

# TWN4

## Simple Protocol

DocRev14, January 25, 2017



Elatec GmbH

# Contents

1 Simple Protocol . . . . .	9
1.1 Command . . . . .	9
1.2 Response . . . . .	9
1.3 Data Transmission . . . . .	10
1.3.1 ASCII . . . . .	10
1.3.2 Binary . . . . .	10
1.3.3 CRC . . . . .	10
1.3.4 Reference messages . . . . .	10
1.4 Data Types . . . . .	11
1.5 Commands . . . . .	11
1.5.1 API SYS . . . . .	11
1.5.1.1 Reset . . . . .	11
1.5.1.2 StartBootloader . . . . .	11
1.5.1.3 GetSysTicks . . . . .	12
1.5.1.4 GetVersionString . . . . .	12
1.5.1.5 GetUSBType . . . . .	12
1.5.1.6 GetDeviceType . . . . .	12
1.5.1.7 Sleep . . . . .	13
1.5.1.8 GetDeviceUID . . . . .	13
1.5.1.9 SetParameters . . . . .	13
1.5.1.10 GetLastError . . . . .	13
1.5.2 API IO . . . . .	14
1.5.2.1 WriteByte . . . . .	14
1.5.2.2 ReadByte . . . . .	14
1.5.2.3 TestEmpty . . . . .	14
1.5.2.4 TestFull . . . . .	15
1.5.2.5 GetBufferSize . . . . .	15
1.5.2.6 GetByteCount . . . . .	15
1.5.2.7 SetCOMParameters . . . . .	16
1.5.2.8 GetUSBDeviceState . . . . .	16
1.5.2.9 GetHostChannel . . . . .	16
1.5.2.10 USBRemoteWakeup . . . . .	16
1.5.2.11 WriteBytes . . . . .	17
1.5.2.12 ReadBytes . . . . .	17
1.5.3 API PERIPH . . . . .	17
1.5.3.1 GPIOConfigureOutputs . . . . .	17
1.5.3.2 GPIOConfigureInputs . . . . .	17
1.5.3.3 GPIOSetBits . . . . .	18
1.5.3.4 GPIOClearBits . . . . .	18
1.5.3.5 GPIOToggleBits . . . . .	18
1.5.3.6 GPIOBlinkBits . . . . .	18
1.5.3.7 GPIOTestBit . . . . .	19
1.5.3.8 Beep . . . . .	19
1.5.3.9 DiagLEDOOn . . . . .	19

1.5.3.10	DiagLEDOff . . . . .	19
1.5.3.11	DiagLEDToggle . . . . .	20
1.5.3.12	DiagLEDIsOn . . . . .	20
1.5.3.13	SendWiegand . . . . .	20
1.5.3.14	SendOmron . . . . .	20
1.5.4	API RF . . . . .	21
1.5.4.1	SearchTag . . . . .	21
1.5.4.2	SetRFOff . . . . .	21
1.5.4.3	SetTagTypes . . . . .	21
1.5.4.4	GetTagTypes . . . . .	21
1.5.4.5	GetSupportedTagTypes . . . . .	22
1.5.5	API TILF . . . . .	22
1.5.5.1	TILF_SearchTag . . . . .	22
1.5.5.2	TILF_ChargeOnlyRead . . . . .	22
1.5.5.3	TILF_ChargeOnlyReadLo . . . . .	22
1.5.5.4	TILF_SPPProgramPage . . . . .	23
1.5.5.5	TILF_SPPProgramPageLo . . . . .	23
1.5.5.6	TILF_MPGeneralReadPage . . . . .	23
1.5.5.7	TILF_MPSelectiveReadPage . . . . .	23
1.5.5.8	TILF_MPPProgramPage . . . . .	24
1.5.5.9	TILF_MPSelectiveProgramPage . . . . .	24
1.5.5.10	TILF_MPLockPage . . . . .	24
1.5.5.11	TILF_MPSelectiveLockPage . . . . .	24
1.5.5.12	TILF_MPGeneralReadPageLo . . . . .	25
1.5.5.13	TILF_MPSelectiveReadPageLo . . . . .	25
1.5.5.14	TILF_MPPProgramPageLo . . . . .	25
1.5.5.15	TILF_MPSelectiveProgramPageLo . . . . .	25
1.5.5.16	TILF_MPLockPageLo . . . . .	26
1.5.5.17	TILF_MPSelectiveLockPageLo . . . . .	26
1.5.5.18	TILF_MUGeneralReadPage . . . . .	26
1.5.5.19	TILF_MUSelectiveReadPage . . . . .	26
1.5.5.20	TILF_MUSpecialReadPage . . . . .	27
1.5.5.21	TILF_MUProgramPage . . . . .	27
1.5.5.22	TILF_MUSelectiveProgramPage . . . . .	27
1.5.5.23	TILF_MUSpecialProgramPage . . . . .	28
1.5.5.24	TILF_MULockPage . . . . .	28
1.5.5.25	TILF_MUSelectiveLockPage . . . . .	28
1.5.5.26	TILF_MUSpecialLockPage . . . . .	29
1.5.6	API HITAG1S . . . . .	29
1.5.6.1	Hitag1S_ReadPage . . . . .	29
1.5.6.2	Hitag1S_ReadBlock . . . . .	29
1.5.6.3	Hitag1S_WritePage . . . . .	30
1.5.6.4	Hitag1S_WriteBlock . . . . .	30
1.5.6.5	Hitag1S_Halt . . . . .	30
1.5.7	API HITAG2 . . . . .	31
1.5.7.1	Hitag2_ReadPage . . . . .	31
1.5.7.2	Hitag2_WritePage . . . . .	31
1.5.7.3	Hitag2_Halt . . . . .	31
1.5.7.4	Hitag2_SetPassword . . . . .	31
1.5.8	API SM4X00 . . . . .	32
1.5.8.1	SM4X00_GenericRaw . . . . .	32

1.5.8.2	SM4X00_Generic . . . . .	32
1.5.9	API I2C . . . . .	32
1.5.9.1	I2CInit . . . . .	32
1.5.9.2	I2CDeInit . . . . .	33
1.5.9.3	I2CMasterStart . . . . .	33
1.5.9.4	I2CMasterStop . . . . .	33
1.5.9.5	I2CMasterTransmitByte . . . . .	33
1.5.9.6	I2CMasterReceiveByte . . . . .	34
1.5.9.7	I2CMasterBeginWrite . . . . .	34
1.5.9.8	I2CMasterBeginRead . . . . .	34
1.5.9.9	I2CMasterSetAck . . . . .	34
1.5.10	API MIFARECLASSIC . . . . .	35
1.5.10.1	MifareClassic_Login . . . . .	35
1.5.10.2	MifareClassic_ReadBlock . . . . .	35
1.5.10.3	MifareClassic_WriteBlock . . . . .	35
1.5.10.4	MifareClassic_ReadValueBlock . . . . .	36
1.5.10.5	MifareClassic_WriteValueBlock . . . . .	36
1.5.10.6	MifareClassic_IncrementValueBlock . . . . .	36
1.5.10.7	MifareClassic_DecrementValueBlock . . . . .	36
1.5.10.8	MifareClassic_CopyValueBlock . . . . .	37
1.5.11	API MIFAREULTRALIGHT . . . . .	37
1.5.11.1	MifareUltralight_ReadPage . . . . .	37
1.5.11.2	MifareUltralight_WritePage . . . . .	37
1.5.11.3	MifareUltralightC_Authenticate . . . . .	38
1.5.11.4	MifareUltralightC_SAMAuthenticate . . . . .	38
1.5.11.5	MifareUltralightC_WriteKeyFromSAM . . . . .	38
1.5.11.6	MifareUltralightEV1_FastRead . . . . .	38
1.5.11.7	MifareUltralightEV1_IncCounter . . . . .	39
1.5.11.8	MifareUltralightEV1_ReadCounter . . . . .	39
1.5.11.9	MifareUltralightEV1_ReadSig . . . . .	39
1.5.11.10	MifareUltralightEV1_GetVersion . . . . .	39
1.5.11.11	MifareUltralightEV1_PwdAuth . . . . .	40
1.5.11.12	MifareUltralightEV1_CheckTearingEvent . . . . .	40
1.5.12	API ISO15693 . . . . .	40
1.5.12.1	ISO15693_GenericCommand . . . . .	40
1.5.12.2	ISO15693_GetSystemInformation . . . . .	41
1.5.12.3	ISO15693_GetSystemInformationExt . . . . .	41
1.5.12.4	ISO15693_GetTagTypeFromUID . . . . .	41
1.5.12.5	ISO15693_GetTagTypeFromSystemInfo . . . . .	41
1.5.12.6	ISO15693_ReadSingleBlock . . . . .	42
1.5.12.7	ISO15693_ReadSingleBlockExt . . . . .	42
1.5.12.8	ISO15693_WriteSingleBlock . . . . .	42
1.5.12.9	ISO15693_WriteSingleBlockExt . . . . .	42
1.5.13	API CRYPTO . . . . .	43
1.5.13.1	Crypto_Init . . . . .	43
1.5.13.2	Encrypt . . . . .	43
1.5.13.3	Decrypt . . . . .	43
1.5.13.4	CBC_ResetInitVector . . . . .	43
1.5.14	API DESFIRE . . . . .	44
1.5.14.1	DESFire_GetApplicationIDs . . . . .	44
1.5.14.2	DESFire_CreateApplication . . . . .	44

