</td>
</tr>
<tr>
<td>5.3 Personalization of MDS F4xx Transponders</td>
<td>81</td>
</tr>
<tr>
<td>5.3.1 Definition of Keys and Logdata</td>
<td>81</td>
</tr>
<tr>
<td>5.3.2 Changing Keys and Logdata</td>
<td>82</td>
</tr>
<tr>
<td>5.4 Personalization of MDS F2xx Transponders</td>
<td>84</td>
</tr>
<tr>
<td>5.4.1 Definition of Passwords and Keys</td>
<td>84</td>
</tr>
<tr>
<td>5.4.2 Changing Passwords and Keys</td>
<td>85</td>
</tr>
<tr>
<td>6 Security Considerations</td>
<td>87</td>
</tr>
<tr>
<td>6.1 Data Reliability</td>
<td>87</td>
</tr>
<tr>
<td>6.1.1 Data Stream between Read/Write Device and Transponder</td>
<td>87</td>
</tr>
<tr>
<td>6.1.2 Checking User Data</td>
<td>87</td>
</tr>
<tr>
<td>6.2 Data Privacy</td>
<td>88</td>
</tr>
</tbody>
</table>
1 Introduction

This description refers to the interface between a host (e.g. PC) and a contactless 125 kHz read/write device (SIM).

For easy and quick development of application specific host software Siemens provides a C-Library, Source- and Header-Files. These tools can be found on the floppy disk “C-Library MFDAPI” (Order no. 6GT2 481-0AA00).

Following transponders of the 125 kHz family are supported:

- MDS F4xx
- MDS F2xx
- MDS F1xx

Additional Features:

- High security by using cryptography, mutual authentication and password verification
- Programmable port pins: 1 output; 1 input
- 85 bytes of user-defined data can be stored in an EEPROM of the read/write device (SIM)

System Requirements to use Siemens C-Library PROLIB6:

- IBM-PC or compatible (minimum 286 processor) with available serial interface
- Borland C-Compiler (Version 3.1 recommended)

Life support applications

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Siemens customers using or selling these products for use in such applications do so on their own risk and agree to fully indemnify Siemens for any damages resulting from such improper use or sale.
## 1.1 Abbreviations

Please find in the following a list of the abbreviations used in this document.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addr</td>
<td>Address</td>
</tr>
<tr>
<td>BCC</td>
<td>Block Check Character</td>
</tr>
<tr>
<td>BYTE_T</td>
<td>Byte (unsigned character)</td>
</tr>
<tr>
<td>char</td>
<td>Character</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic Redundancy Check</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processor</td>
</tr>
<tr>
<td>DWORD_T</td>
<td>Double Word (unsigned)</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transformation</td>
</tr>
<tr>
<td>HF</td>
<td>High Frequency</td>
</tr>
<tr>
<td>LSB</td>
<td>Least Significant Byte</td>
</tr>
<tr>
<td>MDS</td>
<td>Mobile Data Memory</td>
</tr>
<tr>
<td>MSB</td>
<td>Most Significant Byte</td>
</tr>
<tr>
<td>nmb</td>
<td>Number</td>
</tr>
<tr>
<td>OTP</td>
<td>One Time Programmable</td>
</tr>
<tr>
<td>pagenr</td>
<td>Page Number</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>ro</td>
<td>Read Only</td>
</tr>
<tr>
<td>r/w</td>
<td>Read/Write</td>
</tr>
<tr>
<td>RWD</td>
<td>Read/Write Device, SIM</td>
</tr>
<tr>
<td>SIM</td>
<td>Serial Interface Module, Read/Write Device</td>
</tr>
<tr>
<td>snr</td>
<td>Serial Number</td>
</tr>
<tr>
<td>TAG (tag)</td>
<td>Transponder, MDS</td>
</tr>
<tr>
<td>wo</td>
<td>Write Only</td>
</tr>
</tbody>
</table>
2 Host Software Modules

2.1 Introduction

On the Floppy Disk "C-Library MFDAPI" you will find the following tools.

**Library Files:**

| PROLIB6C.LIB  | Compact Memory Model |
| PROLIB6H.LIB  | Huge Memory Model    |
| PROLIB6L.LIB  | Large Memory Model   |
| PROLIB6M.LIB  | Medium Memory Model  |
| PROLIB6S.LIB  | Small Memory Model   |
| PROLIB6T.LIB  | Tiny Memory Model    |

Depending on the Memory Model you choose for host software you have to include the corresponding Library File in your project. These Libraries are helpful for developing DOS software in Standard C language with a Borland C-Compiler.

**Header-File:**

| PROLIB6.H    | MDS F4xx / MDS F2xx / RWD function declarations |
| PROBLT6.H    | MDS F2xx function declarations                  |
| PROLB6MU6.H  | Multiple RWD function declarations (Network)    |
| PROLVEG6.H   | Function declarations for a special project requiring a special Reader-Hardware and -Software |

Each Header File provides function declarations with detailed information about the use of commands. Depending on the used command function (e.g. proloc_GetSnr) you have to include the corresponding Header File(s) in your application specific source file.

**Source Files:**

| PROLIB6.C    | MDS F4xx / MDS F2xx / RWD functions               |
| PROBLT6.C    | MDS F2xx functions                                |
| PROLVEG6.C   | Functions for a special project requiring a special Reader-Hardware and -Software |

The code in the Source Files is identical with the code included in the Library Files. There is no Source File for using multiple RWD's because all necessary code is located in the Header File PROLB6MU6.H.

For developing software on other platforms by using other compilers we recommend to use our Source Files and Header Files and make your specific adaptations.
2.2 Using the Host Software Modules

Communication via the serial interface between the host and the read/write device is handled by using interrupts. As a consequence a host-program only has to test a flag (RWDEot) periodically to recognize the end of a communication sequence. Meanwhile the program can execute other functions while the data transfer is running in the background.

To do this, however, some flags are needed:

**RWDEot:** Helps to identify the end of a communication sequence
- flag is set to 0 at a library function request
- flag is set to 1 at the end of the serial protocol

**RWDErr:** Saves the error code
- 0 = errorfree execution
- <0 = error has occured

**RWDDataLen:** Saves the number of bytes received via the serial interface.
- Can take any value between 0 and 24.

- To open the serial port on the host system use function (Header File PROLIB6.H)
  ```c
  proloc_open(char *ComStr) // *ComStr = "COM1" for COM1
  // "ComStr = "COM2" for COM2
  ```

- To close the serial port on the host system use function (Header File PROLIB6.H)
  ```c
  proloc_close()
  ```
  In order to prevent undesired side effects use proloc_close before program end or before a new proloc_open.

- To change BCC calculation (when entering or leaving KeyInitMode) use function (Header File PROLIB6.H)
  ```c
  proloc_SetBCCMode(BYTE_T mode) // mode=0x00: Operating Mode
  // mode=0x01: KeyInit-Mode
  ```

- For communication in Extended Protocol use commands with ‘Proloc_M’-prefix (Header File PROLBMU6.h)

The names of the commands described in the following have to be prefixed with „proloc_“ to get the corresponding names in the C-Library (e.g. function proloc_GetSnr() for command GetSnr) for Ordinary Protocol.

All Header Files contain short examples to illustrate the usage of each command.
3 Communication Read/Write Device-Host

3.1 Introduction

The host (e.g. PC) communicates with the contactless 125 kHz read/write device via a serial interface using a baud rate of 9600 baud.

Data transfer details are: 1 start bit, 8 data bits, 1 stop bit and no parity bit, the Least Significant Bit is sent first.

Each communication sequence consists of a block of bytes sent by the host, and a block of bytes answered by the reader.

All bytes are transmitted transparently, i.e. you can use any character between 0x00 and 0xFF.

Block Length:

Block Length is the sum of all transferred bytes including Block Length but excluding BCC.

Block Title:

The Command Byte if sent from host to reader.
The Status Byte if sent from reader to host.

Data:

Data bytes are only transmitted if data is transferred.

BCC:

The BCC (Block Check Character) is calculated by bytes 1 to n-1 (n=number of bytes of the whole communication sequence).

A different BCC calculation in Operating Mode (mode of the reader for using standard commands) and in KeyInit Mode (mode of the reader device for using personalization commands) helps to avoid the overwriting of secret data accidentally.

BCC calculation in Operating Mode of the reader:

The BCC is computed by EXOR-operation of all block data bytes including Block Length.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>EXOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Example for command GetSnr:

| Byte 1: Block Length | 0000 0010 0x02 |
| Byte 2: Command Byte | 0100 0111 0x47 |
| Byte 3: BCC          | 0100 0101 0x45 |

BCC calculation in KeyInit Mode of the reader:

The BCC is computed by adding all block data bytes including Block Length. The least significant eight bits are used as BCC.
3.2 Ordinary Protocol

If only a single read/write device with a node address equal to zero is connected to the host (e.g. on a RS232 serial line) the Ordinary Protocol is used to address this reader.

Format of the Ordinary Protocol (Host→Reader and Reader→Host):

<table>
<thead>
<tr>
<th>Byte</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>......</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Block Length</td>
<td>Block Title</td>
<td>data</td>
<td>data</td>
<td>......</td>
<td>BCC</td>
</tr>
</tbody>
</table>

3.3 Extended Protocol

If more than one read/write devices with node addresses different from zero are connected to the host (e.g. on a RS485 serial line) the Extended Protocol is used to address a single reader.

Format of the Extended Protocol (Host→Reader and Reader→Host):

<table>
<thead>
<tr>
<th>Byte</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>......</th>
<th>n-1</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Block Length + 0x80</td>
<td>Block Title</td>
<td>data</td>
<td>data</td>
<td>......</td>
<td>Node Address</td>
<td>BCC</td>
</tr>
</tbody>
</table>

Differences to Ordinary Protocol: Bit 7 of Block Length is set, and the Node Address is inserted just before BCC.

If a reader’s node address is different from zero, the reader enters net-mode. In this mode the reader expects all commands from the host to be sent in Extended Protocol including the right Node Address (except SetModuleAdr). If the host transmits a string that does not meet these conditions, the command is ignored, and there will be no answer from the reader (whereas a reader being not in net-mode - with node address equal to zero - would at least answer with a SERIAL ERROR message).

The command SetModuleAdr is used to assign a unique node address to a device whose serial number is known. This command should be sent in Ordinary Protocol. If the right serial number was sent, there will be an answer from the read/write device. This answer is sent in Ordinary Protocol if the former node address of the reader was zero, otherwise the answer is sent in Extended Protocol.

For communication in Extended Protocol use commands with ‘Proloc_M’-prefix. For further information see Header File PROLBMU6.h.
3.4 Transfer Timeout Intervals

Character Delay:
Character Delay is the maximum time permitted to elapse between sending two consecutive characters of a block.

Character Delay $\leq 150$ ms

Block Delay:
Block Delay is only necessary if an error has occurred in the serial communication. To allow for re-synchronization in that case of malfunction there must be a minimum interval - defined as Block Delay - until sending the next block.

Block Delay $\geq 160$ ms
### 3.5 Command Set

The Command Byte is part of the block sent from the host.

**Command Bytes used in a Proximity (P) and/or SIM (L) Reader:**

**Operating Mode:**

<table>
<thead>
<tr>
<th>Command Byte</th>
<th>Command Name</th>
<th>Reader</th>
<th>Transponders</th>
</tr>
</thead>
<tbody>
<tr>
<td>'A'</td>
<td>0x41 MutualAuthent</td>
<td>P/L</td>
<td>MDS F4xx MDS F2xx MDS F2xx</td>
</tr>
<tr>
<td>'B'</td>
<td>0x42 ReadBlock</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'D'</td>
<td>0x44 SetPowerDown</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>'E'</td>
<td>0x45 ReadEEData</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'F'</td>
<td>0x46 StartFFT/SetBCD</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>'G'</td>
<td>0x47 GetSnr</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'H'</td>
<td>0x48 HaltSelected</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'I'</td>
<td>0x49 ReadInput</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'K'</td>
<td>0x4B KeyInitMode</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'L'</td>
<td>0x4C SetHFMode</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'M'</td>
<td>0x4D Read_F1</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'O'</td>
<td>0x4F SetOutput</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'P'</td>
<td>0x50 ReadPage</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'R'</td>
<td>0x52 Reset</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'S'</td>
<td>0x53 SelectSnr/SelectLast</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'V'</td>
<td>0x56 GetVersion</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'a'</td>
<td>0x61 TagAuthent</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'b'</td>
<td>0x62 WriteBlock</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'c'</td>
<td>0x63 ConfigPorts</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'e'</td>
<td>0x65 WriteEEData</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'f'</td>
<td>0x66 ReadBCD</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>'h'</td>
<td>0x68 HFReset</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'i'</td>
<td>0x69 ReadPorts</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'l'</td>
<td>0x6C PollTags</td>
<td>P/L</td>
<td>P/L</td>
</tr>
<tr>
<td>'o'</td>
<td>0x6F WritePorts</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'p'</td>
<td>0x70 WritePage</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'r'</td>
<td>0x72 ReadLRStatus</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>'80'</td>
<td>GetSnr_LT/GetSnrReset_LT</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'81'</td>
<td>HaltSelected_LT</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'82'</td>
<td>ReadPage_LT</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'83'</td>
<td>ReadPageInv_LT</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'84'</td>
<td>WritePage_LT</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'90'</td>
<td>PollKbTags</td>
<td>P/L</td>
<td>P/L</td>
</tr>
<tr>
<td>'91'</td>
<td>SetModuleAdr</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'9E'</td>
<td>ReadPublicB_LT</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'A1'</td>
<td>SetProxTrmTime</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'A2'</td>
<td>GetSnr_Adv</td>
<td>P/L</td>
<td></td>
</tr>
<tr>
<td>'A4'</td>
<td>SetBCDOffset</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>'A6'</td>
<td>StopCommand</td>
<td>P/L</td>
<td></td>
</tr>
</tbody>
</table>
Additional commands for a special project requiring a special Reader-Hardware and -Software:

<table>
<thead>
<tr>
<th>Command Byte</th>
<th>Command Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x98</td>
<td>ReadAllPage</td>
</tr>
<tr>
<td>'v'</td>
<td>GetDspVersion</td>
</tr>
</tbody>
</table>

**KeyInit Mode:**

The KeyInit Mode is a mode of all MOBY F Readers for using a set of personalization commands.