1.5.14.3	DESFire_DeleteApplication . . . . .	44
1.5.14.4	DESFire_SelectApplication . . . . .	45
1.5.14.5	DESFire_Authenticate . . . . .	45
1.5.14.6	DESFire_GetKeySettings . . . . .	45
1.5.14.7	DESFire_GetFileIDs . . . . .	46
1.5.14.8	DESFire_GetFileSettings . . . . .	46
1.5.14.9	DESFire_ReadData . . . . .	46
1.5.14.10	DESFire_WriteData . . . . .	47
1.5.14.11	DESFire_GetValue . . . . .	47
1.5.14.12	DESFire_Credit . . . . .	47
1.5.14.13	DESFire_Debit . . . . .	48
1.5.14.14	DESFire_LimitedCredit . . . . .	48
1.5.14.15	DESFire_FreeMem . . . . .	48
1.5.14.16	DESFire_FormatTag . . . . .	48
1.5.14.17	DESFire_CreateDataFile . . . . .	49
1.5.14.18	DESFire_CreateValueFile . . . . .	49
1.5.14.19	DESFire_GetVersion . . . . .	49
1.5.14.20	DESFire_DeleteFile . . . . .	50
1.5.14.21	DESFire_CommitTransaction . . . . .	50
1.5.14.22	DESFire_AbortTransaction . . . . .	50
1.5.14.23	DESFire_GetUID . . . . .	50
1.5.14.24	DESFire_GetKeyVersion . . . . .	51
1.5.14.25	DESFire_ChangeKeySettings . . . . .	51
1.5.14.26	DESFire_ChangeKey . . . . .	51
1.5.14.27	DESFire_ChangeFileSettings . . . . .	52
1.5.14.28	DESFire_DisableFormatCard . . . . .	52
1.5.14.29	DESFire_EnableRandomID . . . . .	52
1.5.14.30	DESFire_SetDefaultKey . . . . .	53
1.5.14.31	DESFire_SetATS . . . . .	53
1.5.14.32	DESFire_CreateRecordFile . . . . .	53
1.5.14.33	DESFire_ReadRecords . . . . .	54
1.5.14.34	DESFire_WriteRecord . . . . .	54
1.5.14.35	DESFire_ClearRecordFile . . . . .	54
1.5.15	API ISO7816 . . . . .	55
1.5.15.1	ISO7816_GetSlotStatus . . . . .	55
1.5.15.2	ISO7816_IccPowerOn . . . . .	55
1.5.15.3	ISO7816_IccPowerOff . . . . .	55
1.5.15.4	ISO7816_SetCommSettings . . . . .	56
1.5.15.5	ISO7816_Transceive . . . . .	56
1.5.15.6	ISO7816_ExchangeAPDU . . . . .	56
1.5.15.7	ISO7816_T0_TPDU . . . . .	57
1.5.16	API ICLASS . . . . .	57
1.5.16.1	ICLASS_GetPACBits . . . . .	57
1.5.17	API ISO14443 . . . . .	57
1.5.17.1	ISO14443A_GetATS . . . . .	57
1.5.17.2	ISO14443B_GetATQB . . . . .	58
1.5.17.3	ISO14443_4_CheckPresence . . . . .	58
1.5.17.4	ISO14443_4_TDX . . . . .	58
1.5.17.5	ISO14443A_GetATQA . . . . .	58
1.5.17.6	ISO14443A_GetSAK . . . . .	59
1.5.17.7	ISO14443B_GetAnswerToATTRIB . . . . .	59

1.5.17.8	ISO14443_3_TDX . . . . .	59
1.5.17.9	ISO14443A_SearchMultiTag . . . . .	59
1.5.17.10	ISO14443A_SelectTag . . . . .	60
1.5.18	API AT55 . . . . .	60
1.5.18.1	AT55_Begin . . . . .	60
1.5.18.2	AT55_ReadBlock . . . . .	60
1.5.18.3	AT55_ReadBlockProtected . . . . .	60
1.5.18.4	AT55_WriteBlock . . . . .	61
1.5.18.5	AT55_WriteBlockProtected . . . . .	61
1.5.18.6	AT55_WriteBlockAndLock . . . . .	61
1.5.18.7	AT55_WriteBlockProtectedAndLock . . . . .	61
1.5.19	API NFCSNEP . . . . .	62
1.5.19.1	SNEP_Init . . . . .	62
1.5.19.2	SNEP_GetConnectionState . . . . .	62
1.5.19.3	SNEP_GetFragmentByteCount . . . . .	62
1.5.19.4	SNEP_BeginMessage . . . . .	62
1.5.19.5	SNEP_SendMessageFragment . . . . .	63
1.5.19.6	SNEP_TestMessage . . . . .	63
1.5.19.7	SNEP_ReceiveMessageFragment . . . . .	63
1.5.19.8	SNEP_RequestMessage . . . . .	63
1.5.20	API EM4150 . . . . .	64
1.5.20.1	EM4150_Login . . . . .	64
1.5.20.2	EM4150_ReadWord . . . . .	64
1.5.20.3	EM4150_WriteWord . . . . .	64
1.5.20.4	EM4150_WritePassword . . . . .	65
1.5.20.5	EM4150_GetTagInfo . . . . .	65
1.5.21	API FILESYS . . . . .	65
1.5.21.1	FSMount . . . . .	65
1.5.21.2	FSFormat . . . . .	66
1.5.21.3	FSOpen . . . . .	66
1.5.21.4	FSClose . . . . .	66
1.5.21.5	FSCloseAll . . . . .	66
1.5.21.6	FSSeek . . . . .	67
1.5.21.7	FSTell . . . . .	67
1.5.21.8	FSReadBytes . . . . .	67
1.5.21.9	FSWriteBytes . . . . .	67
1.5.21.10	FSFindFirst . . . . .	68
1.5.21.11	FSFindNext . . . . .	68
1.5.21.12	FSDelete . . . . .	68
1.5.21.13	FSRename . . . . .	68
1.5.21.14	FSGetStorageInfo . . . . .	69
1.5.22	API MIFAREPLUS . . . . .	69
1.5.22.1	MFP_WritePerso . . . . .	69
1.5.22.2	MFP_CommitPerso . . . . .	69
1.5.22.3	MFP_Authenticate . . . . .	70
1.5.22.4	MFP_ReadBlock . . . . .	70
1.5.22.5	MFP_WriteBlock . . . . .	70
1.5.22.6	MFP_ReadValueBlock . . . . .	70
1.5.22.7	MFP_WriteValueBlock . . . . .	71
1.5.22.8	MFP_IncrementValueBlock . . . . .	71
1.5.22.9	MFP_DecrementValueBlock . . . . .	71

1.5.22.10	MFP_CopyValueBlock . . . . .	71
1.5.23	API ADC . . . . .	72
1.5.23.1	ADCInitChannel . . . . .	72
1.5.23.2	ADCGetConversionValue . . . . .	72
1.5.24	API FELICA . . . . .	72
1.5.24.1	FeliCa_TDX . . . . .	72
1.5.24.2	FeliCa_ReadWithoutEncryption . . . . .	73
1.5.24.3	FeliCa_WriteWithoutEncryption . . . . .	73
1.5.24.4	FeliCa_RequestSystemCode . . . . .	73
1.5.24.5	FeliCa_Poll . . . . .	74
1.5.24.6	FeliCa_RequestService . . . . .	74
1.5.25	API SLE44XX . . . . .	74
1.5.25.1	SLE_GetATR . . . . .	74
1.5.25.2	SLE_ReadMainMemory . . . . .	75
1.5.25.3	SLE_UpdateMainMemory . . . . .	75
1.5.25.4	SLE_ReadSecurityMemory . . . . .	75
1.5.25.5	SLE_UpdateSecurityMemory . . . . .	75
1.5.25.6	SLE_ReadProtectionMemory . . . . .	76
1.5.25.7	SLE_WriteProtectionMemory . . . . .	76
1.5.25.8	SLE_CompareVerificationData . . . . .	76
1.5.26	API NTAG . . . . .	77
1.5.26.1	NTAG_Read . . . . .	77
1.5.26.2	NTAG_Write . . . . .	77
1.5.26.3	NTAG_FastRead . . . . .	77
1.5.26.4	NTAG_ReadCounter . . . . .	78
1.5.26.5	NTAG_ReadSig . . . . .	78
1.5.26.6	NTAG_GetVersion . . . . .	78
1.5.26.7	NTAG_PwdAuth . . . . .	78
1.5.26.8	NTAG_SectorSelect . . . . .	79
1.5.27	API SRX . . . . .	79
1.5.27.1	SRX_ReadBlock . . . . .	79
1.5.27.2	SRX_WriteBlock . . . . .	79
1.5.28	API SAMAVX . . . . .	80
1.5.28.1	SAMAVx_AuthenticateHost . . . . .	80
1.5.28.2	SAMAVx_GetKeyEntry . . . . .	80
1.5.29	API EM4102 . . . . .	80
1.5.29.1	EM4102_GetTagInfo . . . . .	80
1.5.30	API SPI . . . . .	81
1.5.30.1	SPIInit . . . . .	81
1.5.30.2	SPIDeInit . . . . .	81
1.5.30.3	SPIMasterBeginTransfer . . . . .	81
1.5.30.4	SPIMasterEndTransfer . . . . .	81
1.5.30.5	SPITransmit . . . . .	82
1.5.30.6	SPIReceive . . . . .	82
1.5.30.7	SPITransceive . . . . .	82
1.5.31	API BLE . . . . .	83
1.5.31.1	BLEPresetConfig . . . . .	83
1.5.31.2	BLEPresetUserData . . . . .	83
1.5.31.3	BLEInit . . . . .	83
1.5.31.4	BLECheckEvent . . . . .	84
1.5.31.5	BLEGetAddress . . . . .	84

1.5.31.6	BLEGetVersion . . . . .	84
1.5.31.7	BLEGetEnvironment . . . . .	84
1.5.31.8	BLEGetGattServerAttributeValue . . . . .	85
1.5.31.9	BLESetGattServerAttributeValue . . . . .	85
1.5.31.10	BLERequestRssi . . . . .	85
1.5.31.11	BLERequestEndpointClose . . . . .	85
2	Disclaimer . . . . .	86



# 1 Simple Protocol

This document describes the serial protocol of TWN4.

In order to operate this protocol, a firmware type TWN4\_Cxvvv\_PRSwww.bix is required, where vvv and www are the version numbers.

A firmware as mentioned above combines virtual USB (CDC) or true serial communication with an TWN4 app, which implements the simple protocol (PRS = PRotocol Simple).

This protocol is called simple because it is based on a communication with ASCII characters which can also be tested manually by using a terminal program. There is no additional overhead for things like packet repetition, address bytes...

The simple protocol is also available in binary mode. This means, that the data is not transmitted via ASCII characters but as single bytes.

Moreover it is possible to add a CRC at the end of every transmission. This lets you detect transmission errors.

The communication is based on a command/response structure: TWN4 will only send data to the host as a response of a command. Command and response are lines of bytes terminated by a carriage return. Carriage return is not shown explicitly anymore in the following documentation. A byte is always represented and transmitted by two hexadecimal ASCII characters.

## 1.1 Command

A command always starts with two bytes which reflect the API and function number to be executed.

## 1.2 Response

A response always starts with a byte, which reflects execution of the command on protocol level. Following possible error values:

ERR_NONE	0
ERR_UNKNOWN_FUNCTION	1
ERR_MISSING_PARAMETER	2
ERR_UNUSED_PARAMETERS	3
ERR_INVALID_FUNCTION	4
ERR_PARSER	5

## 1.3 Data Transmission

Data can be transmitted in two ways:

- by sending ASCII characters
- by sending binary values

### 1.3.1 ASCII

To transmit a value of e.g. 0x1F, it is necessary to split this into two ASCII characters '1' and 'F'. These characters has to be sent sequentially.

### 1.3.2 Binary

To transmit a value of e.g. 0x1F, it can be sent directly in binary format.

### 1.3.3 CRC

On both ASCII and binary format, a CRC can be added at the end of each transmission. The CRC is calculated as follows:

```
uint16_t UpdateCRC(uint16_t CRC, byte Byte)
{
    // Update CCITT CRC (reverse polynom 0x8408)
    Byte ^= (byte)CRC;
    Byte ^= (byte)(Byte << 4);
    return (uint16_t)((((Byte << 8) | (CRC >> 8)) ^ (Byte >> 4) ^ (Byte << 3)));
}
```

The CRC calculation starts with CRC = 0xFFFF

### 1.3.4 Reference messages

The following table shows reference messages for function GetUSBType

Mode	CRC	Command (Host -> TWN4)	Response (TWN4 -> Host)
ASCII	Off	"0005\r"	"0001\r"
	On	"000515A7\r"	"000131E1\r"
Binary	Off	0x02 0x00 0x00 0x05	0x02 0x00 0x00 0x01
	On	0x04 0x00 0x00 0x05 0x15 0xA7	0x04 0x00 0x00 0x01 0x31 0xE1

## 1.4 Data Types

The description of the commands is using data types, which have to be built-up as follows:

Data Type	Description
[Byte]:	One single byte (sent as two hex digits)
[UInt16]:	Two bytes (LSB first)
[UInt32]:	Four bytes (LSB first)
[Bool]:	One single byte which can hold two values: 0 or 1
[Byte Array(n)]:	A sequence of bytes with known and fixed number of bytes. The number of bytes is not transferred explicitly, because both host and TWN4 do know this number.
[Byte Array(Var)]:	A sequence of bytes, where the first byte holds the number of following bytes
[Byte Array(Var), x LB]:	A sequence of bytes, where the first x bytes hold the number of following bytes
[ASCII string]:	A sequence of bytes which contain ASCII characters, except the first byte which holds the number of following bytes

In Simple Protocol, all numbers are sent with LSB first. For example, the number 0x1234 has to be sent as 3412.

## 1.5 Commands

### 1.5.1 API SYS

#### 1.5.1.1 Reset

Command:	[0001]
Response:	[00]
Example Command: Response:	0001

#### 1.5.1.2 StartBootloader

Command:	[0002]
Response:	[00]
Example Command: Response:	0002

**1.5.1.3 GetSysTicks**

Command:	[0003]
Response:	[00][UInt32: <i>Ticks</i> ]
Example	
Command:	0003
Response:	00D3480700 (Ticks: 477395)

**1.5.1.4 GetVersionString**

Command:	[0004][Byte: <i>MaxLen</i> ]
Response:	[00][ASCII string: <i>Version</i> ]
Example	
Command:	0004FF (MaxLen: FF)
Response:	001D54574E342F42312E30332F434346312E35372F505253312E3033-2F5049 (Version: TWN4/B1.03/CCF1.57/PRS1.03/PI)

**1.5.1.5 GetUSBType**

Command:	[0005]
Response:	[00][Byte: <i>Type</i> ]
Example	
Command:	0005
Response:	0001 (Type: 1)

**1.5.1.6 GetDeviceType**

Command:	[0006]
Response:	[00][Byte: <i>Type</i> ]
Example	
Command:	0006
Response:	000B (Type: 11)

**1.5.1.7 Sleep**

Command:	[0007][UInt32: <i>Ticks</i> ][UInt32: <i>Flags</i> ]
Response:	[00][Byte: <i>Result</i> ]
Example	
Command:	0007E803000001000000 (Ticks: E8030000, Flags: 01000000)
Response:	0000 (Result: 0)

**1.5.1.8 GetDeviceUID**

Command:	[0008]
Response:	[00][Byte Array(12): <i>UID</i> ]
Example	
Command:	0008
Response:	002D002F000B47303531353233 (UID: 2D002F000B47303531353233)

**1.5.1.9 SetParameters**

Command:	[0009][Byte Array(Var): <i>TLV</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	00090707010103010200 (TLV: 07010103010200)
Response:	0001 (Result: true)

**1.5.1.10 GetLastError**

Command:	[000A]
Response:	[00][UInt32: <i>LastError</i> ]
Example	
Command:	000A
Response:	00CB000000 (LastError: 203)

## 1.5.2 API IO

### 1.5.2.1 WriteByte

Command:	[0100][Byte: <i>Channel</i> ][Byte: <i>Byte</i> ]
Response:	[00]
Example	
Command:	01000041 (Channel: 00, Byte: 41)
Response:	00

### 1.5.2.2 ReadByte

Command:	[0101][Byte: <i>Channel</i> ]
Response:	[00][Byte: <i>Byte</i> ]
Example	
Command:	010100 (Channel: 00)
Response:	0000 (Byte: 0)

### 1.5.2.3 TestEmpty

Command:	[0102][Byte: <i>Channel</i> ][Byte: <i>Dir</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	01020001 (Channel: 00, Dir: 01)
Response:	0001 (Result: Yes)

**1.5.2.4 TestFull**

Command:	[0103][Byte: <i>Channel</i> ][Byte: <i>Dir</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	01030001 (Channel: 00, Dir: 01)
Response:	0000 (Result: No)

**1.5.2.5 GetBufferSize**

Command:	[0104][Byte: <i>Channel</i> ][Byte: <i>Dir</i> ]
Response:	[00][UInt16: <i>BufferSize</i> ]
Example	
Command:	01040001 (Channel: 00, Dir: 01)
Response:	000000 (BufferSize: 0)

**1.5.2.6 GetByteCount**

Command:	[0105][Byte: <i>Channel</i> ][Byte: <i>Dir</i> ]
Response:	[00][UInt16: <i>ByteCount</i> ]
Example	
Command:	01050001 (Channel: 00, Dir: 01)
Response:	000000 (ByteCount: 0)

**1.5.2.7 SetCOMParameters**

Command:	[0109][Byte: <i>Channel</i> ][UInt32: <i>Baudrate</i> ][Byte: <i>WordLength</i> ][Byte: <i>Parity</i> ][Byte: <i>StopBits</i> ][Byte: <i>FlowControl</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0109028025000008000100 (Channel: 02, Baudrate: 80250000, WordLength: 08, Parity: 00, StopBits: 01, FlowControl: 00)
Response:	0001 (Result: true)

**1.5.2.8 GetUSBDeviceState**

Command:	[010A]
Response:	[00][Byte: <i>State</i> ]
Example	
Command:	010A
Response:	0003 (State: USB_DEVICE_STATE_CONFIGURED)

**1.5.2.9 GetHostChannel**

Command:	[010B]
Response:	[00][Byte: <i>Channel</i> ]
Example	
Command:	010B
Response:	0001 (Channel: CHANNEL_USB)

**1.5.2.10 USBRemoteWakeup**

Command:	[010C]
Response:	[00]
Example	
Command:	010C
Response:	00



**1.5.2.11 WriteBytes**

Command:	[010D][Byte: <i>Channel</i> ][Byte Array(Var), 2 LB: <i>Bytes</i> ]
Response:	[00][UInt16: <i>BytesWritten</i> ]
Example	
Command:	010D020300000815 (Channel: 02, Bytes: 000815)
Response:	000300 (BytesWritten: 3)

**1.5.2.12 ReadBytes**

Command:	[010E][Byte: <i>Channel</i> ][UInt16: <i>MaxBytes</i> ]
Response:	[00][Byte Array(Var), 2 LB: <i>Bytes</i> ]
Example	
Command:	010E020F00 (Channel: 02, MaxBytes: 0F00)
Response:	000300000815 (Bytes: 000815)

**1.5.3 API PERIPH****1.5.3.1 GPIOConfigureOutputs**

Command:	[0400][Byte: <i>Bits</i> ][Byte: <i>PullUpDown</i> ][Byte: <i>OutputType</i> ]
Response:	[00]
Example	
Command:	0400010000 (Bits: 01, PullUpDown: 00, OutputType: 00)
Response:	00

**1.5.3.2 GPIOConfigureInputs**

Command:	[0401][Byte: <i>Bits</i> ][Byte: <i>PullUpDown</i> ]
Response:	[00]
Example	
Command:	04010100 (Bits: 01, PullUpDown: 00)
Response:	00

**1.5.3.3 GPIOSetBits**

Command:	[0402][Byte: <i>Bits</i> ]
Response:	[00]
Example	
Command:	040201 (Bits: 01)
Response:	00

**1.5.3.4 GPIOClearBits**

Command:	[0403][Byte: <i>Bits</i> ]
Response:	[00]
Example	
Command:	040301 (Bits: 01)
Response:	00

**1.5.3.5 GPIToggleBits**

Command:	[0404][Byte: <i>Bits</i> ]
Response:	[00]
Example	
Command:	040401 (Bits: 01)
Response:	00

**1.5.3.6 GPIOBlinkBits**

Command:	[0405][Byte: <i>Bits</i> ][UInt16: <i>TimeHi</i> ][UInt16: <i>TimeLo</i> ]
Response:	[00]
Example	
Command:	04050164006400 (Bits: 01, TimeHi: 6400, TimeLo: 6400)
Response:	00

**1.5.3.7 GPIOTestBit**

Command:	[0406][Byte: <i>Bit</i> ]
Response:	[00][Byte: <i>Result</i> ]
Example	
Command:	040601 (Bit: 01)
Response:	0000 (Result: 0)

**1.5.3.8 Beep**

Command:	[0407][Byte: <i>Volume</i> ][UInt16: <i>Frequency</i> ][UInt16: <i>OnTime</i> ][UInt16: <i>OffTime</i> ]
Response:	[00]
Example	
Command:	0407646009F401F401 (Volume: 64, Frequency: 6009, OnTime: F401, OffTime: F401)
Response:	00

**1.5.3.9 DiagLEDOn**

Command:	[0408]
Response:	[00]
Example	
Command:	0408
Response:	00

**1.5.3.10 DiagLEDOff**

Command:	[0409]
Response:	[00]
Example	
Command:	0409
Response:	00

**1.5.3.11 DiagLEDToggle**

Command:	[040A]
Response:	[00]
Example	
Command:	040A
Response:	00

**1.5.3.12 DiagLEDIsOn**

Command:	[040B]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	040B
Response:	0000 (Result: No)

**1.5.3.13 SendWiegand**

Command:	[040C][Byte: <i>GPIOData0</i> ][Byte: <i>GPIOData1</i> ][UInt16: <i>PulseTime</i> ][UInt16: <i>IntervalTime</i> ][Byte Array(Var): <i>Bits</i> ][Byte: <i>BitCount</i> ]
Response:	[00]
Example	
Command:	040C08106400E80301AA08 (GPIOData0: 08, GPIOData1: 10, PulseTime: 6400, IntervalTime: E803, Bits: AA, BitCount: 08)
Response:	00

**1.5.3.14 SendOmron**

Command:	[040D][Byte: <i>GPIOClock</i> ][Byte: <i>GPIOData</i> ][UInt16: <i>T1</i> ][UInt16: <i>T2</i> ][UInt16: <i>T3</i> ][Byte Array(Var): <i>Bits</i> ][Byte: <i>BitCount</i> ]
Response:	[00]
Example	
Command:	040D0810F401F401F40101AA08 (GPIOClock: 08, GPIOData: 10, T1: F401, T2: F401, T3: F401, Bits: AA, BitCount: 08)
Response:	00