<table>
<thead>
<tr>
<th>Command Byte</th>
<th>Command Name</th>
<th>Reader</th>
</tr>
</thead>
<tbody>
<tr>
<td>'C'</td>
<td>0x43 ReadControl</td>
<td>P/L</td>
</tr>
<tr>
<td>'R'</td>
<td>0x52 Reset (Switch to Operating Mode)</td>
<td>P/L</td>
</tr>
<tr>
<td>'V'</td>
<td>0x56 ReadSecret_LT</td>
<td>P/L</td>
</tr>
<tr>
<td>'W'</td>
<td>0x57 WriteSecret_LT</td>
<td>P/L</td>
</tr>
<tr>
<td>'X'</td>
<td>0x58 ReadEEPROM</td>
<td>P/L</td>
</tr>
<tr>
<td>'Y'</td>
<td>0x59 WriteEEPROM</td>
<td>P/L</td>
</tr>
<tr>
<td>'c'</td>
<td>0x63 WriteControl</td>
<td>P/L</td>
</tr>
<tr>
<td>'s'</td>
<td>0x73 WriteSerNum</td>
<td>P/L</td>
</tr>
<tr>
<td>0x90</td>
<td>ReadControl_LT</td>
<td>P/L</td>
</tr>
<tr>
<td>0x91</td>
<td>WriteControl_LT</td>
<td>P/L</td>
</tr>
</tbody>
</table>
### 3.6 Status Byte

The read/write device returns a Status Byte indicating an error if different from 0.

**The following Error Codes are defined:**

<table>
<thead>
<tr>
<th>Value</th>
<th>Error Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>no error</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td>SERIAL ERROR</td>
<td>Error at the serial interface</td>
</tr>
<tr>
<td>-3</td>
<td>NOTAG</td>
<td>There was no answer of a transponder detected by the read/write device.</td>
</tr>
<tr>
<td>-4</td>
<td>TIMEOUT</td>
<td>There is not enough energy available to write to the transponder.</td>
</tr>
<tr>
<td>-5</td>
<td>INCORRECT PASSWORD RWD</td>
<td>The MDS F2xx password of the read/write device is invalid.</td>
</tr>
<tr>
<td>-6</td>
<td>INCORRECT PASSWORD TAG</td>
<td>The MDS F2xx password of the transponder is invalid.</td>
</tr>
<tr>
<td>-7</td>
<td>AUTHENTICATION ERROR</td>
<td>An error occurred during the authentication process.</td>
</tr>
<tr>
<td>-8</td>
<td>ACKNOWLEDGEMENT ERROR</td>
<td>The acknowledgement was not received correctly.</td>
</tr>
<tr>
<td>-9</td>
<td>CRYPTOBLOCK NOT INIT</td>
<td>A cryptographic command was transmitted without authentication between the read/write device and transponder.</td>
</tr>
<tr>
<td>-10</td>
<td>EEPROM ERROR</td>
<td>EEPROM (of the read/write device) acknowledgement error.</td>
</tr>
<tr>
<td>-11</td>
<td>EEPROM WRONG OLD DATA</td>
<td>On comparison old and new data prove inconsistent.</td>
</tr>
<tr>
<td>-12</td>
<td>EEPROM WRITE PROTECTED</td>
<td>You attempted to write to the read/write device EEPROM, although writing was not allowed.</td>
</tr>
<tr>
<td>-13</td>
<td>EEPROM READ PROTECTED</td>
<td>You attempted to read from the read/write device EEPROM, although reading was not allowed.</td>
</tr>
<tr>
<td>-15</td>
<td>CRC ERROR</td>
<td>Wrong CRC from a MDS F4xx transponder in Advanced Protocol Mode.</td>
</tr>
<tr>
<td>-20</td>
<td>ANTENNA OVERLOAD</td>
<td>SIM: Broken or badly detuned antenna (error only after command ReadLRStatus).</td>
</tr>
</tbody>
</table>

---

1 Status byte is currently not evaluated.
3.7 Command Description for Operating Mode

The Operating Mode is a mode of the reader for using a set of standard commands as described in the following.
In this mode the BCC is computed by EXOR-operation of all block data bytes including Block Length.

The command `KeyInitMode` is used to enter the KeyInit Mode (mode of the read/write device for using personalization commands), and a different set of commands becomes available.

3.7.1 GetSnr

This command provides the serial number of a MDS F4xx transponder in Standard Protocol Mode.
For further information on the Standard Protocol Mode see chapter 4 „Transponders“.

**C-Function:** void proloc_GetSnr (DWORD_T *snr, BYTE_T *more);

**Header-File:** PROLIB6.H

**Serial protocol:**

*Host - SIM*

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td>'G'</td>
<td>0x45</td>
</tr>
</tbody>
</table>

*SIM - Host*

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>n+2</td>
<td>Status</td>
<td>SNR-LSB</td>
<td>----</td>
<td>SNR-MSB</td>
<td>more</td>
<td>BCC</td>
</tr>
</tbody>
</table>

n = 0 if an error occurred (error code in Status).
n = 5 if data were read from a transponder (Status = 0).

more: Proximity Reader¹: more is always 0.

SIM: more equal to one indicates that there is at least one additional transponder in the reading area of the SIM.

Status:

- 0 = no error
- -1 = SERIAL ERROR
- -3 = NOTAG

---

¹ Proximity reader is currently not available.
3.7.2 GetSnr_Adv

This command provides the serial number of a MDS F4xx transponder and sets the transponder into Advanced Protocol Mode.

For further information on the Advanced Protocol Mode see chapter 4 "Transponders".

C-Function: void proloc_GetSnr_Adv (DWORD_T *snr, BYTE_T *more);

Header-File: PROLIB6.H

Serial protocol:

Host - SIM

| 0x02 | 0xA2 | BCC |

SIM - Host

| n+2 | Status | SNR-LSB | ----- | SNR-MSB | more | BCC |

n = 0 if an error occurred (error code in Status).
n = 5 if data were read from a transponder (Status = 0).

more: Proximity Reader:

SIM: more is always 0.

more equal to one indicates that there is at least one additional transponder in the reading area of the SIM.

Status:

0 = no error
-1 = SERIAL ERROR
-3 = NOTAG

1 Proximity reader is currently not available.
3.7.3 SelectSnr

This command selects the MDS F4xx transponder with a specified serial number. The content of the transponder's Configuration Page is returned. If there is no such transponder in the field, a NOTAG error message is displayed.

**Note:**
The serial number has to be the same as received with the preceding GetSnr.

**C-Function:** void proloc_SelectSnr (DWORD_T snr, DWORD_T *otp);
**Header-File:** PROLIB6.H

**Serial protocol:**

**Host - SIM**

```
0x06 'S' SNR-LSB ------ SNR-MSB BCC
```

**SIM - Host**

```
7 0 ------ 31 24
n+2 Status OTP-LSB ------ OTP-MSB BCC
```

**OTP:** Configuration Page of MDS F4xx

- \( n = 0 \) if an error occurred (error code in Status).
- \( n = 4 \) if data were read from a transponder (Status = 0).

**Status:**

- \( 0 \) = no error
- \( -1 \) = SERIAL ERROR
- \( -3 \) = NOTAG

3.7.4 SelectLast

Selects the MDS F4xx transponder with the serial number that was read executing the last error-free GetSnr command. This command is an abbreviated version of SelectSnr as no serial number has to be transmitted via the serial interface and the content of the Configuration Page is not returned.

**C-Function:** void proloc_SelectLast (void);
**Header-File:** PROLIB6.H

**Serial protocol:**

**Host - SIM**

```
0x02 'S' 0x51
```

**SIM - Host**

```
0x02 Status BCC
```

**Status:**

- \( 0 \) = no error
- \( -1 \) = SERIAL ERROR
- \( -3 \) = NOTAG
3.7.5 HaltSelected

Puts the selected MDS F4xx transponder into Halt Mode, which means that this transponder remains silent until it leaves and reenters the RF field.

You can reset a transponder previously turned off by HaltSelected using the command HFReset or putting it out of RF field.

C-Function: void proloc_HaltSelected (void);
Header-File: PROLIB6.H

Serial protocol:

<table>
<thead>
<tr>
<th>Host - SIM</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td>'H'</td>
<td>0x4A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIM - Host</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Status:
- 0  = no error
- -1 = SERIAL ERROR
- -8 = ACKNOWLEDGEMENT ERROR
3.7.6 ReadPage

Reads a page (4 bytes) of a selected MDS F4xx transponder. If no transponder is selected, a NOTAG message will be generated even if there is a transponder in the communication field of the antenna.

Using the byte -crypto- you define whether you want to work in Plain or in Crypto Mode. Access to the secret area is only possible in Crypto Mode after a mutual authentication.

If -crypto- equals 1 (Crypto Mode) and you did not run an authentication procedure before, Status will be set to -9.

**C-Function:**
void proloc_ReadPage (BYTE_T crypto, BYTE_T pagenr, char *data);

**Header-File:**
PROLIB6.H

**Serial protocol:**

**Host - SIM**

\[
\begin{array}{c|c|c|c|c}
0x04 & 'P' & crypto & pagenr & BCC \\
\end{array}
\]

- \(crypto\): 0x00 = Plain Mode
- \(0x01 = Crypto Mode\)
- \(pagenr\): page number

**SIM - Host**

\[
\begin{array}{c|c|c|c|c|c|c}
\text{n+2} & \text{Status} & \text{data[0]} & \ldots & \text{data[3]} & \text{BCC} \\
\end{array}
\]

- \(n = 0\) if an error occurred (error code in Status).
- \(n = 4\) if data were read from a transponder (Status = 0).

**Status:**
- 0 = no error
- -1 = SERIAL ERROR
- -3 = NOTAG
- -9 = CRYPTOBLOCK NOT INIT
3.7.7 ReadBlock

Reads a block (16 bytes) of the selected MDS F4xx transponder. If no transponder is selected, a NOTAG message will be generated even if there is a transponder in the communication field of the antenna.

The start address is specified by -pagenr-. Data is read from the start address until the end of the corresponding block. Thus a data length of 4, 8, 12 or 16 bytes is possible.

Use byte -crypto- to define whether you want to work in Plain or in Crypto Mode. Access to the secret area is only possible in Crypto Mode after a mutual authentication.

If -crypto- equals 1 (Crypto Mode) and you did not run an authentication procedure before, Status will be set to -9.

C-Function: void proloc_ReadBlock (BYTE_T crypto, BYTE_T pagenr, char *data);
Header-File: PROLIB6.H

Serial protocol:

Host - SIM

| 0x04 | 'B' | crypto | pagenr | BCC |

crypto: 0x00 = Plain Mode
0x01 = Crypto Mode

pagenr: page number (for start address)

SIM - Host

| n+2 | Status | data[0] | ...... | data[n-1] | BCC |

n = 0 if an error occurred (error code in Status).
n = 4, 8, 12, 16 depending on the page address if data were read from a transponder (Status = 0).

Status:
0 = no error
-1 = SERIAL ERROR
-3 = NOTAG
-9 = CRYPTOBLOCK NOT INIT
3.7.8 WritePage

Writes a page (4 bytes) to the selected MDS F4xx transponder. If no transponder is selected, a NOTAG message will be generated even if there is a transponder in the communication field of the antenna.

Use byte -crypto- to define whether you want to work in Plain or in Crypto Mode. Access to the secret area is only possible in Crypto Mode after a mutual authentication.

If -crypto- equals 1 (Crypto Mode) and you did not run an authentication process before, Status will be set to -9.

**Note:**
To substantially increase the data reliability we strictly recommend to read the previously written data (read after write).

**C-Function:**

```c
void proloc_WritePage (BYTE_T crypto, BYTE_T pagenr, char *data);
```

**Header-File:**

`PROLIB6.H`

**Serial protocol:**

**Host - SIM**

<table>
<thead>
<tr>
<th>0x08</th>
<th>'p'</th>
<th>crypto</th>
<th>pagenr</th>
<th>data[0]</th>
<th>......</th>
<th>data[3]</th>
<th>BCC</th>
</tr>
</thead>
</table>

- crypto: 0x00 = Plain Mode
- 0x01 = Crypto Mode
- pagenr: page number

**SIM - Host**

<table>
<thead>
<tr>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

- Status: 0 = no error
- -1 = SERIAL ERROR
- -3 = NOTAG
- -4 = TIMEOUT
- -9 = CRYPTOBLOCK NOT INIT
3.7.9 WriteBlock

Writes a block (16 bytes) to the selected MDS F4xx transponder. If no transponder is selected, a NOTAG message will be generated even if there is a transponder in the communication field of the antenna.

The start address is specified by -pagenr-. Data is written from the start address until the end of the corresponding block. Thus a data length of 4, 8, 12 or 16 bytes is possible.

Use byte -crypto- to define whether you want to work in Plain or in Crypto Mode. Access to the secret area is only possible in Crypto Mode after a mutual authentication.

If -crypto- equals 1 (Crypto Mode) and you did not run an authentication procedure before, Status will be set to -9.

Note:

To substantially increase the data reliability we strictly recommend to read the previously written data (read after write).

C-Function: void proloc_WriteBlock (BYTE_T crypto, BYTE_T pagenr, char *data);

Header-File: PROLIB6.H

Serial protocol:

<table>
<thead>
<tr>
<th>n+4</th>
<th>'b'</th>
<th>crypto</th>
<th>pagenr</th>
<th>data[0]</th>
<th>......</th>
<th>data[n-1]</th>
<th>BCC</th>
</tr>
</thead>
</table>

crypto: 0x00 = Plain Mode
        0x01 = Crypto Mode

pagenr: page number (for start address)

SIM - Host

<table>
<thead>
<tr>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

Status: 0 = no error
        -1 = SERIAL ERROR
        -3 = NOTAG
        -4 = TIMEOUT
        -9 = CRYPTOBLOCK NOT INIT
3.7.10 TagAuthent

Carries out the single authentication procedure for MDS F4xx transponders (authentication of the transponder). The authentication procedure is aborted after sending the transponder logdata.

Using -keyinfo- you can choose between Key/Logdata Set A and B. This command can be used - e.g. - to check if Keys and Logdata in the transponder and the read/write device are the same. ("Check, if the transponder is member of the same ‘family´ as the read/write device").