## 1.5.4 API RF

### 1.5.4.1 SearchTag

Command:	[0500][Byte: <i>MaxIDBytes</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte: <i>TagType</i> ][Byte: <i>IDBitCount</i> ][Byte Array(Var): <i>ID</i> ]
Example	
Command:	050010 (MaxIDBytes: 10)
Response:	000180200466CF4DC2 (Result: true, TagType: ISO14443A/MIFARE, IDBitCount: 32, ID: 66CF4DC2)

### 1.5.4.2 SetRFOff

Command:	[0501]
Response:	[00]
Example	
Command:	0501
Response:	00

### 1.5.4.3 SetTagTypes

Command:	[0502][UInt32: <i>TagTypesLF</i> ][UInt32: <i>TagTypesHF</i> ]
Response:	[00]
Example	
Command:	0502FFFFFFFFFFFFFFFF (TagTypesLF: FFFFFFFF, TagTypesHF: FFFFFFFF)
Response:	00

### 1.5.4.4 GetTagTypes

Command:	[0503]
Response:	[00][UInt32: <i>LFTagTypes</i> ][UInt32: <i>HFTagTypes</i> ]
Example	
Command:	0503
Response:	002FFE0700F7000000 (LFTagTypes: 523823, HFTagTypes: 247)

**1.5.4.5 GetSupportedTagTypes**

Command:	[0504]
Response:	[00][UInt32: <i>LFTagTypes</i> ][UInt32: <i>HFTagTypes</i> ]
Example	
Command:	0504
Response:	002FFE0700F7000000 (LFTagTypes: 523823, HFTagTypes: 247)

**1.5.5 API TILF****1.5.5.1 TILF\_SearchTag**

Command:	[0600][Byte: <i>MaxIDBytes</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte: <i>IDBitCount</i> ][Byte Array(Var): <i>ID</i> ]
Example	
Command:	060010 (MaxIDBytes: 10)
Response:	00014008000000000042E8653 (Result: true, IDBitCount: 64, ID: 00000000042E8653)

**1.5.5.2 TILF\_ChargeOnlyRead**

Command:	[0601]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>Data</i> ]
Example	
Command:	0601
Response:	000100000000042E8653 (Result: true, Data: 00000000042E8653)

**1.5.5.3 TILF\_ChargeOnlyReadLo**

Command:	[0602]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	0602
Response:	000100007F7E7EFFFDFDFDFDFDFDFDFDFDFD (Result: true, ReadData: 00007F7E7EFFFDFDFDFDFDFDFDFDFDFD)

**1.5.5.4 TILF\_SPProgramPage**

Command:	[0603][Byte Array(8): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	06030001020304050607 (WriteData: 0001020304050607)
Response:	000100007ECA61742000000000DADF7E0000 (Result: true, ReadData: 00007ECA61742000000000DADF7E0000)

**1.5.5.5 TILF\_SPProgramPageLo**

Command:	[0604][Byte Array(10): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	060400010203040506070809 (WriteData: 00010203040506070809)
Response:	000100007ECA61742000000000DADF7E0000 (Result: true, ReadData: 00007ECA61742000000000DADF7E0000)

**1.5.5.6 TILF\_MPGeneralReadPage**

Command:	[0605][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>ReadData</i> ]
Example	
Command:	060500 (Address: 00)
Response:	0001000000000042E8653 (Result: true, ReadData: 000000000042E8653)

**1.5.5.7 TILF\_MPSelectiveReadPage**

Command:	[0606][Byte: <i>Address</i> ][Byte Array(3): <i>SelectiveAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>ReadData</i> ]
Example	
Command:	060600000102 (Address: 00, SelectiveAddress: 000102)
Response:	0001000000000042E8653 (Result: true, ReadData: 000000000042E8653)

**1.5.5.8 TILF\_MPProgramPage**

Command:	[0607][Byte: <i>Address</i> ][Byte Array(8): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>ReadData</i> ]
Example	
Command:	0607004469726563746F72 (Address: 00, WriteData: 4469726563746F72)
Response:	000100000000042E8653 (Result: true, ReadData: 00000000042E8653)

**1.5.5.9 TILF\_MPSelectiveProgramPage**

Command:	[0608][Byte: <i>Address</i> ][Byte Array(3): <i>SelectiveAddress</i> ][Byte Array(8): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>ReadData</i> ]
Example	
Command:	0608000001024469726563746F72 (Address: 00, SelectiveAddress: 000102, WriteData: 4469726563746F72)
Response:	000100000000042E8653 (Result: true, ReadData: 00000000042E8653)

**1.5.5.10 TILF\_MPLockPage**

Command:	[0609][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>ReadData</i> ]
Example	
Command:	060900 (Address: 00)
Response:	0000 (Result: fail, ReadData: )

**1.5.5.11 TILF\_MPSelectiveLockPage**

Command:	[060A][Byte: <i>Address</i> ][Byte Array(3): <i>SelectiveAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>ReadData</i> ]
Example	
Command:	060A00000102 (Address: 00, SelectiveAddress: 000102)
Response:	0000 (Result: fail, ReadData: )



**1.5.5.12 TILF\_MPGeneralReadPageLo**

Command:	[060B][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	060B00 (Address: 00)
Response:	000100007ECA61742000000000DADF7E0000 (Result: true, ReadData: 00007ECA61742000000000DADF7E0000)

**1.5.5.13 TILF\_MPSelectiveReadPageLo**

Command:	[060C][Byte: <i>Address</i> ][Byte Array(3): <i>SelectiveAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	060C00000102 (Address: 00, SelectiveAddress: 000102)
Response:	000100007ECA61742000000000DADF7E0000 (Result: true, ReadData: 00007ECA61742000000000DADF7E0000)

**1.5.5.14 TILF\_MPProgramPageLo**

Command:	[060D][Byte: <i>Address</i> ][Byte Array(10): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	060D00536F6D6520746578742E (Address: 00, WriteData: 536F6D6520746578742E)
Response:	000100007ECA61742000000000DADF7E0000 (Result: true, ReadData: 00007ECA61742000000000DADF7E0000)

**1.5.5.15 TILF\_MPSelectiveProgramPageLo**

Command:	[060E][Byte: <i>Address</i> ][Byte Array(3): <i>SelectiveAddress</i> ][Byte Array(10): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	060E00000102536F6D6520746578742E (Address: 00, SelectiveAddress: 000102, WriteData: 536F6D6520746578742E)
Response:	000100007ECA61742000000000DADF7E0000 (Result: true, ReadData: 00007ECA61742000000000DADF7E0000)

**1.5.5.16 TILF\_MPLockPageLo**

Command:	[060F][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	060F00 (Address: 00)
Response:	0000 (Result: fail, ReadData: )

**1.5.5.17 TILF\_MPSelectiveLockPageLo**

Command:	[0610][Byte: <i>Address</i> ][Byte Array(3): <i>SelectiveAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>ReadData</i> ]
Example	
Command:	061000000102 (Address: 00, SelectiveAddress: 000102)
Response:	000100007FEFFFFFFFBFF7FFFAFFFFFFFFF7 (Result: true, ReadData: 00007FEFFFFFFFBFF7FFFAFFFFFFFFF7)

**1.5.5.18 TILF\_MUGeneralReadPage**

Command:	[0611][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>Data</i> ]
Example	
Command:	061100 (Address: 00)
Response:	0000 (Result: fail, Data: )

**1.5.5.19 TILF\_MUSelectiveReadPage**

Command:	[0612][Byte: <i>Address</i> ][Byte: <i>SelectiveAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>Data</i> ]
Example	
Command:	06120000 (Address: 00, SelectiveAddress: 00)
Response:	0000 (Result: fail, Data: )

**1.5.5.20 TILF\_MUSpecialReadPage**

Command:	[0613][Byte: <i>Address</i> ][Byte Array(5): <i>SpecialAddress1</i> ][Byte Array(3): <i>SpecialAddress2</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>Data</i> ]
Example	
Command:	0613000001020304000102 (Address: 00, SpecialAddress1: 0001020304, SpecialAddress2: 000102)
Response:	0000 (Result: fail, Data: )

**1.5.5.21 TILF\_MUProgramPage**

Command:	[0614][Byte: <i>Address</i> ][Byte Array(5): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>ReadData</i> ]
Example	
Command:	06140048656C6C6F (Address: 00, WriteData: 48656C6C6F)
Response:	0000 (Result: fail, ReadData: )

**1.5.5.22 TILF\_MUSelectiveProgramPage**

Command:	[0615][Byte: <i>Address</i> ][Byte: <i>SelectiveAddress</i> ][Byte Array(5): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>ReadData</i> ]
Example	
Command:	0615000048656C6C6F (Address: 00, SelectiveAddress: 00, WriteData: 48656C6C6F)
Response:	0000 (Result: fail, ReadData: )

**1.5.5.23 TILF\_MUSpecialProgramPage**

Command:	[0616][Byte: <i>Address</i> ][Byte Array(5): <i>SpecialAddress1</i> ][Byte Array(3): <i>SpecialAddress2</i> ][Byte Array(5): <i>WriteData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>ReadData</i> ]
Example	
Command:	061600000102030400010248656C6C6F (Address: 00, SpecialAddress1: 0001020304, SpecialAddress2: 000102, WriteData: 48656C6C6F)
Response:	0000 (Result: fail, ReadData: )

**1.5.5.24 TILF\_MULockPage**

Command:	[0617][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>ReadData</i> ]
Example	
Command:	061700 (Address: 00)
Response:	0000 (Result: fail, ReadData: )

**1.5.5.25 TILF\_MUSelectiveLockPage**

Command:	[0618][Byte: <i>Address</i> ][Byte: <i>SelectiveAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>ReadData</i> ]
Example	
Command:	06180000 (Address: 00, SelectiveAddress: 00)
Response:	0000 (Result: fail, ReadData: )

**1.5.5.26 TILF\_MUSpecialLockPage**

Command:	[0619][Byte: <i>Address</i> ][Byte Array(5): <i>SpecialAddress1</i> ][Byte Array(3): <i>SpecialAddress2</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(7): <i>ReadData</i> ]
Example	
Command:	0619000001020304000102 (Address: 00, SpecialAddress1: 0001020304, SpecialAddress2: 000102)
Response:	0000 (Result: fail, ReadData: )

**1.5.6 API HITAG1S****1.5.6.1 Hitag1S\_ReadPage**

Command:	[0701][Byte: <i>PageAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>Data</i> ]
Example	
Command:	070104 (PageAddress: 04)
Response:	0001FF8CA64A (Result: true, Data: FF8CA64A)

**1.5.6.2 Hitag1S\_ReadBlock**

Command:	[0702][Byte: <i>BlockAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>Data</i> ]
Example	
Command:	070204 (BlockAddress: 04)
Response:	0001100001020398F8C802FFFFFFFFFFFFFFFFFFFF (Result: true, Data: 0001020398F8C802FFFFFFFFFFFFFFFFFFFF)