**Note:**
You cannot use any Crypto commands after TagAuthent.

After this abbreviated authentication procedure the transponder can only be accessed using GetSnr or the command HFReset.

**C-Function:**
void proloc_TagAuthent (BYTE_T keyinfo);

**Header-File:**
PROLIB6.H

**Serial protocol:**

*Host - SIM*

| 0x03 | 'a' | keyinfo | BCC |

*keyinfo:*  
0x00 = Key/Logdata-Set A  
0x01 = Key/Logdata-Set B

*SIM - Host*

| 0x02 | Status | BCC |

*Status:*  
0 = no error  
-1 = SERIAL ERROR  
-7 = AUTHENTICATION ERROR
3.7.11 MutualAuthent

Carries out the full authentication procedure between the transponder and the read/write device. After this mutual authentication you are allowed to edit areas which can only be accessed in Crypto Mode.

Using -keyinfo- you can choose between Key/Logdata Set A and B.

Use a Plain command (that is still encrypted), HFReset or GetSnr (resets the already selected transponder) to exit this mode.

C-Function: void proloc_MutualAuthent (BYTE_T keyinfo);
Header-File: PROLIB6.H

Serial protocol:

Host - SIM

| 0x03 | ‘A’ | keyinfo | BCC |

keyinfo:
- 0x00 = Key/Logdata-Set A
- 0x01 = Key/Logdata-Set B

SIM - Host

| 0x02 | Status | BCC |

Status:
- 0 = no error
- -1 = SERIAL ERROR
- -7 = AUTHENTICATION ERROR
3.7.12 GetSnr_LT

This command is applied to a MDS F2xx transponder being in Password or Crypto Mode. The command selects the transponder and provides its serial number and Configuration Byte -config-.

If the byte -Status- shows „no error“ the transponder is selected and ready for read or write accesses.

The byte -mode- selects one of two possible modes: Password or Crypto.

C-Function: void proloc_GetSnr_LT (BYTE_T mode, DWORD_T *snr, BYTE_T *config);
Header-File: PROLBLT6.H

Serial protocol:

Host - SIM

| 0x03 | 0x80 | mode | BCC |

SIM - Host

| 7 | 0 | ------ | 31 | 24 |

| n+2 | Status | SNR-LSB | ------ | SNR-MSB | config | BCC |

config: Configuration Byte of MDS F2xx
n = 0 if an error occurred (error code in Status).
n = 5 if data were read from a transponder (Status = 0).

Status:  
0 = no error  
-1 = SERIAL ERROR  
-3 = NOTAG  
-5 = INCORRECT PASSWORD RWD  
-6 = INCORRECT PASSWORD TAG  
-7 = AUTHENTICATION ERROR
3.7.13 GetSnrReset_LT

This command is applied to a MDS F2xx transponder which is currently not in Password or Crypto Mode but in one of the Public Modes. The command selects the transponder and provides its serial number and Configuration Byte.

If the byte -Status- shows „no error“ the transponder is selected and ready for read or write accesses.

The byte -mode- decides whether the selection process for the transponder is done corresponding to the Password Mode or the Crypto Mode.

**C-Function:**

```c
void proloc_GetSnrReset_LT (BYTE_T Mode, DWORD_T *snr,
                            BYTE_T *config);
```

**Header-File:** PROBLT6.H

**Serial protocol:**

**Host - SIM**

```
0x04 0x80 mode 'M' BCC
```

*mode:*
- 0x00 = Password
- 0x01 = Crypto

**SIM - Host**

```
7 0 ------ 31 24
```

*config:* Configuration Byte of MDS F2xx

n = 0 if an error occurred (error code in Status).
n = 5 if data were read from a transponder (Status = 0).

*Status:*
- 0 = no error
- -1 = SERIAL ERROR
- -3 = NOTAG
- -5 = INCORRECT PASSWORD RWD
- -6 = INCORRECT PASSWORD TAG
- -7 = AUTHENTICATION ERROR
### 3.7.14 HaltSelected_LT

Puts the selected MDS F2xx transponder into Halt Mode, which means that this transponder remains silent until it leaves the RF field.

You can reset a transponder previously turned off by `HaltSelected_LT` using the command `HFReset` or putting it out of RF field.

**C-Function:** `void proloc_HaltSelected_LT(void);`

**Header-File:** PROLBLT6.H

**Serial protocol:**

**Host - SIM**

| 0x02 | 0x81 | 0x83 |

**SIM - Host**

| 0x02 | Status | BCC |

Status:

- 0 = no error
- -1 = SERIAL ERROR
- -8 = ACKNOWLEDGEMENT ERROR

### 3.7.15 ReadPage_LT

Reads a page (4 bytes) of a selected MDS F2xx transponder.

If no transponder is selected, a NOTAG message will be generated.

This command should be used together with `ReadPageInv_LT` to compare plain data with the bit-inverted data to gain maximum data reliability.

**C-Function:** `void proloc_ReadPage_LT(BYTE_T pagenr, char *data);`

**Header-File:** PROLBLT6.H

**Serial protocol:**

**Host - SIM**

| 0x03 | 0x82 | pagenr | BCC |

**SIM - Host**

| n+2 | Status | data[0] | ...... | data[3] | BCC |

n = 0 if an error occurred (error code in Status).

n = 4 if data were read from a transponder (Status = 0).

Status:

- 0 = no error
- -1 = SERIAL ERROR
- -3 = NOTAG
3.7.16 ReadPageInv_LT

Reads a bit-inverted page (4 bytes) of a selected MDS F2xx transponder. If no transponder is selected, a NOTAG message will be generated.

This command should be used together with ReadPage_LT to compare plain data with the bit-inverted data to gain maximum data reliability.

C-Function:      void proloc_ReadPageInv_LT (BYTE_T pagenr, char *data);
Header-File:     PROLBLT6.H

Serial protocol:

Host - SIM

| 0x03 | 0x83 | pagenr | BCC |

pagenr:    page number

SIM - Host

| n+2 | Status | data[0] | ...... | data[3] | BCC |

n = 0 if an error occurred (error code in Status).
n = 4 if data were read from a transponder (Status = 0).

Status:
0 = no error
-1 = SERIAL ERROR
-3 = NOTAG
3.7.17 WritePage_LT

 Writes a page (4 bytes) onto the selected MDS F2xx transponder. If no transponder is selected, a NOTAG message will be generated.

**Note:**
To check if WritePage_LT was successful it is important that the immediately following command is a ReadPage_LT. If ReadPage_LT does not return „no error“ and the right data, you have to repeat WritePage_LT.

**C-Function:**
void proloc_WritePage_LT (BYTE_T pagenr, char *data);

**Header-File:**
PROLBLT6.H

**Serial protocol:**

<table>
<thead>
<tr>
<th>Host - SIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x07</td>
</tr>
</tbody>
</table>

pagenr: page number

<table>
<thead>
<tr>
<th>SIM - Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
</tr>
</tbody>
</table>

Status:
- 0 = no error
- -1 = SERIAL ERROR
- -3 = NOTAG
- -4 = TIMEOUT
3.7.18 ReadPublicB_LT

This command sets the read/write device to Permanent Reading Mode for MDS F2xx transponders being in Public Mode B.

The read/write device attempts continuously to synchronize on and read a MDS F2xx transponder in Public Mode B. If it succeeds and all checks report positive results, the device sends the 16 data bytes (a 128-bit-stream that has to be prepared afterwards for subsequent treatment) via the serial interface. After that the read/write device returns to Normal Mode. The software running on the host has to decode the read data depending on the chosen data protocol.

To put the read/write device back to normal mode, a StopCommand should be sent. Do not use a Reset, since Reset can cause undesirable side effects (resetting output pins).

Since the tag sends its 128-bit data continuously, the user must store its data on the tag in a way which allows for synchronization.

C-Function: void proloc_ReadPublicB_LT (BYTE_T *data);
Header-File: PROLBLT6.H

Serial protocol:

<table>
<thead>
<tr>
<th>Host - SIM</th>
<th>SIM - Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td>0x12</td>
</tr>
<tr>
<td>0x9E</td>
<td>Status</td>
</tr>
<tr>
<td>BCC</td>
<td>data[0]</td>
</tr>
<tr>
<td></td>
<td>-----</td>
</tr>
<tr>
<td></td>
<td>data[15]</td>
</tr>
<tr>
<td></td>
<td>BCC</td>
</tr>
</tbody>
</table>

Status: 0 = no error
-1 = SERIAL ERROR
3.7.19 Read_F1

Sets the read/write device to Permanent Reading Mode for MDS F1xx compatible transponders.
In this mode you can read either MDS F2xx transponders in Public Mode A or MDS F1xx transponders.
The unique serial number of a MDS F1xx transponder consists of 5 bytes.
The read/write device attempts continuously to synchronize on and read a MDS F1xx transponder.
If it succeeds and all checks report positive results, the device sends the 5 data bytes via the serial interface.
After that the read/write device returns to Normal Mode.
To put the read/write device back to normal mode, a StopCommand should be sent. Do not use a Reset, since Reset can cause undesirable side effects (resetting output pins).

MDS F2xx-compatible data protocol for using MDS F2xx transponders in Public Mode A (data is stored on Pages 4 and 5 of a MDS F2xx transponder):

| 9 bit header (= ‘1’) | 9 bit |
| 10 * 4 bit ID data + 10 * 1 bit even parity | 50 bit |
| 4 bits even parity for columns (of ID data nibbles) | 4 bit |
| last bit (= ‘0’) | 1 bit |
| total | 64 bit |

C-Function: void proloc_Read_F1 (char *data);
Header-File: PROLIB6.H

Serial protocol:

Host - SIM

| 0x02 | ‘M’ | 0x4F |

SIM - Host

| 0x07 | Status | data[0] | ----- | data[4] | BCC |

Status: 0 = no error
-1 = SERIAL ERROR
3.7.20 PollTags

This command sets the read/write device to Permanent Reading Mode for specified types of transponders.

The read/write device attempts continuously to synchronize on and read specified types of transponders. If it succeeds and all checks report positive results, the device sends data for transponder identification via the serial interface. After that the read/write device returns to Normal Mode.

Using the byte -mode- you select the types of transponders for poll-operation.

To avoid conflicts it is important to set only one bit at a time for following transponder types:
- MDS F2xx PublicC or MDS F2xx PublicB
- MDS F4xx Standard Protocol Mode or MDS F4xx Advanced Protocol Mode

If bit 3 (MDS F2xx Password Mode) or bit 4 (MDS F2xx Crypto Mode) is selected, we recommend to activate the „Check PW TAG“ option in Control_LT (located in the EEPROM of the read/write device ... see Chapter 5 „Personalization“) to reduce the possibility to erroneously identify other types of (especially read-only) transponders as MDS F2xx Password or MDS F2xx Crypto.

C-Function: void proloc_PollTags (BYTE_T mode, char *data);
Header-File: PROLIB6.H

Serial protocol:

Host - SIM

| 0x03 | 'l' | mode | BCC |

mode:

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
</table>

SIM - Host

If polling for MDS F4xx Standard Protocol Mode was successful:

| 0x08 | Status | 0x01 | SNR-LSB | ----- | SNR-MSB | more | BCC |

more: Proximity Reader1: more is always 0.
SIM: more equal to one indicates that there is at least one additional transponder in the reading area of the SIM.

---

1 Proximity reader is currently not available.
If polling for MDS F4xx Advanced Protocol Mode was successful:

<table>
<thead>
<tr>
<th>Status</th>
<th>0x08</th>
</tr>
</thead>
<tbody>
<tr>
<td>more</td>
<td>0x80</td>
</tr>
<tr>
<td>SNR-LSB</td>
<td>-----</td>
</tr>
<tr>
<td>SNR-MSB</td>
<td>more</td>
</tr>
<tr>
<td>BCC</td>
<td>----</td>
</tr>
</tbody>
</table>

more: Proximity Reader:

SIM: more is always 0.

more is equal to one indicates that there is at least one additional transponder in the reading area of the SIM.

If polling for MDS F2xx / MDS F2xx Public Mode A was successful:

<table>
<thead>
<tr>
<th>Status</th>
<th>0x08</th>
</tr>
</thead>
<tbody>
<tr>
<td>data[0]</td>
<td>0x02</td>
</tr>
<tr>
<td>data[4]</td>
<td>-----</td>
</tr>
<tr>
<td>BCC</td>
<td>----</td>
</tr>
</tbody>
</table>

If polling for MDS F2xx Password Mode was successful:

<table>
<thead>
<tr>
<th>Status</th>
<th>0x08</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNR-LSB</td>
<td>-----</td>
</tr>
<tr>
<td>SNR-MSB</td>
<td>config</td>
</tr>
<tr>
<td>BCC</td>
<td>----</td>
</tr>
</tbody>
</table>

If polling for MDS F2xx Crypto Mode was successful:

<table>
<thead>
<tr>
<th>Status</th>
<th>0x10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNR-LSB</td>
<td>-----</td>
</tr>
<tr>
<td>SNR-MSB</td>
<td>config</td>
</tr>
<tr>
<td>BCC</td>
<td>----</td>
</tr>
</tbody>
</table>

If polling for MDS F2xx Public Mode C was successful:

<table>
<thead>
<tr>
<th>Status</th>
<th>0x20</th>
</tr>
</thead>
<tbody>
<tr>
<td>data[0]</td>
<td>-----</td>
</tr>
<tr>
<td>data[15]</td>
<td>BCC</td>
</tr>
</tbody>
</table>

If polling for MDS F2xx Public Mode B was successful:

<table>
<thead>
<tr>
<th>Status</th>
<th>0x40</th>
</tr>
</thead>
<tbody>
<tr>
<td>data[0]</td>
<td>-----</td>
</tr>
<tr>
<td>data[15]</td>
<td>BCC</td>
</tr>
</tbody>
</table>

Status:

0 = no error

-1 = SERIAL ERROR

---

1 Proximity reader is currently not available.
3.7.21 PollKbTags

This command polls once for a transponder in the RF-field of the read/write device antenna and reads the keyboard-buffer and the digital inputs In1. The read/write device does not enter the Permanent Reading Mode.

C-Function:    void proloc_PollKBTags (BYTE_T mode, char *data);
Header-File:   PROLIB6.H

Serial protocol:

Host - SIM

<table>
<thead>
<tr>
<th>0x03</th>
<th>0x90</th>
<th>mode</th>
<th>BCC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

mode:
- 0x00 = poll keyboard-matrix and inputs
- 0x80 = poll serial number of MDS F4xx transponder in Standard Protocol Mode
- 0x81 = poll serial number of MDS F2xx transponder in Crypto Mode, keyboard-matrix and inputs
- 0x82 = poll serial number of MDS F2xx transponder in Password Mode, keyboard matrix and inputs
- 0x83 = poll serial number of MDS F4xx transponder in Advanced Protocol Mode, keyboard-matrix and inputs

SIM - Host

<table>
<thead>
<tr>
<th>n+2</th>
<th>info</th>
<th>optional 4 bytes</th>
<th>optional 4 bytes</th>
<th>BCC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>keyboard-buffer</td>
<td>serial number</td>
<td></td>
</tr>
</tbody>
</table>

n = 0  no transponder in RF-filed
n = 4  keyboard-buffer not empty or transponder in RF-field
n = 8  keyboard-buffer not empty and transponder in RF-field

info:
- bit0 = state of input In1
- bit1 = state of input In2 1
- bit6 = when set protocol contains keyboard-buffer
- bit7 = when set protocol contains serial number

1 For SIM: Only input In1
**Keyboard-Buffer**
(appended to protocol when keyboard-buffer is not empty)

|---------|---------|---------|---------|

Keyb[0] Bit 4-7       oldest key-code  
Keyb[0] Bit 0-3  
.  
.  
.  

Keyb[3] Bit 4-7       second newest key-code  
Keyb[3] Bit 0-3       newest key-code  

**Serial number**
(appended to the protocol if a transponder of requested type was found)

<table>
<thead>
<tr>
<th>SNR-LSB</th>
<th>------</th>
<th>SNR-MSB</th>
</tr>
</thead>
</table>

3.7.22 GetVersion

This command retrieves the serial number of the read/write device, the version number of the MOBY F SIM software and its date of creation.

C-Function:    void proloc_GetVersion (char *data);
Header-File:   PROLIB6.H

**Serial protocol:**

*Host - SIM*

| 0x02 | 'V' | BCC |

*SIM - Host*

| 0x1D | Status | data[0] | ------ | data[26] | BCC |

- `data[0]` to `data[7]`: Version (format: Vx.yy.zz)
- `data[8]` to `data[15]`: Date (format: dd-mm-yyyy)
- `data[16]` to `data[26]`: Serial number (11 characters)

Status:

- 0 = no error
- -1 = SERIAL ERROR
### 3.7.23 Reset

This command resets basic functions of the read/write device. All port-pins of the MOBY F SIM are reset to an initial state (output pins are set to '0', input pins are set to '1').
You should not interrupt the Permanent Reading Mode (activated after Read_F1, ...) of the read/write device by invoking this command, since Reset can cause undesirable side effects (resetting output pins). Use StopCommand instead.

**C-Function:**
```c
void proloc_Reset (void);
```
**Header-File:**
`PROLIB6.H`

**Serial protocol:**

<table>
<thead>
<tr>
<th>Host - SIM</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td>'R'</td>
</tr>
<tr>
<td>0x50</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIM - Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
</tr>
<tr>
<td>0x02</td>
</tr>
</tbody>
</table>

**Status:**
- `0` = no error
- `-1` = SERIAL ERROR

### 3.7.24 HFReset

This function turns off the RF-part of the read/write device for a certain time (about 100 ms in a Proximity Reader, about 40 ms in a SIM).
This means that all MOBY F transponders are reset and transponders that were in Halt Mode will respond again.

**C-Function:**
```c
void proloc_HFReset (void);
```
**Header-File:**
`PROLIB6.H`

**Serial protocol:**

<table>
<thead>
<tr>
<th>Host - SIM</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td>'h'</td>
</tr>
<tr>
<td>0x6A</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIM - Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
</tr>
<tr>
<td>0x02</td>
</tr>
</tbody>
</table>

**Status:**
- `0` = no error
- `-1` = SERIAL ERROR
### 3.7.25 StopCommand

The command *StopCommand* interrupts the Permanent Reading Mode (activated after *Read_F1*...) of the read/write device.

You should not use the command *Reset* instead, since *Reset* can cause undesirable side effects (resetting output pins).

**C-Function:** void proloc_StopCommand (void);

**Header-File:** PROLIB6.H

**Serial protocol:**

<table>
<thead>
<tr>
<th>Host - SIM</th>
<th>0x02</th>
<th>0xA6</th>
<th>BCC</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SIM - Host</th>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

Status:

- 0 = no error
- -1 = SERIAL ERROR
3.7.26 **ReadInput**

You can read input-ports of the MOBY F SIM by using the command *ReadInput*.

---

**Note:**

- Pins are internally pulled up!
- Using Siemens SIM the state of input In1 is inverted (input is buffered by an inverting schmitt trigger input driver).

---

**There are certain restrictions concerning the applied hardware:**

- **In1:** available for Proximity Readers and SIMs
- **In2:** available only for Proximity Readers

**C-Function:**

```c
void proloc_ReadInput (BYTE_T *input);
```

**Header-File:**

```c
PROLIB6.H
```

**Serial protocol:**

**Host - SIM**

```
0x02 'I' 0x4B
```

**SIM - Host**

```
0x03 Status input BCC
```

**input:**

```
<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>In2</td>
<td>In1</td>
<td></td>
</tr>
</tbody>
</table>
```

Proximity Reader\(^1\) (In1,In2):
- 0 = reset (0 V)
- 1 = set (5 V)

SIM (In1):
- 0 = reset (0 V)\(^2\)
- 1 = set (5 V)\(^2\)

**Status:**

- 0 = no error
- -1 = SERIAL ERROR

---

1. Proximity reader is currently not available.
2. For exact definition of the voltage level, see chapter 6.6.2 of the MOBY F manual on configuration, installation and service (order no. 6GT2 497-4BA00-0EA2)
3.7.27 SetOutput

You can set (5 V) or reset (0 V) output-ports of the MOBY F SIM by SetOutput.

**Note:**
Using SIM the state of output Out1 is inverted (output is buffered by an inverting CMOS driver).

**There are certain restrictions concerning the applied hardware**

Out1 (P2.0): available for Proximity Readers and SIMs
Out2 (P2.1): available only for Proximity Readers
Out3 (P1.4): available only with a special hardware including connection of this pin (signal is not available on pin connectors of Siemens Core Module)
Out4 (P2.7): available only with a special hardware including connection of this pin (signal is not available on pin connectors of Siemens Core Module)

**C-Function:**
void proloc_SetOutput (BYTE_T output);

**Header-File:**
PROLIB6.H

**Serial protocol:**

*Host - SIM*

<table>
<thead>
<tr>
<th>0x03</th>
<th>'O'</th>
<th>output</th>
<th>BCC</th>
</tr>
</thead>
</table>

output:

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3 (Out4)</th>
<th>(Out3)</th>
<th>Out2</th>
<th>Out1</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Proximity Reader (Out1,Out2): 0 = reset (0 V) 1 = set (5 V)
SIM (Out1): 0 = reset (0 V) 1 = set (24 V)

*SIM - Host*

<table>
<thead>
<tr>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

Status:
0 = no error
-1 = SERIAL ERROR

---

1 Proximity reader is currently not available.
2 For exact definition of the voltage level, see chapter 6.6.2 of the MOBY F manual on configuration, installation and service (order no. 6GT2 497-4BA00-0EA2)
3.7.28 ConfigPorts

This command writes a new Port 0 Configuration-Byte into the EEPROM of the read/write device. The Port 0 Configuration-Byte (\texttt{-config-}) defines, whether a Port-0-pin of the MOBY F Communication Controller has to be handled as an input or as an output.

Initial value stored in the EEPROM of a delivered read/write device:

\begin{verbatim}
config = 0x00
\end{verbatim}

\textit{ConfigPorts} automatically initializes the status of input-configurated pins to ‘1’ (5V). The status of output-configurated pins is left unchanged.

\begin{itemize}
  \item \textbf{Note:}
  \end{itemize}

To use commands referring to Port 0 you need a special hardware with connected Port 0 signals is required (Port 0 signals are not available on pin connectors of Core Modules of the SIMs).

\begin{itemize}
  \item Power-Up or Reset of the read/write device:
    \begin{itemize}
      \item The Port 0 Configuration-Byte is not lost (because stored in EEPROM).
      \item Input-configurated pins are initialized to HIGH (5 V), output-configurated to LOW (0 V) by the read/write device operating system.
    \end{itemize}
\end{itemize}

\textbf{C-Function:} \texttt{void proloc\_ConfigPorts (BYTE\_T config);}  
\textbf{Header-File:} \texttt{PROLIB6.H}

\textbf{Serial protocol:}

\textit{Host - SIM}

\begin{verbatim}
0x03  'c' config  BCC
\end{verbatim}

\underline{config}:

\begin{verbatim}
<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0.7</td>
<td>P0.6</td>
<td>P0.5</td>
<td>P0.4</td>
<td>P0.3</td>
<td>P0.2</td>
<td>P0.1</td>
<td>P0.0</td>
</tr>
<tr>
<td>0 = configure as input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = configure as output</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\end{verbatim}

\textit{SIM - Host}

\begin{verbatim}
0x02  Status  BCC
\end{verbatim}

\underline{Status}:

\begin{verbatim}
0 = no error  
-1 = SERIAL ERROR
\end{verbatim}
## 3.7.29 ReadPorts

This command reads the status of those port Pins of the MOBY F SIM (Port 0) that are configured as inputs.

### Note:
To use commands referring to Port 0 you need a special hardware with connected Port 0 signals is required (Port 0 signals are not available on pin connectors of Core Modules of the SIMs).

Bit-positions of output-configured pins are read as ‘0’

**C-Function:**
```c
void proloc_ReadPorts (BYTE_T *input);
```

**Header-File:**
`PROLIB6.H`

### Serial protocol:

**Host - SIM**
```
0x02  'i'  BCC
```

**SIM - Host**
```
0x03 | Status | input | BCC
```

**input:**

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0.7</td>
<td>P0.6</td>
<td>P0.5</td>
<td>P0.4</td>
<td>P0.3</td>
<td>P0.2</td>
<td>P0.1</td>
<td>P0.0</td>
</tr>
</tbody>
</table>
| 0 = reset (0 V)  
1 = set (5 V) |

**Status:**

- 0 = no error
- -1 = SERIAL ERROR
3.7.30 WritePorts

This command changes the status of those port pins of the MOBY F SIM (Port 0) that are configured as outputs.

Note:
To use commands referring to Port 0 you need a special hardware with connected Port 0 signals is required (Port 0 signals are not available on pin connectors of Core Modules of the SIMs).

Write accesses to input-configured pins always result in writing ‘1’ (5 V) to the pins.

C-Function: void proloc_WritePorts (BYTE_T mode, BYTE_T output);

Header-File: PROLIB6.H

Serial protocol:

Host - SIM

| 0x04 | 'o' | output | mode | BCC |

output:

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0.7</td>
<td>P0.6</td>
<td>P0.5</td>
<td>P0.4</td>
<td>P0.3</td>
<td>P0.2</td>
<td>P0.1</td>
<td>P0.0</td>
</tr>
</tbody>
</table>

mode: 0 = The bits in -output- are directly written to the output-configured port pins.
1 = The current status of the output-configured port pins is AND-combined with the bits in -output-. The result is written to the output-configured port pins
2 = The current status of the output-configured port pins is OR-combined with the bits in -output-. The result is written to the output-configured port pins.
3 = The current status of the output-configured port pins is EXOR-combined with the bits in -output-. The result is written to the output-configured port pins.

SIM - Host

| 0x02 | Status | BCC |

Status: 0 = no error
-1 = SERIAL ERROR
3.7.31 ReadEEData

Reads - starting with the chosen address - up to 16 data bytes from the user memory in the EEPROM of the MOBY F read/write device. If you reach the limit of the address area the command is finished.

C-Function: \sa{void proloc_ReadEEData (BYTE_T addr, BYTE_T bytenmb, char *data);}

Header-File: \sa{PROLIB6.H}

Serial protocol:

Host - SIM

| 0x04 | 'E' | addr | bytenmb | BCC |

addr: EEPROM user address (0 ≤ addr ≤ 84)
bytenmb: number of bytes to read (1 ≤ bytenmb ≤ 16)

SIM - Host

| n + 2 | Status | data[0] | ....... | data[n-1] | BCC |

Status: 0 = no error
-1 = SERIAL ERROR
-10 = EEPROM ERROR

3.7.32 WriteEEData

Writes - starting with the chosen address - up to 16 data bytes into the user memory of the EEPROM of the read/write device. If you reach the limit of the address area the command is finished.

C-Function: \sa{void proloc_WriteEEData (BYTE_T addr, BYTE_T bytenmb, char *data);}

Header-File: \sa{PROLIB6.H}

Serial protocol:

Host - SIM

| bytenmb+4 | 'e' | addr | bytenmb | data[0] | ....... | data[bytenmb-1] | BCC |

addr: EEPROM user address (0 ≤ addr ≤ 84)
bytenmb: number of bytes to write (1 ≤ bytenmb ≤ 16)

SIM - Host

| 0x02 | Status | BCC |

Status: 0 = no error
-1 = SERIAL ERROR
-10 = EEPROM ERROR
### 3.7.33 SetProxTrmTime

This Proximity Reader command writes new RF-bit-times $t_0$, $t_1$, $t_p$ into the EEPROM of the read/write device.