## 1.5.7 API HITAG2

### 1.5.7.1 Hitag2\_ReadPage

Command:	[0801][Byte: <i>PageAddress</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>Data</i> ]
Example	
Command:	080104 (PageAddress: 04)
Response:	0001FF800000 (Result: true, Data: FF800000)

### 1.5.7.2 Hitag2\_WritePage

Command:	[0802][Byte: <i>PageAddress</i> ][Byte Array(4): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	080204FF800000 (PageAddress: 04, Data: FF800000)
Response:	0001 (Result: true)

### 1.5.7.3 Hitag2\_Halt

Command:	[0803]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0803
Response:	0001 (Result: true)

### 1.5.7.4 Hitag2\_SetPassword

Command:	[0804][Byte Array(4): <i>Password</i> ]
Response:	[00]
Example	
Command:	080400010203 (Password: 00010203)
Response:	00

## 1.5.8 API SM4X00

### 1.5.8.1 SM4X00\_GenericRaw

Command:	[0900][Byte Array(Var): <i>TXData</i> ][Byte: <i>MaxRXDataLength</i> ][UInt16: <i>Timeout</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>RXData</i> ]
Example	
Command:	090005040A000000040B80B (TXData: 040A000000, MaxRXDataLength: 40, Timeout: B80B)
Response:	00010D0A000009010501001801030100 (Result: true, RXData: 0A000009010501001801030100)

### 1.5.8.2 SM4X00\_Generic

Command:	[0901][Byte Array(Var): <i>TXData</i> ][Byte: <i>MaxRXDataLength</i> ][UInt16: <i>Timeout</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>RXData</i> ]
Example	
Command:	0901020A0040B80B (TXData: 0A00, MaxRXDataLength: 40, Timeout: B80B)
Response:	0001100F0A000009010501001801030100EB63 (Result: true, RXData: 0F0A000009010501001801030100EB63)

## 1.5.9 API I2C

### 1.5.9.1 I2CInit

Command:	[0A00][UInt16: <i>Mode</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0A000000 (Mode: 0000)
Response:	0001 (Result: true)



**1.5.9.2 I2CDeInit**

Command:	[0A01]
Response:	[00]
Example	
Command:	0A01
Response:	00

**1.5.9.3 I2CMasterStart**

Command:	[0A02]
Response:	[00]
Example	
Command:	0A02
Response:	00

**1.5.9.4 I2CMasterStop**

Command:	[0A03]
Response:	[00]
Example	
Command:	0A03
Response:	00

**1.5.9.5 I2CMasterTransmitByte**

Command:	[0A04][Byte: <i>Data</i> ]
Response:	[00]
Example	
Command:	0A0400 (Data: 00)
Response:	00

**1.5.9.6 I2CMasterReceiveByte**

Command:	[0A05]
Response:	[00][Byte: <i>Data</i> ]
Example	
Command:	0A05
Response:	0000 (Data: 0)

**1.5.9.7 I2CMasterBeginWrite**

Command:	[0A06][Byte: <i>Address</i> ]
Response:	[00]
Example	
Command:	0A0630 (Address: 30)
Response:	00

**1.5.9.8 I2CMasterBeginRead**

Command:	[0A07][Byte: <i>Address</i> ]
Response:	[00]
Example	
Command:	0A0730 (Address: 30)
Response:	00

**1.5.9.9 I2CMasterSetAck**

Command:	[0A08][Byte: <i>SetOn</i> ]
Response:	[00]
Example	
Command:	0A0801 (SetOn: 01)
Response:	00

## 1.5.10 API MIFARECLASSIC

### 1.5.10.1 MifareClassic\_Login

Command:	[0B00][Byte Array(6): <i>Key</i> ][Byte: <i>KeyType</i> ][Byte: <i>Sector</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0B00A0A1A2A3A4A50000 (Key: A0A1A2A3A4A5, KeyType: 00, Sector: 00)
Response:	0001 (Result: true)

### 1.5.10.2 MifareClassic\_ReadBlock

Command:	[0B01][Byte: <i>Block</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>Data</i> ]
Example	
Command:	0B0102 (Block: 02)
Response:	00010000000000000000000000000000 (Result: true, Data: 00000000000000000000000000000000)

### 1.5.10.3 MifareClassic\_WriteBlock

Command:	[0B02][Byte: <i>Block</i> ][Byte Array(16): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0B02020000000000000000000000000000 (Block: 02, Data: 00000000000000000000000000000000)
Response:	0001 (Result: true)

**1.5.10.4 MifareClassic\_ReadValueBlock**

Command:	[0B03][Byte: <i>Block</i> ]
Response:	[00][Bool: <i>Result</i> ][UInt32: <i>Value</i> ]
Example	
Command:	0B0302 (Block: 02)
Response:	000101000000 (Result: true, Value: 1)

**1.5.10.5 MifareClassic\_WriteValueBlock**

Command:	[0B04][Byte: <i>Block</i> ][UInt32: <i>Value</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0B040201000000 (Block: 02, Value: 01000000)
Response:	0001 (Result: true)

**1.5.10.6 MifareClassic\_IncrementValueBlock**

Command:	[0B05][Byte: <i>Block</i> ][UInt32: <i>Value</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0B050201000000 (Block: 02, Value: 01000000)
Response:	0001 (Result: true)

**1.5.10.7 MifareClassic\_DecrementValueBlock**

Command:	[0B06][Byte: <i>Block</i> ][UInt32: <i>Value</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0B060201000000 (Block: 02, Value: 01000000)
Response:	0001 (Result: true)

**1.5.10.8 MifareClassic\_CopyValueBlock**

Command:	[0B07][Byte: <i>SourceBlock</i> ][Byte: <i>DestBlock</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0B07090A (SourceBlock: 09, DestBlock: 0A)
Response:	0001 (Result: true)

**1.5.11 API MIFAREULTRALIGHT****1.5.11.1 MifareUltralight\_ReadPage**

Command:	[0C00][Byte: <i>Page</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>Data</i> ]
Example	
Command:	0C0004 (Page: 04)
Response:	000100010203147870672E636F6D3A636172 (Result: true, Data: 00010203147870672E636F6D3A636172)

**1.5.11.2 MifareUltralight\_WritePage**

Command:	[0C01][Byte: <i>Page</i> ][Byte Array(4): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0C010400010203 (Page: 04, Data: 00010203)
Response:	0001 (Result: true)

**1.5.11.3 MifareUltralightC\_Authenticate**

Command:	[0C02][Byte Array(16): <i>Key</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0C0249454D4B41455242214E4143554F5946 (Key: 49454D4B41455242214E4143554F5946)
Response:	0001 (Result: true)

**1.5.11.4 MifareUltralightC\_SAMAuthenticate**

Command:	[0C03][Byte: <i>KeyNo</i> ][Byte: <i>KeyVersion</i> ][Byte Array(Var): <i>DIVInput</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0C03010000 (KeyNo: 01, KeyVersion: 00, DIVInput: )
Response:	0001 (Result: true)

**1.5.11.5 MifareUltralightC\_WriteKeyFromSAM**

Command:	[0C04][Byte: <i>KeyNo</i> ][Byte: <i>KeyVersion</i> ][Byte Array(Var): <i>DIVInput</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0C04010000 (KeyNo: 01, KeyVersion: 00, DIVInput: )
Response:	0000 (Result: fail)

**1.5.11.6 MifareUltralightEV1\_FastRead**

Command:	[0C05][Byte: <i>StartPage</i> ][Byte: <i>NumberOfPages</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>Data</i> ]
Example	
Command:	0C050401 (StartPage: 04, NumberOfPages: 01)
Response:	0001040000000000 (Result: true, Data: 00000000)

**1.5.11.7 MifareUltralightEV1\_IncCounter**

Command:	[0C06][Byte: <i>CounterAddr</i> ][UInt32: <i>IncrValue</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0C06000000000000 (CounterAddr: 00, IncrValue: 00000000)
Response:	0001 (Result: true)

**1.5.11.8 MifareUltralightEV1\_ReadCounter**

Command:	[0C07][Byte: <i>CounterAddr</i> ]
Response:	[00][Bool: <i>Result</i> ][UInt32: <i>CounterValue</i> ]
Example	
Command:	0C0700 (CounterAddr: 00)
Response:	000102000000 (Result: true, CounterValue: 2)

**1.5.11.9 MifareUltralightEV1\_ReadSig**

Command:	[0C08]
Response:	[00][Bool: <i>Result</i> ][Byte Array(32): <i>ECCSig</i> ]
Example	
Command:	0C08
Response:	00013A4F2622AF2039E47F8AA1BF84C52EE949860DD07125BEF75EC4- 17833B80C105 (Result: true, ECCSig: 3A4F2622AF2039E47F8AA1BF84C52EE949860DD07125BEF75EC417833B80C105)

**1.5.11.10 MifareUltralightEV1\_GetVersion**

Command:	[0C09]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>Version</i> ]
Example	
Command:	0C09
Response:	00010004030101000E03 (Result: true, Version: 0004030101000E03)

**1.5.11.11 MifareUltralightEV1\_PwdAuth**

Command:	[0C0A][Byte Array(4): <i>Password</i> ][Byte Array(2): <i>PwdAck</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0C0AFFFFFFFF0000 (Password: FFFFFFFF, PwdAck: 0000)
Response:	0001 (Result: true)

**1.5.11.12 MifareUltralightEV1\_CheckTearingEvent**

Command:	[0C0B][Byte: <i>CounterAddr</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte: <i>ValidFlag</i> ]
Example	
Command:	0C0B00 (CounterAddr: 00)
Response:	0001BD (Result: true, ValidFlag: 189)

**1.5.12 API ISO15693****1.5.12.1 ISO15693\_GenericCommand**

Command:	[0D00][Byte: <i>Flags</i> ][Byte: <i>Command</i> ][Byte Array(Var): <i>Data</i> ][Byte: <i>BufferSize</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>Data</i> ]
Example	
Command:	0D001020010020 (Flags: 10, Command: 20, Data: 00, BufferSize: 20)
Response:	000104000000000 (Result: true, Data: 00000000)



**1.5.12.2 ISO15693\_GetSystemInformation**

Command:	[0D01]
Response:	[00][Bool: <i>Result</i> ][Byte Array(15): <i>SystemInfo</i> ]
Example	
Command:	0D01
Response:	0001EF50781B06013C16E002000442000F (Result: true, SystemInfo: EF50781B06013C16E002000442000F)

**1.5.12.3 ISO15693\_GetSystemInformationExt**

Command:	[0D02]
Response:	[00][Bool: <i>Result</i> ][Byte Array(15): <i>SystemInfo</i> ]
Example	
Command:	0D02
Response:	0001EF7D50C3ED084402E0000004000844 (Result: true, SystemInfo: EF7D50C3ED084402E0000004000844)

**1.5.12.4 ISO15693\_GetTagTypeFromUID**

Command:	[0D03][Byte Array(8): <i>UID</i> ]
Response:	[00][Byte: <i>TagType</i> ]
Example	
Command:	0D03E0163C01061B7850 (UID: E0163C01061B7850)
Response:	00FF (TagType: 255)

**1.5.12.5 ISO15693\_GetTagTypeFromSystemInfo**

Command:	[0D04][Byte Array(15): <i>SystemInfo</i> ]
Response:	[00][Byte: <i>TagType</i> ]
Example	
Command:	0D04EF7D50C3ED084402E0000004000844 (SystemInfo: EF7D50C3ED084402E0000004000844)
Response:	0043 (TagType: 67)

**1.5.12.6 ISO15693\_ReadSingleBlock**

Command:	[0D05][UInt16: <i>BlockNumber</i> ][Byte: <i>BufferSize</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>BlockData</i> ]
Example	
Command:	0D050500FF (BlockNumber: 0500, BufferSize: FF)
Response:	000104000000000 (Result: true, BlockData: 00000000)

**1.5.12.7 ISO15693\_ReadSingleBlockExt**

Command:	[0D06][UInt16: <i>BlockNumber</i> ][Byte: <i>BufferSize</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>BlockData</i> ]
Example	
Command:	0D060000FF (BlockNumber: 0000, BufferSize: FF)
Response:	00010401020304 (Result: true, BlockData: 01020304)

**1.5.12.8 ISO15693\_WriteSingleBlock**

Command:	[0D07][UInt16: <i>BlockNumber</i> ][Byte Array(Var): <i>BlockData</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0D0705000411223344 (BlockNumber: 0500, BlockData: 11223344)
Response:	0001 (Result: true)

**1.5.12.9 ISO15693\_WriteSingleBlockExt**

Command:	[0D08][UInt16: <i>BlockNumber</i> ][Byte Array(Var): <i>BlockData</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0D08000004426C612E (BlockNumber: 0000, BlockData: 426C612E)
Response:	0001 (Result: true)



## 1.5.14 API DESFIRE

### 1.5.14.1 DESFire\_GetApplicationIDs

Command:	[0F00][Byte: <i>CryptoEnv</i> ][Byte: <i>MaxAIDCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][variable number of UInt32: <i>AIDs</i> ]
Example	
Command:	0F00001C (CryptoEnv: 00, MaxAIDCnt: 1C)
Response:	00010133221100 (Result: true, AIDs: 00112233)

### 1.5.14.2 DESFire\_CreateApplication

Command:	[0F01][Byte: <i>CryptoEnv</i> ][UInt32: <i>AID</i> ][4 Bit: <i>ChangeKeyAccessRights</i> ][1 Bit: <i>ConfigurationChangeable</i> ][1 Bit: <i>FreeCreateDelete</i> ][1 Bit: <i>FreeDirectoryList</i> ][1 Bit: <i>AllowChangeMasterKey</i> ][UInt32: <i>NumberOfKeys</i> ][UInt32: <i>KeyType</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F0100907856000F0100000000000000 (CryptoEnv: 00, AID: 90785600, ChangeKeyAccessRights: 15, ConfigurationChangeable: 1, FreeCreateDelete: 1, FreeDirectoryList: 1, AllowChangeMasterKey: 1, NumberOfKeys: 01000000, KeyType: 00000000)
Response:	0001 (Result: true)

### 1.5.14.3 DESFire\_DeleteApplication

Command:	[0F02][Byte: <i>CryptoEnv</i> ][UInt32: <i>AID</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F020090785600 (CryptoEnv: 00, AID: 90785600)
Response:	0001 (Result: true)



#### 1.5.14.7 DESFire\_GetFileIDs

Command:	[0F06][Byte: <i>CryptoEnv</i> ][Byte: <i>MaxFileIDCount</i> ]
Response:	[00][Bool: <i>Result</i> ][variable number of Bytes: <i>FileIDList</i> ]
Example	
Command:	0F0600FF (CryptoEnv: 00, MaxFileIDCount: FF)
Response:	00010400010203 (Result: true, FileIDList: 00, 01, 02, 03)

#### 1.5.14.8 DESFire GetFileSettings

[illegible]

#### 1.5.14.9 DESFire\_ReadData

Command:	[0F08][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][UInt16: <i>Offset</i> ][Byte: <i>Length</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>Data</i> ]
Example	
Command:	0F080000000000300 (CryptoEnv: 00, FileNo: 00, Offset: 0000, Length: 03, CommSet: 00)
Response:	000103001122 (Result: true, Data: 001122)

**1.5.14.10 DESFire\_WriteData**

Command:	[0F09][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][UInt16: <i>Offset</i> ][Byte Array(Var): <i>Data</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F09000000000300112200 (CryptoEnv: 00, FileNo: 00, Offset: 0000, Data: 001122, CommSet: 00)
Response:	0001 (Result: true)

**1.5.14.11 DESFire\_GetValue**

Command:	[0F0A][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ][UInt32: <i>Value</i> ]
Example	
Command:	0F0A000000 (CryptoEnv: 00, FileNo: 00, CommSet: 00)
Response:	000100000000 (Result: true, Value: 0)

**1.5.14.12 DESFire\_Credit**

Command:	[0F0B][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][UInt32: <i>Value</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F0B00040000000000 (CryptoEnv: 00, FileNo: 04, Value: 00000000, CommSet: 00)
Response:	0001 (Result: true)

**1.5.14.13 DESFire\_Debit**

Command:	[0F0C][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][UInt32: <i>Value</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F0C0004000000000000 (CryptoEnv: 00, FileNo: 04, Value: 00000000, CommSet: 00)
Response:	0001 (Result: true)

**1.5.14.14 DESFire\_LimitedCredit**

Command:	[0F0D][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][UInt32: <i>Value</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F0D0004000000000000 (CryptoEnv: 00, FileNo: 04, Value: 00000000, CommSet: 00)
Response:	0001 (Result: true)

**1.5.14.15 DESFire\_FreeMem**

Command:	[0F0E][Byte: <i>CryptoEnv</i> ]
Response:	[00][Bool: <i>Result</i> ][UInt16: <i>FreeMemory</i> ]
Example	
Command:	0F0E00 (CryptoEnv: 00)
Response:	00016011 (Result: true, FreeMemory: 4448)

**1.5.14.16 DESFire\_FormatTag**

Command:	[0F0F][Byte: <i>CryptoEnv</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F0F00 (CryptoEnv: 00)
Response:	0001 (Result: true)





**1.5.14.20 DESFire\_DeleteFile**

Command:	[0F13][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F130005 (CryptoEnv: 00, FileNo: 05)
Response:	0001 (Result: true)

**1.5.14.21 DESFire\_CommitTransaction**

Command:	[0F14][Byte: <i>CryptoEnv</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F1400 (CryptoEnv: 00)
Response:	0001 (Result: true)

**1.5.14.22 DESFire\_AbortTransaction**

Command:	[0F15][Byte: <i>CryptoEnv</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F1500 (CryptoEnv: 00)
Response:	0001 (Result: true)

**1.5.14.23 DESFire\_GetUID**

Command:	[0F16][Byte: <i>CryptoEnv</i> ][Byte: <i>BufferSize</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>UID</i> ]
Example	
Command:	0F1600FF (CryptoEnv: 00, BufferSize: FF)
Response:	000107045243523D2480 (Result: true, UID: 045243523D2480)

#### 1.5.14.24 DESFire\_GetKeyVersion

Command:	[0F17][Byte: <i>CryptoEnv</i> ][Byte: <i>KeyNo</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(1): <i>KeyVersion</i> ]
Example	
Command:	0F170000 (CryptoEnv: 00, KeyNo: 00)
Response:	0001FF (Result: true, KeyVersion: FF)

#### 1.5.14.25 DESFire\_ChangeKeySettings

Command:	[0F18][Byte: <i>CryptoEnv</i> ][4 Bit: <i>ChangeKeyAccessRights</i> ][1 Bit: <i>ConfigurationChangeable</i> ][1 Bit: <i>FreeCreateDelete</i> ][1 Bit: <i>FreeDirectoryList</i> ][1 Bit: <i>AllowChangeMasterKey</i> ][UInt32: <i>NumberOfKeys</i> ][UInt32: <i>KeyType</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F18000F000000000000000000 (CryptoEnv: 00, ChangeKeyAccessRights: 15, ConfigurationChangeable: 1, FreeCreateDelete: 1, FreeDirectoryList: 1, AllowChangeMasterKey: 1, NumberOfKeys: 00000000, KeyType: 00000000)
Response:	0001 (Result: true)

#### 1.5.14.26 DESFire\_ChangeKey

[illegible]

**1.5.14.27 DESFire\_ChangeFileSettings**

Command:	[0F1A][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][Byte: <i>NewCommSet</i> ][UInt16: <i>OldAccessRights</i> ][UInt16: <i>NewAccessRights</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F1A000000EEEEEEEE (CryptoEnv: 00, FileNo: 00, NewCommSet: 00, OldAccessRights: EEEE, NewAccessRights: EEEE)
Response:	0001 (Result: true)

**1.5.14.28 DESFire\_DisableFormatCard**

Command:	[0F1B][Byte: <i>CryptoEnv</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F1B00 (CryptoEnv: 00)
Response:	0001 (Result: true)

**1.5.14.29 DESFire\_EnableRandomID**

Command:	[0F1C][Byte: <i>CryptoEnv</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F1C00 (CryptoEnv: 00)
Response:	0001 (Result: true)



**1.5.14.33 DESFire\_ReadRecords**

Command:	[0F20][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][UInt16: <i>Offset</i> ][Byte: <i>NumberOfRecords</i> ][Byte: <i>RecordSize</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>Data</i> ]
Example	
Command:	0F2000000000030000 (CryptoEnv: 00, FileNo: 00, Offset: 0000, NumberOfRecords: 03, RecordSize: 00, CommSet: 00)
Response:	000103001122 (Result: true, Data: 001122)

**1.5.14.34 DESFire\_WriteRecord**

Command:	[0F21][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ][UInt16: <i>Offset</i> ][Byte Array(Var): <i>Data</i> ][Byte: <i>CommSet</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F21000000000300112200 (CryptoEnv: 00, FileNo: 00, Offset: 0000, Data: 001122, CommSet: 00)
Response:	0001 (Result: true)

**1.5.14.35 DESFire\_ClearRecordFile**

Command:	[0F22][Byte: <i>CryptoEnv</i> ][Byte: <i>FileNo</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	0F220005 (CryptoEnv: 00, FileNo: 05)
Response:	0001 (Result: true)

**1.5.15 API ISO7816****1.5.15.1 ISO7816\_GetSlotStatus**

Command:	[1000][Byte: <i>Channel</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(3): <i>SlotStatus</i> ]
Example	
Command:	100020 (Channel: 20)
Response:	0001000000 (Result: true, SlotStatus: 000000)