**Note:**
It is not necessary to use this command when working with Proximity Readers because EEPROM is already initialized to following standard values:

- $t_0 = 0xA9 \text{ (176}\,\mu\text{s}):$ Duration of a ‘0’-bit including $t_p$
- $t_1 = 0x81 \text{ (224}\,\mu\text{s}):$ Duration of a ‘1’-bit including $t_p$
- $t_p = 0xEC \text{ (48}\,\mu\text{s):}$ Duration of a Modulation Gap

**Note:**
The values for $t_0$, $t_1$, $t_p$ do not represent the RF-bit-times in $\mu$s. They have to be computed. If you provide $T_0$, $T_1$, $T_P$ in $\mu$s you can compute $t_0$, $t_1$ and $t_p$ using following code sequence:

```c
/* TP >= 43 \mu s; T0 >= T_P + 40 \mu s; T1 >= T_P + 40 \mu s; */
t_0=( unsigned char)(32768-((T0-T_P-24)/1.2));
t_1=( unsigned char)(32768-((T1-T_P-24)/1.2));
t_p=( unsigned char)(32768-((TP-24)/1.2));
```

**C-Function:**
```c
void proloc_SetProxTrmTime (BYTE_T t_0, BYTE_T t_1, BYTE_T t_p);
```

**Header-File:** PROLIB6.H

**Serial protocol:**

**Host - SIM**

| 0x05 | 0xA1 | t_0 | t_1 | t_p | BCC |

**SIM - Host**

| 0x02 | Status | BCC |

**Status:**
- $0$ = no error
- $-1$ = SERIAL ERROR
3.7.34 SetModuleAdr

The command SetModuleAdr is used to assign a unique node-address to a device whose serial number is known. The new node-address is written into the EEPROM of the read/write device.

Initial value stored in the EEPROM of a delivered read/write device:

$$addr = 0x00$$

SetModuleAdr should be sent in Ordinary Protocol. If the right serial number was sent, the read/write device answers with Ordinary Protocol if its former node-address was zero, otherwise it answers in Extended Protocol.

If the serial number does not match, the command is ignored, and there will be no answer from the reader.

You can read the serial number of the read/write device by using the command GetVersion.

**C-Function:**

```c
void proloc_SetModuleAdr (BYTE_T addr, char *snr);
```

**Header-File:**

PROLIB6.H

For communication in Extended Protocol use commands with ‘Proloc_M’-prefix. For further information see Header File PROLBMU6.h

**Serial protocol:**

**Host - SIM**

<table>
<thead>
<tr>
<th>0x0E</th>
<th>0x91</th>
<th>Snr[0]</th>
<th>......</th>
<th>Snr[10]</th>
<th>addr</th>
<th>BCC</th>
</tr>
</thead>
</table>

$addr$: 0x00 = for communication with a single reader using the Ordinary Protocol.
>0x00 = for communication with multiple readers (e.g. in RS485 net) using the Extended protocol. Each reader gets a specific address.

**SIM - Host**

((only if serial number matches!))

<table>
<thead>
<tr>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

Status:

0 = no error
-1 = SERIAL ERROR
3.7.35 SetHFMode

This command sets the read/write device into Proximity or SIM Mode.

In standard applications (e.g. using SIM) SetHFMode is not used because the MOBY F SIM automatically sets the right mode after power-up.

Examples:

- SetHFMode setting the mode to SIM sets a Proximity Reader in a powerdown-state with reduced power consumption.
- SetHFMode can be used in a system with a Proximity RF-part and a SIM RF-part to select one part at a time.

C-Function: void proloc_SetHFMode (BYTE_T mode);

Header-File: PROLIB6.H

Serial protocol:

<table>
<thead>
<tr>
<th>Host - SIM</th>
<th>0x03</th>
<th>'L'</th>
<th>mode</th>
<th>BCC</th>
</tr>
</thead>
</table>

mode: 0x00 = Proximity mode
0x01 = SIM mode

<table>
<thead>
<tr>
<th>SIM - Host</th>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

Status: 0 = no error
-1 = SERIAL ERROR
3.7.36 StartFFT

This SIM command starts the FFT (Fast Fourier Transformation) of the DSP (Digital Signal Processor) with the current BitClockDelay (BCD) value in the EEPROM of the read/write device.

This function suppresses up to two harmonic electromagnetic disturbers in the RF Band of the receiver (105 kHz to 145 kHz), e.g. from computers or monitors. Use this function as often as new RF background noise arises near the SIM antenna.

**Note:**
As the answer to this command appears before the FFT is ready (duration of FFT is approximately 110 ms), the host program has to wait at least 50 ms until sending the next transponder command.

**C-Function:**
void proloc_StartFFT (void);

**Header-File:**
PROLIB6.H

**Serial protocol:**

<table>
<thead>
<tr>
<th>Host - SIM</th>
<th>SIM - Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02 'F' 0x44</td>
<td>0x02 Status BCC</td>
</tr>
</tbody>
</table>

**Status:**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>no error</td>
</tr>
<tr>
<td>-1</td>
<td>SERIAL ERROR</td>
</tr>
<tr>
<td>-8</td>
<td>ACKNOWLEDGEMENT ERROR</td>
</tr>
</tbody>
</table>

The DSP did not send a correct acknowledge.
The error leads to a reset of the read/write device.
3.7.37 ReadBCD

This SIM command reads the BCD (BitClockDelay) value from the EEPROM of the read/write device. This value adjusts the timing of the read/write device in accordance to the connected antenna.

C-Function:    void proloc_ReadBCD (BYTE_T *bitclockdata);
Header-File:   PROLIB6.H

Serial protocol:

| Host - SIM | 0x02 | 'f' | BCC |
| SIM - Host | 0x03 | Status | bitclockdata | BCC |

| bitclockdata: |
| 7 6 5 4 3 2 1 0 |
| Bit 3 Bit 2 Bit 1 Bit 0 0 0 0 0 |

bitclockdelay: 0 to 15 dec possible

Status: 0 = no error
 -1 = SERIAL ERROR
3.7.38 SetBCD

This SIM command effects that a new BCD (BitClockDelay) value is passed to the DSP (Digital Signal Processor) and written into the EEPROM of the read/write device. This value adjusts the timing of the read/write device in accordance to the connected antenna.

A new adjustment may be necessary whenever a new type of antenna is connected to the read/write device.

Bits 4 to 7 of -bitclockdata- represent the BCD value. If Bit 3 of -bitclockdata- is set to 0, a FFT (Fast Fourier Transformation) of the DSP with the new BCD value is started in addition.

Standard value stored in the EEPROM:

\[
\text{bitclockdata} = 0x90
\]

C-Function: void proloc_SetBCD (BYTE_T bitclockdata);

Header-File: PROLIB6.H

Serial protocol:

**Host - SIM**

| 0x03 | 'F' | bitclockdata | BCC |

**bitclockdata:**

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>bitclockdelay Bit 3</td>
<td>bitclockdelay Bit 2</td>
<td>bitclockdelay Bit 1</td>
<td>bitclockdelay Bit 0</td>
<td>mode</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- **bitclockdelay:** 0 to 15 possible
- **mode:**
  - 0 = start a FFT
  - 1 = no FFT

**SIM - Host**

| 0x02 | Status | BCC |

**Status:**

| 0 | = no error  
| -1 | = SERIAL ERROR  
| -8 | = ACKNOWLEDGEMENT ERROR

The DSP did not send a correct acknowledge. The error leads to a reset of the read/write device.
3.7.39 SetBCDOffset

This SIM command writes a new BCD (BitClockDelay) -Offset value into the EEPROM of the read/write device. This value adjusts the difference of the timing between MDS F4xx and MDS F2xx transponders.

**Note:**
It is not necessary to use this command when working with SIM because EEPROM is already initialized to following standard value:

\[ bcd\_offset = 5 \]

**C-Function:**
void proloc_SetBCDOffset (BYTE_T bcd_offset);

**Header-File:**
PROLBLT6.H

**Serial protocol:**

**Host - SIM**

| 0x03 | 0xA4 | bcd\_offset | BCC |

**SIM - Host**

| 0x02 | Status | BCC |

Status:

0 = no error  
-1 = SERIAL ERROR

3.7.40 ReadLRStatus

This SIM command reads the antenna overload bit. In case of broken or badly detuned antenna the overload bit is high.

**C-Function:**
void proloc_ReadLRStatus (void);

**Header-File:**
PROLIB6.H

**Serial protocol:**

**Host - SIM**

| 0x02 | ‘r’ | 0x70 |

**SIM - Host**

| 0x02 | Status | BCC |

Status:

0 = no error  
-1 = SERIAL ERROR  
-20 = ANTENNA OVERLOAD
### 3.7.41 SetPowerDown

This command turns the SIM into Standby Mode

The byte -mode- is set to zero for Standby Mode. To activate the amplifier again this byte must be set to one

By default the read/write device is in Active Mode.

**C-Function:**

```c
void proloc_SetPowerDown (BYTE_T mode);
```

**Header-File:**

PROLIB6.H

**Serial protocol:**

**Host - SIM**

<table>
<thead>
<tr>
<th>0x03</th>
<th>'D'</th>
<th>mode</th>
<th>BCC</th>
</tr>
</thead>
</table>

**mode:**

- 0x00 = Standby-Mode
- 0x01 = Active Mode

**SIM - Host**

<table>
<thead>
<tr>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

**Status:**

- 0 = no error
- -1 = SERIAL ERROR
3.7.42 ReadAllPage

Reads one page of all MDS F4xx transponders in the active antenna field.

Note:
This command was developed for a special project requiring a special Reader-Hardware and -Software.

C-Function: void proloc_ReadAllPage (BYTE_T mode, BYTE_T pagenr, char *data,
WORD_T *data_len);

Header-File: PROLVEG6.H

Caution:
The size of the buffer for read data has to be dimensioned big enough by the user. For further information see Header File PROLVEG6.h.

Serial protocol:

Host - SIM

<table>
<thead>
<tr>
<th>0x04</th>
<th>0x98</th>
<th>mode</th>
<th>pagenr</th>
<th>BCC</th>
</tr>
</thead>
</table>

mode: Bit0 = 0: use Key A for Authentication
       Bit0 = 1: use Key B for Authentication
       Bit1 = 0: Plain (without Authentication)
       Bit1 = 1: Crypto (with Authentication)
       Bits 2-7: must be zero

pagenr: page number

SIM - Host

<table>
<thead>
<tr>
<th>0x06</th>
<th>Status</th>
<th>data[0]</th>
<th>.......</th>
<th>data[3]</th>
<th>BCC</th>
</tr>
</thead>
</table>

An answer string includes one page of each data carrier.

(n+1) strings are transmitted (n = number of data carriers). The last string contains the last error-condition.

Status: 0 = no error
         -1 = SERIAL ERROR
         -3 = NOTAG
         -7 = AUTHENTICATION ERROR
         -8 = ACKNOWLEDGEMENT ERROR
         -9 = CRYPTOBLOCK NOT INIT
3.7.43 GetDSpVersion

This command retrieves the version number of the DSP-software.

Note:
This command was developed for a special project requiring a special Reader-Hardware and Software.

C-Function:    void proloc_GetDspVersion (char *data);
Header-File:   PROLVEG6.H

Serial protocol:

<table>
<thead>
<tr>
<th>Host - SIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIM - Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x12</td>
</tr>
</tbody>
</table>

data[0] to data[15]: Version
- bytes with even index: ASCII
- bytes with odd index: 0

Status:
- 0 = no error
- -1 = SERIAL ERROR
3.7.44 KeyInitMode

To be able to personalize the read/write device it is necessary to enter a special mode, the KeyInit Mode.

The password (is different from Keys or Logdata) ensures that none but authorized persons are able to enter the KeyInitMode.

Note:
After the successful execution of this command (answer sent with Operating Mode BCC calculation) the read/write device enters the KeyInit Mode and BCC calculation changes.

The read/write device changes BCC calculation automatically. On the host system the user is responsible for the new BCC calculation. The C-Library provides the function proloc_SetBCCMode() .

C-Function: void proloc_KeyInitMode (DWORD_T password);
Header-File: PROLIB6.H

Serial protocol:

Host - SIM

<table>
<thead>
<tr>
<th>0x06</th>
<th>'K'</th>
<th>PW0</th>
<th>PW1</th>
<th>PW2</th>
<th>PW3</th>
<th>BCC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>/&lt;--</td>
<td>Password</td>
<td>--&gt;/</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SIM - Host

<table>
<thead>
<tr>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

Status:

0   = no error
-1   = SERIAL ERROR
-11   = EEPROM WRONG OLD DATA

The password was incorrect. The read/write device remains in Operating Mode.
3.8 Command Description for KeyInit Mode

The KeyInit Mode is a mode of the reader for using a set of personalization commands as described in the following (See also Chapter 5 "Personalization").

In this mode the BCC is computed by adding all block data bytes including Block Length. The least significant eight bits are used as BCC.

The command KeyInitMode is used to get from Operating Mode to KeyInit Mode.

Exit of KeyInit Mode is done by the command Reset or by a failing WriteEEPROM, WriteSecret_LT or WriteControl_LT.

3.8.1 Reset

This command switches the read/write device back to the Operating Mode.

Note:

- After the successful execution of this command (answer with KeyInit Mode BCC calculation) the read/write device enters the Operating Mode and BCC calculation changes.
- In Operating Mode the same command Reset (different BCC calculation) has a different functionality.

The read/write device changes BCC calculation automatically. On the host system the user is responsible for the new BCC calculation. The C-Library provides the function proloc_SetBCCMode().

C-Function: void proloc_Reset (void);
Header-File: PROLIB6.H

Serial protocol:

*Host - SIM*

| 0x02 | R | 0x54 |

*SIM - Host*

| 0x02 | Status | BCC |

Status:

- 0 = no error
- -1 = SERIAL ERROR
### 3.8.2 WriteSerNum

Writes a 11 byte serial number into the EEPROM of the SIM.

**Note:**
The serial number in Core Module of the SIM is already fixed and write-protected at delivery.

**C-Function:**
```c
void proloc_WriteSerNum (char *snr);
```

**Header-File:**
```c
PROLIB6.H
```

**Serial protocol:**

**Host - SIM**
```
0x0D  's'  Snr[0]  .....  Snr[10]  BCC
```

**SIM - Host**
```
0x02  Status  BCC
```

**Status:**
- 0 = no error
- -1 = SERIAL ERROR (in this case the error can also mean, that the read/write device is not in the KeyInit Mode!)
- -12 = EEPROM WRITE PROTECTED

If any error occurs KeyInit Mode is exited immediately.
3.8.3 ReadEEPROM

This command reads personalization data (4 data bytes) from the EEPROM of the read/write device.

Access rights are verified automatically by the read/write device before this command is executed. If a Read command is not permitted, Status is set to -13 (EEPROM READ PROTECTED).