**1.5.15.2 ISO7816\_IccPowerOn**

Command:	[1001][Byte: <i>Channel</i> ][Byte: <i>MaxATRByteCnt</i> ][Byte: <i>bPowerSelect</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>ATR</i> ][Byte: <i>bStatus</i> ][Byte: <i>bError</i> ]
Example	
Command:	100120FF00 (Channel: 20, MaxATRByteCnt: FF, bPowerSelect: 00)
Response:	00010F3B959680B1FE551FC74772616365130000 (Result: true, ATR: 3B959680B1FE551FC7477261636513, bStatus: 0, bError: 0)

**1.5.15.3 ISO7816\_IccPowerOff**

Command:	[1002][Byte: <i>Channel</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(3): <i>SlotStatus</i> ]
Example	
Command:	100220 (Channel: 20)
Response:	0001010000 (Result: true, SlotStatus: 010000)

**1.5.15.4 ISO7816\_SetCommSettings**

Command:	[1003][Byte: <i>Channel</i> ][Byte Array(13): <i>CommSettings</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1003200100740101000000FF5500FE00 (Channel: 20, CommSettings: 0100740101000000FF5500FE00)
Response:	0001 (Result: true)

**1.5.15.5 ISO7816\_Transceive**

Command:	[1004][Byte: <i>Channel</i> ][Byte Array(Var), 2 LB: <i>TX</i> ][Byte: <i>MaxRXByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>RX</i> ]
Example	
Command:	100420050000C10120E0FF (Channel: 20, TX: 00C10120E0, MaxRXByteCnt: FF)
Response:	000102006E00 (Result: true, RX: 6E00)

**1.5.15.6 ISO7816\_ExchangeAPDU**

Command:	[1005][Byte: <i>Channel</i> ][Byte Array(9): <i>Header</i> ][Byte Array(Var), 2 LB: <i>TXData</i> ][UInt16: <i>MaxRXByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>RXData</i> ][UInt16: <i>StatusWord</i> ]
Example	
Command:	10052000A40004020000000102003F008000 (Channel: 20, Header: 00A400040200000001, TXData: 3F00, MaxRXByteCnt: 8000)
Response:	00010000006E (Result: true, RXData: , StatusWord: 28160)



**1.5.15.7 ISO7816\_T0\_TPDU**

Command:	[1006][Byte: <i>Channel</i> ][Byte Array(5): <i>Header</i> ][Byte Array(Var), 2 LB: <i>TXData</i> ][UInt16: <i>MaxRXByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>RXData</i> ][UInt16: <i>StatusWord</i> ]
Example	
Command:	10062000A400040202003F008000 (Channel: 20, Header: 00A4000402, TXData: 3F00, MaxRXByteCnt: 8000)
Response:	00010000006E (Result: true, RXData: , StatusWord: 28160)

**1.5.16 API ICLASS****1.5.16.1 ICLASS\_GetPACBits**

Command:	[1100][Byte: <i>MaxPACBytes</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte: <i>PACBitCnt</i> ][Byte Array(Var): <i>PAC</i> ]
Example	
Command:	1100FF (MaxPACBytes: FF)
Response:	00011A0405000980 (Result: true, PACBitCnt: 26, PAC: 00140026)

**1.5.17 API ISO14443****1.5.17.1 ISO14443A\_GetATS**

Command:	[1200][Byte: <i>MaxATSByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>ATS</i> ]
Example	
Command:	120020 (MaxATSByteCnt: 20)
Response:	000106067577810280 (Result: true, ATS: 067577810280)

**1.5.17.2 ISO14443B\_GetATQB**

Command:	[1201][Byte: <i>MaxATQBByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>ATQB</i> ]
Example	
Command:	1201FF (MaxATQBByteCnt: FF)
Response:	00010C5077FB135400000000B37171 (Result: true, ATQB: 5077FB135400000000B37171)

**1.5.17.3 ISO14443\_4\_CheckPresence**

Command:	[1202]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1202
Response:	0001 (Result: true)

**1.5.17.4 ISO14443\_4\_TDX**

Command:	[1203][Byte Array(Var): <i>TX</i> ][Byte: <i>MaxRXByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>RX</i> ]
Example	
Command:	1203016020 (TX: 60, MaxRXByteCnt: 20)
Response:	0001026F00 (Result: true, RX: 6F00)

**1.5.17.5 ISO14443A\_GetATQA**

Command:	[1204]
Response:	[00][Bool: <i>Result</i> ][Byte Array(2): <i>ATQA</i> ]
Example	
Command:	1204
Response:	00010403 (Result: true, ATQA: 0403)

**1.5.17.6 ISO14443A\_GetSAK**

Command:	[1205]
Response:	[00][Bool: <i>Result</i> ][Byte Array(1): <i>SAK</i> ]
Example	
Command:	1205
Response:	000120 (Result: true, SAK: 20)

**1.5.17.7 ISO14443B\_GetAnswerToATTRIB**

Command:	[1206][Byte: <i>MaxAnswerToATTRIBByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>AnswerToATTRIB</i> ]
Example	
Command:	1206FF (MaxAnswerToATTRIBByteCnt: FF)
Response:	00010100 (Result: true, AnswerToATTRIB: 00)

**1.5.17.8 ISO14443\_3\_TDX**

Command:	[1207][Byte Array(Var): <i>TX</i> ][Byte: <i>MaxRXByteCnt</i> ][UInt16: <i>Timeout</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>RX</i> ]
Example	
Command:	1207041A004176FFFF00 (TX: 1A004176, MaxRXByteCnt: FF, Timeout: FF00)
Response:	00010104 (Result: true, RX: 04)

**1.5.17.9 ISO14443A\_SearchMultiTag**

Command:	[1208][Byte: <i>MaxUIDListByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte: <i>UIDCnt</i> ][variable number of Bytes: <i>UIDList</i> ]
Example	
Command:	1208FF (MaxUIDListByteCnt: FF)
Response:	000103180704D7A79A97378007042DA79A973780070450A79A973780 (Result: true, UIDCnt: 3, UIDList: 04D7A79A973780, 042DA79A973780, 0450A79A973780)

**1.5.17.10 ISO14443A\_SelectTag**

Command:	[1209][Byte Array(Var): <i>UID</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	12090704D7A79A973780 (UID: 04D7A79A973780)
Response:	0001 (Result: true)

**1.5.18 API AT55****1.5.18.1 AT55\_Begin**

Command:	[1500]
Response:	[00]
Example	
Command:	1500
Response:	00

**1.5.18.2 AT55\_ReadBlock**

Command:	[1501][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>Data</i> ]
Example	
Command:	150100 (Address: 00)
Response:	0001F0148040 (Result: true, Data: F0148040)

**1.5.18.3 AT55\_ReadBlockProtected**

Command:	[1502][Byte: <i>Address</i> ][Byte Array(4): <i>Password</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>Data</i> ]
Example	
Command:	1502000000000000 (Address: 00, Password: 00000000)
Response:	0001B8A31C02 (Result: true, Data: B8A31C02)

**1.5.18.4 AT55\_WriteBlock**

Command:	[1503][Byte: <i>Address</i> ][Byte Array(4): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	15030000010203 (Address: 00, Data: 00010203)
Response:	0001 (Result: true)

**1.5.18.5 AT55\_WriteBlockProtected**

Command:	[1504][Byte: <i>Address</i> ][Byte Array(4): <i>Data</i> ][Byte Array(4): <i>Password</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1504000001020300000000 (Address: 00, Data: 00010203, Password: 00000000)
Response:	0001 (Result: true)

**1.5.18.6 AT55\_WriteBlockAndLock**

Command:	[1505][Byte: <i>Address</i> ][Byte Array(4): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	15050000010203 (Address: 00, Data: 00010203)
Response:	0001 (Result: true)

**1.5.18.7 AT55\_WriteBlockProtectedAndLock**

Command:	[1506][Byte: <i>Address</i> ][Byte Array(4): <i>Data</i> ][Byte Array(4): <i>Password</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1506000001020300000000 (Address: 00, Data: 00010203, Password: 00000000)
Response:	0001 (Result: true)

**1.5.19 API NFC SNEP****1.5.19.1 SNEP\_Init**

Command:	[1800]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1800
Response:	0001 (Result: true)

**1.5.19.2 SNEP\_GetConnectionState**

Command:	[1801]
Response:	[00][Byte: <i>ConnectionState</i> ]
Example	
Command:	1801
Response:	0002 (ConnectionState: 2)

**1.5.19.3 SNEP\_GetFragmentByteCount**

Command:	[1802][Byte: <i>Direction</i> ]
Response:	[00][UInt16: <i>ByteCount</i> ]
Example	
Command:	180201 (Direction: 01)
Response:	000000 (ByteCount: 0)

**1.5.19.4 SNEP\_BeginMessage**

Command:	[1803][UInt32: <i>MsgByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1803FF000000 (MsgByteCnt: FF000000)
Response:	0001 (Result: true)

**1.5.19.5 SNEP\_SendMessageFragment**

Command:	[1804][Byte Array(Var), 2 LB: <i>MsgFrag</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	18041500D101115501656C617465632D726669642E636F6D2F (MsgFrag: D101115501656C617465632D726669642E636F6D2F)
Response:	0001 (Result: true)

**1.5.19.6 SNEP\_TestMessage**

Command:	[1805]
Response:	[00][Bool: <i>Result</i> ][UInt32: <i>MsgByteCnt</i> ]
Example	
Command:	1805
Response:	0000 (Result: fail, MsgByteCnt: )

**1.5.19.7 SNEP\_ReceiveMessageFragment**

Command:	[1806][UInt16: <i>FragByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>MsgFrag</i> ]
Example	
Command:	1806FF00 (FragByteCnt: FF00)
Response:	0000 (Result: fail, MsgFrag: )

**1.5.19.8 SNEP\_RequestMessage**

Command:	[1807][UInt32: <i>MsgByteCnt</i> ][UInt32: <i>AcceptableLength</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1807FF000000FF000000 (MsgByteCnt: FF000000, AcceptableLength: FF000000)
Response:	0001 (Result: true)

## 1.5.20 API EM4150

### 1.5.20.1 EM4150\_Login

Command:	[1900][Byte Array(4): <i>Password</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	190000000000 (Password: 00000000)
Response:	0001 (Result: true)

### 1.5.20.2 EM4150\_ReadWord

Command:	[1901][Byte: <i>Address</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>Word</i> ]
Example	
Command:	190101 (Address: 01)
Response:	000100010203 (Result: true, Word: 00010203)

### 1.5.20.3 EM4150\_WriteWord

Command:	[1902][Byte: <i>Address</i> ][Byte Array(4): <i>Word</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	19020100010203 (Address: 01, Word: 00010203)
Response:	0001 (Result: true)



**1.5.20.4 EM4150\_WritePassword**

Command:	[1903][Byte Array(4): <i>ActualPassword</i> ][Byte Array(4): <i>NewPassword</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	190300000000001010101 (ActualPassword: 00000000, NewPassword: 01010101)
Response:	0001 (Result: true)

**1.5.20.5 EM4150\_GetTagInfo**

Command:	[1904]
Response:	[00][UInt32: <i>TagInfo</i> ]
Example	
Command:	1904
Response:	0001000000 (TagInfo: 1)