**C-Function:**

```c
void proloc_ReadEEPROM (BYTE_T num, DWORD_T *data);
```

**Header-File:**

PROLIB6.H

**Serial protocol:**

*Host - SIM*

| 0x03 | ‘X’ | num | BCC |

`num:` defines which personalization data is to be read

- 0x00 = Password
- 0x01 = Key A
- 0x02 = Key B
- 0x03 = Logdata 0A
- 0x04 = Logdata 0B
- 0x05 = Logdata 1A
- 0x06 = Logdata 1B

*SIM - Host*

| 0x06 | Status | data[0] | ....... | data[3] | BCC |

`Status:`

- 0 = no error
- -1 = SERIAL ERROR (in this case the error can also mean, that the read/write device is not in the KeyInit Mode!)
- -12 = EEPROM WRITE PROTECTED
  (The read/write device remains in KeyInit-Modus.)
3.8.4 WriteEEPROM

This command writes new personalization data (4 data bytes) into the EEPROM of the read/write device.

This command requires the old data to be transmitted as well, which means that data can only be changed if the user knows the old written data.

Access rights and conformity between the sent old data and the stored data are verified before command execution.

**C-Function:**

```c
void proloc_WriteEEPROM (BYTE_T num, DWORD_T od,
                         DWORD_T nd);
```

**Header-File:** PROLIB6.H

**Serial protocol:**

```
<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>
```

- **num:** defines which personalization data is to be written
  - 0x00 = Password
  - 0x01 = Key A
  - 0x02 = Key B
  - 0x03 = Logdata 0A
  - 0x04 = Logdata 0B
  - 0x05 = Logdata 1A
  - 0x06 = Logdata 1B

- **OD[0] to OD[3]:** Old data
- **ND[0] to ND[3]:** New data to be written

**SIM - Host**

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td>Status</td>
<td>BCC</td>
</tr>
</tbody>
</table>
```

- **Status:**
  - 0 = no error
  - -1 = SERIAL ERROR ((in this case the error can also mean, that the read/write device is not in the KeyInit Model!)
  - -11 = EEPROM WRONG OLD DATA
  - -12 = EEPROM WRITE PROTECTED

If any error occurs KeyInit Mode is exited immediately.
3.8.5 ReadControl

With this command you can read the two control bytes Control_RW and Control_WO from the EEPROM of a read/write device.

**C-Function:** void proloc_ReadControl (BYTE_T *data);

**Header-File:** PROLIB6.H

**Serial protocol:**

*Host - SIM*

| 0x02 | 'C' | BCC |

*SIM - Host*

| 0x04 | Status | data[0] | data[1] | BCC |

*data[0]:* Control_RW; see Chapter 5 "Personalization"

*data[1]:* Control_WO; see Chapter 5 "Personalization"

**Status:**

0 = no error  
-1 = SERIAL ERROR (in this case the error can also mean, that the read/write device is not in the KeyInit Mode!)
3.8.6 WriteControl

This command writes a new value to the control bytes Control_RW and Control_WO into the EEPROM of the read/write device.

Initial values stored in the EEPROM of a delivered read/write device:
- Control_RW = 0x7F
- Control_WO = 0xFF

Note:
Once a bit in Control_RW or Control_WO has been set to ‘0’ it is impossible to change it back to one. We strongly recommend to read Chapter 5 "Personalization" carefully before using this command.

C-Function:     void proloc_WriteControl (BYTE_T control_rw, BYTE_T control_wo);
Header-File:    PROLIB6.H

Serial protocol:

Host - SIM

| 0x04 | 'c' | Control_RW | Control_WO | BCC  |

Control_RW: see Chapter 5 "Personalization"
Control_WO: see Chapter 5 "Personalization"

SIM - Host

| 0x02 | Status | BCC |

Status:
- 0 = no error
- -1 = SERIAL ERROR (in this case the error can also mean, that the read/write device is not in the KeyInit Mode)
3.8.7 ReadSecret_LT

This command reads MDS F2xx personalization data (4 data bytes) from the EEPROM of the read/write device.

Access rights are verified automatically by the read/write device before this command is executed. If a Read command is not permitted, Status is set to -13 (EEPROM READ PROTECTED).

C-Function:     void proloc_ReadSecret_LT (BYTE_T num, DWORD_T *data);
Header-File:    PROLBLT6.H

Serial protocol:

**Host - SIM**

```
0x03 'V' num BCC
```

*num:* defines which information is to be read

- 0x00 = Key LOW
- 0x01 = Key HIGH
- 0x02 = Password TAG
- 0x03 = Password RWD

**SIM - Host**

```
0x06 Status data[0] ...... data[3] BCC
```

*Status:*  

- 0 = no error  
- -1 = SERIAL ERROR (in this case the error can also mean, that the read/write device is not in the KeyInit Mode)!  
- -13 = EEPROM read protected.  

(The read/write device remains in KeyInit Mode.)
3.8.8 WriteSecret_LT

This command writes new MDS F2xx personalization data (4 data bytes) into the EEPROM of the read/write device. This command requires the old data to be transmitted as well, which means that data can only be changed if the user knows the old written data.

Access rights and conformity between the sent old data and the stored data are verified before command execution.

C-Function: void proloc_WriteSecret_LT (BYTE_T num, DWORD_T od, DWORD_T nd);
Header-File: PROLBLT6.H

Serial protocol:

**Host - SIM**

<table>
<thead>
<tr>
<th>7</th>
<th>0</th>
<th>-----</th>
<th>31</th>
<th>24</th>
<th>7</th>
<th>0</th>
<th>-----</th>
<th>31</th>
<th>24</th>
</tr>
</thead>
</table>

**num:** defines which information is to be written
0x00 = Key LOW
0x01 = Key HIGH
0x02 = Password TAG
0x03 = Password RWD

**OD[0] to OD[3]:** Old data
**ND[0] to ND[3]:** New data to be written

**SIM - Host**

<table>
<thead>
<tr>
<th>0x02</th>
<th>Status</th>
<th>BCC</th>
</tr>
</thead>
</table>

**Status:**
0 = no error
-1 = SERIAL ERROR ((in this case the error can also mean, that the read/write device is not in the KeyInit Mode!)
-11 = EEPROM WRONG OLD DATA
-12 = EEPROM WRITE PROTECTED

If any error occurs, KeyInit Mode is exited immediately.
3.8.9 ReadControl_LT

With this command you read the control byte Control_LT from the EEPROM of the read/write device. Control_LT is related to the MDS F2xx transponder.

**C-Function:** void proloc_ReadControl_LT (BYTE_T *data);

**Header-File:** PROLBLT6.H

**Serial protocol:**

*Host - SIM*

| 0x02 | 0x90 | BCC |

*SIM - Host*

| 0x03 | Status | data[0] | BCC |

**data[0]:** Control_LT; see Chapter 5 “Personalization”

**Status:**

- 0 = no error
- -1 = SERIAL ERROR (in this case the error can also mean, that the read/write device is not in the KeyInit Mode!)
3.8.10 WriteControl_LT

This command writes a new value to the control byte Control_LT into the EEPROM of the read/write device. Control_LT is related to the MDS F2xx transponder.

Initial value stored in the EEPROM of a delivered read/write device:
Control_LT = 0xFF

Note:
Once a bit in Control_LT has been set to ‘0’ it is impossible to change it back to one. We strongly recommend to read Chapter 5 “Personalization” carefully before using this command.

C-Function: void proloc_WriteControl_LT (BYTE_T control_lt);
Header-File: PROLBT6.H

Serial protocol:

Host - SIM

| 0x03 | 0x91 | Control_LT | BCC |

Control_LT: see Chapter 5 “Personalization”

SIM - Host

| 0x02 | Status | BCC |

Status:
0 = no error
-1 = SERIAL ERROR (in this case the error can also mean, that the read/write device is not in the KeyInit Mode!)
-12 = EEPROM WRITE PROTECTED

If any error occurs KeyInit Mode is exited immediately.
3.9 Examples to Access MDS F4xx Transponders

In the following please find examples of read/write cycles both for plain and encrypted access in order to illustrate the command sequence.

3.9.1 SIM: Anticollision Cycle

In case of several transponders in the reading area of the read/write device the GetSnr command indicates this by the more byte. To select one of these transponders for following read or write operations an anticollision cycle must be executed as described in the following flow chart.

*) In case you want to access the same tags for several times.
3.9.2 Proximity/SIM: Read Plain

GetSnr
Reads the serial number of a transponder in the communication field of the antenna. Use C-Function `proloc_GetSnr` or `proloc_GetSnr_Adv`.

SelectSnr
Selects (prepares) the transponder for a following read process. Use C-Function `proloc_SelectSnr` or `proloc_SelectLast`.

Read (Plain)
Reads a transponder. Use C-Function `proloc_ReadPage` or `proloc_ReadBlock`.

HaltSelected
Mutes the just treated transponder.

3.9.3 Proximity/SIM: Write Plain

GetSnr
SelectSnr
Write (Plain)
Writes data to a transponder. Use C-Function `proloc_WritePage` or `proloc_WriteBlock`.

Read (Plain)
To substantially increase the data reliability we strictly recommend to read the previously written data (read after write). Use C-Function `proloc_ReadPage` or `proloc_ReadBlock`.

HaltSelected
Mutes the just treated transponder.

3.9.4 Proximity/SIM: Read Crypto

GetSnr
SelectSnr
MutualAuthent
Carries out the mutual authentication of the transponder and the read/write device.

Read (Crypto)
Use C-Function `proloc_ReadPage` or `proloc_ReadBlock`.

HaltSelected

3.9.5 Proximity/SIM: Write Crypto

GetSnr
SelectSnr
MutualAuthent
Carries out the mutual authentication of the transponder and the read/write device.

Write (Crypto)
Use C-Function `proloc_WritePage` or `proloc_WriteBlock`.

Read (Crypto)
To substantially increase the data reliability we strictly recommend to read the previously written data (read after write). Use C-Function `proloc_ReadPage` or `proloc_ReadBlock`.

HaltSelected
3.10 Examples to Access MDS F2xx Transponders

3.10.1 Proximity/SIM: Read

GetSnr_LT  
Reads the serial number of a transponder in the communication field of the antenna. There is a parameter in the C-Function proloc_GetSnr_LT to specify whether you want to access a transponder in Password Mode or in Crypto Mode. If the response of the read/write device includes „no error” the transponder is selected and ready for read or write accesses.

ReadPage_LT  
Reads a transponder. Use C-Function proloc_ReadPage_LT.

HaltSelected_LT  
Schaltet den gerade behandelten Transponder stumm.

3.10.2 Proximity/SIM: Write

GetSnr_LT  
Reads the serial number of a transponder in the communication field of the antenna. There is a parameter in the C-Function proloc_GetSnr_LT to specify whether you want to access a transponder in Password Mode or in Crypto Mode. If the response of the read/write device includes „no error” the transponder is selected and ready for read or write accesses.

WritePage_LT  
Writes to the transponder.
Use C-Function proloc_WritePage_LT.

ReadPage_LT  
Reads from the transponder to verify if the write process was successful.
Use C-Function proloc_ReadPage_LT. If the response of the read/write device (to this first command after WritePage_LT) includes an error-condition, start from the beginning of the Write-Sequence again!

HaltSelected_LT  
Mutes the just treated transponder.
4 Transponders

4.1 MDS F4xx Transponders

4.1.1 Memory Organization

The 2 kBit EEPROM memory on the transponder is divided into 16 blocks. Every block consists of 4 pages. A page is the smallest access unit. Every page consists of 4 bytes (at 8 bits). Block access is only available for Blocks 8-15, page access is available for Pages 1-63. The serial number, and key A and key B cannot be read with ReadPage.

Areas (or settings) with light dark background may be configured by the OEM client using the Configuration Page (Page 1).

Memory locations marked with "secret" can only be accessed after a mutual authentication. An enciphered data communication is used in that area.

Memory locations marked with "public" can be accessed without mutual authentication, no encryption is used.

Block 0 includes the unique serial number (programmed during the production process), the Configuration Page (configuration of the memory area) and the keys, Block 1 includes the logdata.

Blocks 4 to 7 can be used either as secret or public areas (configurable), and Blocks 2 to 7 either as read / write or read only areas (configurable). You can also modify keys and logdata and prevent them from being accessed.

Finally the Configuration Page itself can be set to read only.
It is extremely important to be particularly careful when using the Configuration Page (it only can be set to read only once!), keys and logdata as you can lose access to the secret area on the transponder if you make a mistake.

**Note:**
Changing of the Configuration Page (Page 1), Keys and Logdata must be done in secure environment. The transponder must not be moved out of the communication field of the antenna during programming! We recommend to put the transponder close to the antenna (zero-distance) and not to remove it during programming.

**Note:**
Access to data areas classified as secret for MDS F4xx can only be performed in password mode. Access to a public page is suppressed in password mode with an error message. The mode is automatically reset to Crypto. Password mode must be activated again to continue accessing protected pages.

**Note:**
Writing the configuration with WritePage() to an MDS F4xx data carrier may cause the entire data carrier to be lost. If the data carrier is being written when the antenna malfunctions, page (4 bytes) is set to FF on the data carrier (FORMAT on EPROM), and the subsequent write-access with the transferred data is no longer performed. Several bits are then set to 1. See chapter 4.1.3.2 on configuration. The data carrier can now only be recognized with GetSnr(). All additional accesses cause an error.

**Note:**
If an error occurs while communication is taking place, the affected data carrier is no longer selected. The data carrier must then be selected again with GetSnr() and SelectLast().

### 4.1.2 Anticollision

Anticollision Mode in SIM applications including MDS F4xx transponders permits you to process several transponders simultaneously. Theoretically up to $2^{32}$ transponders can be processed simultaneously. In practice this number is limited, because of the mutual influence of the transponders - they detune each other, if there are too many too close to each other.

In Proximity applications only one transponder is handled even if there are several transponders within the communication field of the antenna. In this case either no communication takes place or the “stronger” or closer transponder takes over.

By muting a selected transponder (HALT Mode) another transponder that is to be found in the communication field of the antenna can be recognized.
4.1.3 Operation-Modes and Configuration

4.1.3.1 Modes of Operation

The MDS F4xx can be operated in following 2 modes that cannot be configured using the Configuration Page, but via host-commands.

Standard Protocol Mode:
This mode is activated using the command GetSnr.

Advanced Protocol Mode:
This mode is activated using the command GetSnr_Adv.
Advanced Protocol Mode uses, above all, an additional Cyclic Redundancy Check (CRC) for read operations.

4.1.3.2 Configuration

The Configuration Page consists of 4 Configuration Bytes, the first two bytes are used for configuration, the other two bytes can be used freely.

Konfigurations-Byte 0 und 1:
The Configuration Page consists of 4 Configuration Bytes, the first two bytes are used for configuration, the other two bytes can be used freely.

The bitmaps in Configuration Bytes 0 and 1 determine the configuration of the memory, i.e. they define which area is secret or public, r/w, ro, wo or neither read nor write.
You can allocate and write the Configuration Page until it is locked (Bit 4 of Configuration Byte 1 is set to '0').
After that these bytes are read only bytes and the configuration of the transponder memory cannot be changed any more.

Note:
Once set to read only the Configuration Page cannot be changed back to r/w again (transponder is hardware protected)!
If you change the configuration, you have to place the transponder directly on the antenna or hold it directly to it (0-distance)! In order to avoid any errors do not move the transponder during this write process and be sure that you are in a safe environment without electrical noise.

Configuration Bytes 2 and 3:
These two bytes, too, are set to read only by the OEM Lock Bit (Configuration Byte 1 / Bit 4 = "0"). Considering that fact you can use these two bytes freely. They will not affect memory configuration. For example, the OEM client can put his own OEM serial number here.

Explanations of abbreviations used:

- r/w read and write
- ro read only
- wo write only
- 0 neither read nor write
Configuration-Byte 0:

<table>
<thead>
<tr>
<th></th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block 7:</td>
<td>&quot;1&quot; = r/w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = ro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block 6:</td>
<td>&quot;1&quot; = r/w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = ro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block 5:</td>
<td>&quot;1&quot; = r/w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = ro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block 4:</td>
<td>&quot;1&quot; = r/w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = ro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block 3:</td>
<td>&quot;1&quot; = r/w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = ro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block 2:</td>
<td>&quot;1&quot; = r/w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = ro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key A and Key B:</td>
<td>&quot;1&quot; = wo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logdata in Block 1:</td>
<td>&quot;1&quot; = r/w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Configurations-Byte 0/Bit 7:

Bit = '1': Logdata can be read and written to.
Bit = '0': Logdata cannot be accessed.

This bit can be set or reset until Bit 4 of Configuration Byte 1 is set to '0'.

For further information on Logdata and Keys see chapter 5 "Personalization".

Configurations-Byte 0/Bit 6:

Bit = '1': Keys can only be written to.
Bit = '0': Keys cannot be accessed.

This bit can be set or reset until Bit 4 of Configuration Byte 1 is set to '0'.

For further information on Logdata and Keys see Chapter 5 "Personalization".

Configuration-Bytes 0/Bits 0 to 5:

If one of these Configuration Bits reads '1', the corresponding block of the transponder can be read and written.
If the bit is set to '0', the corresponding block can only be read.

Within one block the configuration is always identical, that means either all 4 pages are read/write or all of them are read only.

These bits can be set or reset until Bit 4 of Configuration Byte 1 is set to '0'
**Configuration-Byte 1:**

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
</table>

Access type for Blocks 4 to 7: "0" = secret  
"1" = public  
reserved  
reserved  
reserved  
reserved  
reserved  
reserved  
OEM-Lock-Bit: "0" = Configuration-Page is ro  
"1" = Configuration-Page is r/w  
reserved  
reserved  
reserved  
reserved  

**Configuration-Byte 1/Bit 5 to 7:**
These three bits are reserved.

**Note:**
When writing a new value to byte Configuration Byte 1, bit positions marked as "reserved" must not be altered. To meet that condition read the current Configuration Byte 1 value and mask in your new values for bit positions you are allowed to change.

**Configurations-Byte 1/Bit 4:**
Bit= '1': Configuration Page can be read and written to.  
Bit= '0': Configuration Page can only be read. This process is irreversible !!

**Note:**
Do not set Bit 4 of Configuration Byte 1 to '0' before having written the final data into the Configuration Page of the transponder.

**Configuration-Byte 1/Bits 1 to 3:**
These three bits are reserved

**Note:**
When writing a new value to byte Configuration Byte 1, bit positions marked as "reserved" must not be altered. To meet that condition read the current Configuration Byte 1 value and mask in your new values for bit positions you are allowed to change.

**Configuration-Byte 1/Bit 0:**
Bit= '0': Access type for Blocks 4 to 7 is SECRET.  
Bit= '1': Access type for Blocks 4 to 7 is PUBLIC.  
This bit can be set or reset until Bit 4 of Configuration Byte 1 is set to '0'.
4.1.4 Configuration of Delivered MDS F4xx Transponders

MDS F4xx transponders are delivered with the following configuration by Siemens:

**Unique Serial Number:**

| Serial Number: | Read Only | - | fixed |

**Configurations-Byte 0:**

- Logdata: ‘1’ = r/w - can be changed
- Key A, Key B: ‘1’ = wo - can be changed
- Block 2 to 7: ‘1’ = r/w - can be changed

**Configuration-Byte 1:**

- OEM-Lock-Bit: ‘1’ = Configuration Page is r/w - can be changed
- Block 4 to 7: ‘1’ = public - can be changed

**Value for Transport Keys, Transport Logdata:**

0x00000000

**Recommendation:**

Before delivering transponders to end users, the Configuration Page should be set to read only (Configuration Byte 1/Bit 4 = ‘0’).
4.2 MDS F2xx Transponders

4.2.1 Memory Organization

The memory of the transponder consists of 256 bits EEPROM and is organized in 8 pages with 32 bits each.

Depending on the operation-mode the EEPROM is organized differently.

**Crypto-Mode:**

<table>
<thead>
<tr>
<th>Page</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Serial Number</td>
</tr>
<tr>
<td>1</td>
<td>32 bit &quot;Key LOW&quot;</td>
</tr>
<tr>
<td>2</td>
<td>16 bit &quot;Key HIGH&quot;</td>
</tr>
<tr>
<td>3</td>
<td>8 bit Config., 24 Bit Password TAG</td>
</tr>
<tr>
<td>4</td>
<td>read/write page</td>
</tr>
<tr>
<td>5</td>
<td>read/write page</td>
</tr>
<tr>
<td>6</td>
<td>read/write page</td>
</tr>
<tr>
<td>7</td>
<td>read/write page</td>
</tr>
</tbody>
</table>

**Password Mode:**

<table>
<thead>
<tr>
<th>Page</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Serial Number</td>
</tr>
<tr>
<td>1</td>
<td>Password RWD</td>
</tr>
<tr>
<td>2</td>
<td>reserved</td>
</tr>
<tr>
<td>3</td>
<td>8 bit Config., 24 bit Password TAG</td>
</tr>
<tr>
<td>4</td>
<td>read/write page</td>
</tr>
<tr>
<td>5</td>
<td>read/write page</td>
</tr>
<tr>
<td>6</td>
<td>read/write page</td>
</tr>
<tr>
<td>7</td>
<td>read/write page</td>
</tr>
</tbody>
</table>
4.2.2 Operation-Modes and Configuration

With the Configuration Byte the operation-mode and the access rights to the memory can be selected. During Power-Up of the transponder the Configuration Byte is read from the transponder’s EEPROM.

If you change keys, passwords or configuration, you have to place the transponder directly on the antenna or hold it directly to it (0-distance)! In order to avoid any errors do not move the transponder during this write process and be sure that you are in a safe environment without electrical noise.

4.2.2.1 Modes of Operation

The MDS F2xx can be operated in several modes.

**Crypto-Mode:**
Mode for writing or reading the transponder with encrypted data transmission.

**Password Mode:**
Mode for writing or reading the transponder with plain data transmission.

**Public Mode A (Manchester):**
Read-only mode emulating Siemens MDS F2xx transponders.
The 64 bits of the user Pages 4 and 5 are cyclically transmitted to the base station.
See Chapter 3 "Communication Read/Write Device-Host" (command description for Read_F1) for an example of allocating Pages 4 and 5 with MDS F1xx-compatible data.

**Public Mode B (Biphase):**
Read-only mode according to ISO standards 11784 and 11785 for animal identification.
The 128 bits of the user Pages 4 to 7 are cyclically transmitted to the base station.
See Chapter 3 "Communication Read/Write Device-Host" (command description for ReadPublicB_LT) for an example of allocating Pages 4 to 7 for animal identification.

**Public Mode C (Biphase):**
Read-only mode emulating the read operation of the PCF793X (with a slightly different Program Mode Check).
In the Public Mode C the 128 bits of the user Pages 4 to 7 are cyclically transmitted to the base station.
4.2.2.2 Configuration

The Configuration Byte is represented by the first 8 bits of Page 3 of the transponder memory

<table>
<thead>
<tr>
<th>Configuration-Byte:</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Version</th>
<th>Coding</th>
<th>Coding in RW32-Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Public Mode B</td>
<td>Biphase</td>
<td>depending on Bit 0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Public Mode A</td>
<td>Manchester</td>
<td>depending on Bit 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Public Mode C</td>
<td>Biphase</td>
<td>depending on Bit 0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>MDS F2xx</td>
<td>depending on Bit 0</td>
<td>depending on Bit 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0: password mode</th>
<th>1: Crypto-mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PAGE 6 und 7 read/write</td>
<td>PAGE 6 und 7 read only</td>
</tr>
<tr>
<td>1</td>
<td>PAGE 4 und 5 read/write</td>
<td>PAGE 4 und 5 read only</td>
</tr>
</tbody>
</table>

THE SETTING OF THIS BIT IS OTP!

<table>
<thead>
<tr>
<th></th>
<th>0: PAGE 3 read/write</th>
<th>1: PAGE 3 read only; Configuration-Byte and Password TAG fixed</th>
</tr>
</thead>
</table>

THE SETTING OF THIS BIT IS OTP!

<table>
<thead>
<tr>
<th></th>
<th>0: PAGE 1 und 2 read/write</th>
<th>1: PAGE 1 no read/no write</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PAGE 2 read only (when transponder is in password mode)</td>
<td>PAGE 2 no read/no write (when transponder is in crypto mode)</td>
</tr>
</tbody>
</table>

Configuration Byte/Bit 6:

Bit= ‘0’: Page 3 is read/write
Bit= ‘1’: Page 3 can only be read. This process is irreversible!

Note
Do not set Bit 6 of the Configuration Byte to ‘1’ before having written the final data into Page 3 (including the Configuration Byte and Password TAG) of the transponder.

Configuration Byte/Bit 7:

Bit= ‘0’: Pages 1 and 2 are read/write
Bit= ‘1’: Page 3 can only be read. This process is irreversible!

Note:
Do not set Bit 7 of the Configuration Byte to ‘1’ before having written the final data into Pages 1 and 2 of the transponder.
4.2.3 Configuration of Delivered MDS F2xx Transponders

MDS F2xx transponders are delivered with the following configuration by Siemens

Unique Serial Number:

<table>
<thead>
<tr>
<th>Serial Number:</th>
<th>Read Only</th>
<th>-</th>
<th>fixed</th>
</tr>
</thead>
</table>

Configuration-Byte:

<table>
<thead>
<tr>
<th>0x06:</th>
<th>Password Mode (Manchester-Code)</th>
<th>-</th>
<th>can be changed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Page 6 and 7 r/w</td>
<td>-</td>
<td>can be changed</td>
</tr>
<tr>
<td></td>
<td>Page 4 and 5 r/w</td>
<td>-</td>
<td>can be changed</td>
</tr>
<tr>
<td></td>
<td>Page 3 r/w</td>
<td>-</td>
<td>can be changed</td>
</tr>
<tr>
<td></td>
<td>Page 1 and 2 r/w</td>
<td>-</td>
<td>can be changed</td>
</tr>
</tbody>
</table>

Values for Transport Passwords, Transport Keys:

<table>
<thead>
<tr>
<th>Password RWD:</th>
<th>0x4D494B52</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password TAG:</td>
<td>0xAA4854</td>
</tr>
<tr>
<td>Key Low:</td>
<td>0x4D494B52</td>
</tr>
<tr>
<td>Key High:</td>
<td>0x4F4E</td>
</tr>
</tbody>
</table>

Recommendation:

Before delivering transponders to end users, Pages 1 to 3 should be locked (set Configuration Byte/Bit 6 to ‘1’ for Page 3 and set Configuration Byte / Bit 7 to ‘1’ for Pages 1 and 2).

4.3 MDS F1xx Transponder

4.3.1 Memory Organization

In the 64 bit memory the unique 40 bit serial number of the transponder is stored as well as 24 bit header and parity bits. The data is read-only and cannot be changed
5 Personalization

5.1 Introduction

In order to profit from the full functionality of the MOBY F transponders, the read/write device has to support the transponder’s cryptographic feature.

This requires the use of some secret data (keys, logdata). The process of loading these data into the read/write device is called personalization. The same personalization procedure has to be carried out on your transponders.

5.2 Personalization Concept

To enable utmost security and flexibility Siemens worked out a personalization concept that shall be shortly described in the following:

The first stage is a test that is done by the producer respectively Siemens. Here the unique serial number is fixed and defined Transport Keys, Transport Logdata and Transport Passwords are pre-programmed.

In the next stage the customers program their own keys and passwords (to ensure that only persons who got the authorization from the customer are able to access secret data of the transponders) and configure the memory of the transponders. We recommend to lock sensitive areas, that means for example to prevent the possibility to change keys and passwords for the user.

In the last stage the user just reads from and writes to the memory of the transponders.

Note:

If you change these Transport Keys and Transport Logdata (and we strictly recommend to do so if you want to store security-sensitive data) in the course of system integration, you have to be extremely careful. Make sure you are in a safe environment while writing secret data to the transponder or the read/write device. This prevents possible listening in to the communication between HOST and read/write device.

All the security relevant data in the read/write device can be protected from read or write accesses using special serial commands.

Security relevant data for MDS F4xx transponders:

- Key information A and B
- Logdata 0A, 0B
- Logdata 1A, 1B

Security relevant data for MDS F4xx transponders:

- Key information
- Password TAG:
- Password RWD
The mechanism to protect security relevant data in the read/write device has 3 levels

Level 0: All security relevant data can be read and written.

Level 1: The data cannot be read any more. If you want to change an entry, you have to know the old value. Otherwise writing access will be denied.

Level 2: The internal data are locked and can neither be read nor written. At this level it is impossible for the user to change the stored data

You cannot reset levels, e.g. from level 2 to level 1. Once a security level has been chosen it becomes irreversible.

The functionality of this mechanism is based on the control bytes Control_RW (only for MDS F4xx), Control_WO (only for MDS F4xx) and Control_LT (only for MDS F2xx). All control bytes are located in EEPROM of the read/write device.

On delivery of a read/write device all bits of Control_RW, Control_WO and Control_LT are set to 1, except Bit 7 of CONTROL_RW (serial number).
Control_RW (only MDS F4xx):
Control_RW (located in EEPROM of the read/write device) controls read accesses to the following data:

0 = Read only, fixed

<table>
<thead>
<tr>
<th>Bit Number</th>
<th>Bit Name</th>
<th>Bit-Value = 1</th>
<th>Bit-Value = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Serial Number</td>
<td>-</td>
<td>read allowed</td>
</tr>
<tr>
<td>6</td>
<td>Logdata1B</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>5</td>
<td>Logdata1A</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>4</td>
<td>Logdata1A</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>3</td>
<td>Logdata1A</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>2</td>
<td>Key A</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>1</td>
<td>Key B</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>0</td>
<td>Password</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
</tbody>
</table>

Note:
You cannot change bits that have once been set to 0!
Control_WO (only MDS F4xx):
Control_WO (located in EEPROM of the read/write device) controls write accesses to the following data:

```
+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+

6 5 4 3 2 1 0
```

- Password
- Key A
- Key B
- Logdata 0A
- Logdata 0B
- Logdata 1A
- Logdata 1B

**Note:**
You cannot change bits that have once been set to 0!

<table>
<thead>
<tr>
<th>Bit Number</th>
<th>Bit Name</th>
<th>Bit-Value = 1</th>
<th>Bit-Value = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>Logdata1B</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>5</td>
<td>Logdata1A</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>4</td>
<td>Logdata1A</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>3</td>
<td>Logdata1A</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>2</td>
<td>Key A</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>1</td>
<td>Key B</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>0</td>
<td>Password</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
</tbody>
</table>
Control_LT (only MDS F2xx):
Control_LT (located in EEPROM of the read/write device) controls write accesses to the following data:

<table>
<thead>
<tr>
<th>Bit Number</th>
<th>Bit Name</th>
<th>Bit-Value = 1</th>
<th>Bit-Value = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Key RW/WO</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>6</td>
<td>Key WO/0</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>5</td>
<td>Password TAG RW/ WO</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>4</td>
<td>Password TAG WO/0</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>3</td>
<td>Password RWD RW/WO</td>
<td>read allowed</td>
<td>read prohibited</td>
</tr>
<tr>
<td>2</td>
<td>Password RWD WO/0</td>
<td>write allowed</td>
<td>read/write prohibited</td>
</tr>
<tr>
<td>1</td>
<td>check PW TAG *)</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>0</td>
<td>config. Lock</td>
<td>read/write of Control_LT allowed</td>
<td>read/write of Control_LT allowed</td>
</tr>
</tbody>
</table>

*) Note for Bit 1 of Control_LT:
If the MDS F2xx transponder is in Password or Crypto Mode and a GetSnr_LT command is processed, the incoming Password TAG of the transponder is checked whether it matches with the Password TAG of the reader. If it doesn’t, the read/write device transmits the error-message INCORRECT PASSWORD TAG to the host.
We recommend to activate "Check PW TAG" (set bit 1 to zero) in order to increase the security for GetSnr_LT and PollTags commands.
5.3 Personalization of MDS F4xx Transponders

In order to be able to read data from the secret area of a transponder, you have to carry out a procedure called authentication. To do this you need special data (keys). After transmitting the according command the authentication is automatically carried out by the read/write device.

5.3.1 Definition of Keys and Logdata

Keys are cryptographic codes, which determine data encryption during data transfer between read/write device and transponder.

Two keys (Key A and Key B) which you can use independently of each other, have been installed for security and flexibility reasons. The identity of either Key A or Key B on the read/write device and on the transponder is sufficient.

Logdata represent "passwords" needed to gain access to secret areas on the transponder. A pair of logdata is included with every cryptographic key (Key A and Key B). This logdata pair has to be identical both on the transponder and the read/write device.

Key A:

| Logdata 0 A | "Password" which the transponder sends to the read/write device and which is verified by the latter. |
| Logdata 1 A | "Password" which the read/write device sends to the transponder and which is checked for identity by the latter |

Key B:

| Logdata 0 B and Logdata 1 B | analogous to Key A |

The keys and logdata are predefined by Siemens by means of defined Transport Keys (both keys show the same bit map) and Transport Logdata (all logdata show the same bit map). They can be written to, which means that they can be changed.

Note:

Keys and Logdata only can be changed if their current values are known!

It is important that the following values are in accordance with each other, i.e. the respective data on the read/write device and on the transponder have to be identical pairs:

<table>
<thead>
<tr>
<th>on the read/write device</th>
<th>on the transponder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key A</td>
<td>⇔ Key A</td>
</tr>
<tr>
<td>Logdata 0A</td>
<td>⇔ Logdata 0A</td>
</tr>
<tr>
<td>Logdata 1A</td>
<td>⇔ Logdata 1A</td>
</tr>
<tr>
<td>Key B</td>
<td>⇔ Key B</td>
</tr>
<tr>
<td>Logdata 0B</td>
<td>⇔ Logdata 0B</td>
</tr>
<tr>
<td>Logdata 1B</td>
<td>⇔ Logdata 1B</td>
</tr>
</tbody>
</table>
5.3.2 Changing Keys and Logdata

You do not have to change keys and logdata in order to operate a system with the read/write device because access to the secret area of the transponder is possible with the Transport Keys and Transport Logdata. Nevertheless we strictly recommend to change these data to be sure no other person (and nobody of Siemens) than that who got the authorization from you are able to access the secret area of the transponder.

If you change keys and logdata, you have to place the transponder directly on the antenna or hold it directly to it (0-distance)! In order to avoid any errors do not move the transponder during this write process and be sure that you are in a safe environment without electrical noise.

5.3.2.1 Changing Keys

Please, note the order of the steps!!

1. Access the transponder (using the Transport Keys).
2. Change one key (e.g.: Key A) on the transponder, i.e., overwrite the corresponding page on the transponder (in this case Page 2) with the new key.
3. Change Key A on the read/write device to the new value.

Caution:

On the transponder the key can only be written, which means that you cannot call up the entry! Moreover, you need to know the old value if you want to change the key on the read/write device!

Only after carrying out correctly steps 1 through to 3 (execute a read-access test with the changed key to check it!) may the second key be changed following the steps described above. Conveniently you change both keys to the same value!

5.3.2.2 Incorrect Procedures Changing Keys

- You change both keys on the read/write device and then try to access the transponder. This is not possible because there is no identity between any of the keys on the transponder and the read/write device.

- You change only one key (e.g.: Key A) on the read/write device; the second key (in this example B) remains the Transport Key. Then you try again to access the transponder. This can be possible, only if your system works with both keys and checks one after the other, because one key (here it is Key B) on the transponder and the read/write device is still identical.
When you change a key, this does not mean that you also have to change the corresponding logdata and the other way round.

**Note:**
If neither Key A nor Key B of the transponder and the read/write device are identical, you cannot access the secret area on that transponder! Access to the plain area of the transponder (e.g. serial number) is possible in any case.

### 5.3.2.3 Changing Logdata

To change logdata use the same procedure as described for changing keys. Be careful to change them by pairs (on the read/write device and on the transponder):

1. Change, for example, Logdata 0A on the transponder (by overwriting Page 5).
2. Change Logdata 0A on the read/write device to the new value.
3. Change Logdata 1A on the transponder (by overwriting Page 6).
4. Change Logdata 1A on the read/write device to the new value.

**Again, you need to know the old values before they can be changed on the read/write device.**

For changing the logdata of a big number of tags we recommend to doing it in the same way as described in the former Chapter 5.3.2.1 "Changing Keys" in the note.

When you change a key, this does not mean that you also have to change the corresponding logdata and the other way round.
5.4 Personalization of MDS F2xx Transponders

5.4.1 Definition of Passwords and Keys

Keys are cryptographic codes, which determine data encryption during data transfer between read/write device and transponder. They are used to select a MDS F2xx transponder in Crypto Mode. The 16 bit KEY HIGH and 32 bit KEY LOW form one 48 bit key which has to be identical on both the transponder and the read/write device.

Passwords are needed to select a MDS F2xx transponder in Password Mode. There is one pair of passwords (Password TAG, Password RWD) which has to be identical both on the transponder and the read/write device.

Password TAG: Password that the transponder sends to the read/write device and which may be verified by the latter (depending on the configuration of the read/write device).

Password RWD: Password that the read/write device sends to the transponder and which is checked for identity by the latter.

The passwords and keys are predefined by Siemens by means of defined Transport Passwords and a Transport Key. They can be written to, which means that they can be changed.

Note:
Passwords and Keys only can be changed if their current values are known!

It is important that the following values are in accordance with each other, i.e. the respective data on the read/write device and on the transponder have to be identical pairs.

MDS F2xx in Password mode:

<table>
<thead>
<tr>
<th>on the read/write device</th>
<th>on the transponder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password RWD</td>
<td>⇔</td>
</tr>
<tr>
<td></td>
<td>Password RWD</td>
</tr>
</tbody>
</table>

as an option (depending on bit 1 of CONTROL_LT):

<table>
<thead>
<tr>
<th>on the read/write device</th>
<th>on the transponder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password TAG</td>
<td>⇔</td>
</tr>
<tr>
<td></td>
<td>Password TAG:</td>
</tr>
</tbody>
</table>

MDS F2xx in Crypto mode:

<table>
<thead>
<tr>
<th>on the read/write device</th>
<th>on the transponder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key LOW</td>
<td>⇔</td>
</tr>
<tr>
<td>Key HIGH</td>
<td>⇔</td>
</tr>
</tbody>
</table>

as an option (depending on bit 1 of CONTROL_LT):

<table>
<thead>
<tr>
<th>on the read/write device</th>
<th>on the transponder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password TAG</td>
<td>⇔</td>
</tr>
<tr>
<td></td>
<td>Password TAG:</td>
</tr>
</tbody>
</table>
5.4.2 Changing Passwords and Keys

You do not have to change passwords and keys in order to operate a system with the read/write device because access to the secret area of the transponder is possible with the Transport Passwords and Transport Keys. Nevertheless, we strictly recommend to change these data to be sure no other person (and nobody of Siemens) than that who got the authorization from you are able to access the secret area of the transponder.

If you change passwords and keys, you have to place the transponder directly on the antenna or hold it directly to it (0-distance)! In order to avoid any errors do not move the transponder during this write process and be sure that you are in a safe environment without electrical noise.

5.4.2.1 Changing Password RWD and Keys

Please, note the order of the steps!

1. Select the transponder.
2. If the transponder is in Password Mode, you only have to overwrite Page 1 (Password RWD). If the transponder is in Crypto Mode, you have to overwrite Page 1 (KEY LOW) and 2 upper bytes of Page 3 (KEY HIGH).
   It is recommended to execute a read-access test to check the changed values..
3. Change the corresponding data on the read/write device to the new values

⚠️ Caution:
You need to know the old value if you want to change the passwords or keys on the read/write device!

5.4.2.2 Incorrect Procedures Changing Password RWD and Keys

- You change values on the read/write device and then try to access the transponder. This is not possible because there is no identity between any of the keys on the transponder and the read/write device.

The same scenario applies if you first change values on the transponder but leave the corresponding values on the read/write device unchanged (transport key).
5.4.2.3 Changing Password TAG

To change Password TAG on MDS F2xx transponders in Password Mode or Crypto Mode use the same procedure as described for changing Password RWD and keys. Be careful to change them by pairs (on the read/write device and on the transponder):

1. Change Password TAG on the transponder. Password TAG and the Configuration Byte are located on Page 3 of the transponder. In order not to change the value of the Configuration Byte it is recommended to read Page 3 from the transponder. Byte 0 is left unchanged, and Bytes 1 to 3 are set to the new Password TAG value. Then Byte 0 to Byte 3 are written to the transponder.

2. Change Password TAG on the read/write device to the new value

Again, you need to know the old values before they can be changed on the read/write device.
6 Security Considerations

Developing the read/write device special consideration was given to aspects of security. The following items represent the fundamental framework of the security concept:

- cryptography
- mutual authentication
- password verification and
- Cyclic Redundancy Check (CRC)

6.1 Data Reliability

6.1.1 Data Stream between Read/Write Device and Transponder

**MDS F4xx transponders:**

All the commands and data transferred from the read/write device to the transponder are secured by Cyclic Redundancy Check (CRC).

Every data stream sent (commands, addresses, user data) from the read/write device to the transponder is checked for data errors by the transponder by means of an integrated 8-bit CRC generator.

The CRC is formed over commands and addresses or the plain data respectively and in the case of Crypto Mode it is also encrypted.

The generator polynomial of the transponder CRC generator reads:

\[ u^8 + u^4 + u^3 + u^2 + 1 = 0x1D \]

The CRC preassignment is: 0xFF

**MDS F2xx transponders:**

Every command sent from the read/write device to the transponder is checked for data errors by the transponder.

Standard commands transferred from the read/write device to the transponder are divided into two Bit Streams. The second Bit Stream is generated by inverting the bits of the first Bit Stream. This redundancy increases data security.

6.1.2 Checking User Data

Security of the data read from the transponder by the read/write device remains with the user for reasons of flexibility.

Therefore, you can choose flexible check sums and store them in the transponder memory together with the data. You can protect sensitive data better than less sensitive data, thus permitting optimised operation times.
6.2 Data Privacy

The use of cryptography (Stream Cipher), mutual authentication, and password verification prevents monitoring and copying the data channel. Therefore, the area of the transponder that only can be accessed enciphered is called "secret area."

To make use of cryptography you need secret data: keys (MDS F4xx and MDS F2xx transponders) and logdata (MDS F4xx transponders).

The transponders and the read/write device are provided with identical transport keys and transport logdata by Siemens so that you can start operating them right away.

In order to offer our OEM clients high flexibility, the configuration of the transponder memory, password, keys and logdata can be changed.

We strictly recommend to rigorously restrict these possibilities for the end customers (e.g. for MDS F4xx transponders by setting the Configuration Page to read only, setting password, keys and logdata to neither read nor write).