**1.5.21 API FILESYS****1.5.21.1 FSMount**

Command:	[1A00][Byte: <i>StorageID</i> ][UInt32: <i>Mode</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1A000102000000 (StorageID: 01, Mode: 02000000)
Response:	0001 (Result: true)

**1.5.21.2 FSFormat**

Command:	[1A01][Byte: <i>StorageID</i> ][UInt32: <i>MagicValue</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1A0101446F4974 (StorageID: 01, MagicValue: 446F4974)
Response:	0001 (Result: true)

**1.5.21.3 FSOpen**

Command:	[1A02][Byte: <i>FileEnv</i> ][Byte: <i>StorageID</i> ][UInt32: <i>FileID</i> ][Byte: <i>Mode</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1A0200013322110000 (FileEnv: 00, StorageID: 01, FileID: 33221100, Mode: 00)
Response:	0001 (Result: true)

**1.5.21.4 FSClose**

Command:	[1A03][Byte: <i>FileEnv</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1A0300 (FileEnv: 00)
Response:	0001 (Result: true)

**1.5.21.5 FSCloseAll**

Command:	[1A04]
Response:	[00]
Example	
Command:	1A04
Response:	00

**1.5.21.6 FSSeek**

Command:	[1A05][Byte: <i>FileEnv</i> ][Byte: <i>Origin</i> ][UInt32: <i>Pos</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1A05000001000000 (FileEnv: 00, Origin: 00, Pos: 01000000)
Response:	0001 (Result: true)

**1.5.21.7 FSTell**

Command:	[1A06][Byte: <i>FileEnv</i> ][Byte: <i>Origin</i> ]
Response:	[00][Bool: <i>Result</i> ][UInt32: <i>Pos</i> ]
Example	
Command:	1A060000 (FileEnv: 00, Origin: 00)
Response:	000101000000 (Result: true, Pos: 1)

**1.5.21.8 FSReadBytes**

Command:	[1A07][Byte: <i>FileEnv</i> ][UInt16: <i>ByteCount</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>Data</i> ]
Example	
Command:	1A07001E00 (FileEnv: 00, ByteCount: 1E00)
Response:	000107004D792064617461 (Result: true, Data: 4D792064617461)

**1.5.21.9 FSWriteBytes**

Command:	[1A08][Byte: <i>FileEnv</i> ][Byte Array(Var), 2 LB: <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ][UInt16: <i>BytesWritten</i> ]
Example	
Command:	1A080007004D792064617461 (FileEnv: 00, Data: 4D792064617461)
Response:	00010700 (Result: true, BytesWritten: 7)

**1.5.21.10 FSFindFirst**

Command:	[1A09][Byte: <i>StorageID</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>FileInfo</i> ]
Example	
Command:	1A0901 (StorageID: 01)
Response:	00013322110002000000 (Result: true, FileInfo: 3322110002000000)

**1.5.21.11 FSFindNext**

Command:	[1A0A]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>FileInfo</i> ]
Example	
Command:	1A0A
Response:	00013422110002000000 (Result: true, FileInfo: 3422110002000000)

**1.5.21.12 FSDelete**

Command:	[1A0B][Byte: <i>StorageID</i> ][UInt32: <i>FileID</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1A0B0133221100 (StorageID: 01, FileID: 33221100)
Response:	0001 (Result: true)

**1.5.21.13 FSRename**

Command:	[1A0C][Byte: <i>StorageID</i> ][UInt32: <i>OldFileID</i> ][UInt32: <i>NewFileID</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1A0C017766554433221100 (StorageID: 01, OldFileID: 77665544, NewFileID: 33221100)
Response:	0001 (Result: true)

#### 1.5.21.14 FSGetStorageInfo

Command:	[1A0D][Byte: <i>StorageID</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(9): <i>StorageInfo</i> ]
Example	
Command:	1A0D01 (StorageID: 01)
Response:	000101204B00000004B0000 (Result: true, StorageInfo: 01204B00000004B0000)

### 1.5.22 API MIFAREPLUS

#### 1.5.22.1 MFP\_WritePerso

Command:	[1B00][UInt16: <i>BlockNr</i> ][Byte Array(16): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B00004000 (BlockNr: 0040, Data: 00)
Response:	0001 (Result: true)

### 1.5.22.2 MFP\_CommitPerso

Command:	[1B01]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B01
Response:	0001 (Result: true)

### 1.5.22.3 MFP Authenticate

Command:	[1B02][Byte: <i>CryptoEnv</i> ][UInt16: <i>KeyBNr</i> ][Byte Array(16): <i>Key</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B0200004000000000000000000000000000000000000000 (CryptoEnv: 00, KeyBNr: 0040, Key: 00000000000000000000000000000000)
Response:	0001 (Result: true)

#### 1.5.22.4 MFP ReadBlock

Command:	[1B03][Byte: <i>CryptoEnv</i> ][UInt16: <i>Block</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>Data</i> ]
Example	
Command:	1B03000400 (CryptoEnv: 00, Block: 0400)
Response:	000101020304050607080900010203040506 (Result: true, Data: 01020304050607080900010203040506)

#### 1.5.22.5 MFP WriteBlock

Command:	[1B04][Byte: <i>CryptoEnv</i> ][UInt16: <i>Block</i> ][Byte Array(16): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B0400040001020304050607080900010203040506 (CryptoEnv: 00, Block: 0400, Data: 01020304050607080900010203040506)
Response:	0001 (Result: true)

#### 1.5.22.6 MFP ReadValueBlock

Command:	[1B05][Byte: <i>CryptoEnv</i> ][UInt16: <i>Block</i> ]
Response:	[00][Bool: <i>Result</i> ][UInt32: <i>Value</i> ]
Example	
Command:	1B05000400 (CryptoEnv: 00, Block: 0400)
Response:	000100000000 (Result: true, Value: 0)

**1.5.22.7 MFP\_WriteValueBlock**

Command:	[1B06][Byte: <i>CryptoEnv</i> ][UInt16: <i>Block</i> ][UInt32: <i>Value</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B0600040000000000 (CryptoEnv: 00, Block: 0400, Value: 00000000)
Response:	0001 (Result: true)

**1.5.22.8 MFP\_IncrementValueBlock**

Command:	[1B07][Byte: <i>CryptoEnv</i> ][UInt16: <i>Block</i> ][UInt32: <i>Value</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B0700040001000000 (CryptoEnv: 00, Block: 0400, Value: 01000000)
Response:	0001 (Result: true)

**1.5.22.9 MFP\_DecrementValueBlock**

Command:	[1B08][Byte: <i>CryptoEnv</i> ][UInt16: <i>Block</i> ][UInt32: <i>Value</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B0800040001000000 (CryptoEnv: 00, Block: 0400, Value: 01000000)
Response:	0001 (Result: true)

**1.5.22.10 MFP\_CopyValueBlock**

Command:	[1B09][Byte: <i>CryptoEnv</i> ][UInt16: <i>SourceBlock</i> ][UInt16: <i>DestBlock</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1B090004000500 (CryptoEnv: 00, SourceBlock: 0400, DestBlock: 0500)
Response:	0001 (Result: true)

### 1.5.23 API ADC

#### 1.5.23.1 ADCInitChannel

Command:	[1C00][Byte: <i>ADCChannel</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1C0001 (ADCChannel: 01)
Response:	0001 (Result: true)

#### 1.5.23.2 ADCGetConversionValue

Command:	[1C01][Byte: <i>ADCChannel</i> ]
Response:	[00][UInt16: <i>Value</i> ]
Example	
Command:	1C0101 (ADCChannel: 01)
Response:	003700 (Value: 55)

### 1.5.24 API FELICA

#### 1.5.24.1 FeliCa\_TDX

Command:	[1D00][Byte Array(Var): <i>TX</i> ][Byte: <i>MaxRXByteCnt</i> ][Byte: <i>MaximumResponseTime</i> ][Byte: <i>NumberOfBlocks</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>RX</i> ]
Example	
Command:	1D00060600FFFF0000FFFF04 (TX: 0600FFFF0000, MaxRXByteCnt: FF, MaximumResponseTime: FF, NumberOfBlocks: 04)
Response:	000112120101010701450F16000120220427674EFF (Result: true, RX: 120101010701450F16000120220427674EFF)





**1.5.24.5 FeliCa\_Poll**

Command:	[1D04][UInt16: <i>SystemCode</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>IDm</i> ][Byte Array(8): <i>PMm</i> ]
Example	
Command:	1D04FFFF (SystemCode: FFFF)
Response:	0001011603002D0CA50B03014B024F4993FF (Result: true, IDm: 011603002D0CA50B, PMm: 03014B024F4993FF)

**1.5.24.6 FeliCa\_RequestService**

Command:	[1D05][variable number of UInt16: <i>ServiceCodeList</i> ]
Response:	[00][Bool: <i>Result</i> ][variable number of UInt16: <i>KeyVersionList</i> ]
Example	
Command:	1D05010000 (ServiceCodeList: 0000)
Response:	0001010100 (Result: true, KeyVersionList: 0001)

**1.5.25 API SLE44XX****1.5.25.1 SLE\_GetATR**

Command:	[1F00][Byte: <i>Channel</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>ATR</i> ]
Example	
Command:	1F0028 (Channel: 28)
Response:	0001FFFFFFFF (Result: true, ATR: FFFFFFFFFF)

**1.5.25.2 SLE\_ReadMainMemory**

Command:	[1F01][Byte: <i>Channel</i> ][UInt16: <i>Address</i> ][UInt16: <i>ByteCnt</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>Data</i> ]
Example	
Command:	1F012800000100 (Channel: 28, Address: 0000, ByteCnt: 0100)
Response:	00010100FF (Result: true, Data: FF)

**1.5.25.3 SLE\_UpdateMainMemory**

Command:	[1F02][Byte: <i>Channel</i> ][UInt16: <i>Address</i> ][Byte: <i>Value</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1F0228000000 (Channel: 28, Address: 0000, Value: 00)
Response:	0001 (Result: true)

**1.5.25.4 SLE\_ReadSecurityMemory**

Command:	[1F03][Byte: <i>Channel</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>SecMemData</i> ]
Example	
Command:	1F0328 (Channel: 28)
Response:	0001FFFFFFFF (Result: true, SecMemData: FFFFFFFFFF)

**1.5.25.5 SLE\_UpdateSecurityMemory**

Command:	[1F04][Byte: <i>Channel</i> ][Byte: <i>Address</i> ][Byte: <i>SecMemData</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1F042800FF (Channel: 28, Address: 00, SecMemData: FF)
Response:	0001 (Result: true)

**1.5.25.6 SLE\_ReadProtectionMemory**

Command:	[1F05][Byte: <i>Channel</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>ProtMemData</i> ]
Example	
Command:	1F0528 (Channel: 28)
Response:	0001FFFFFFFF (Result: true, ProtMemData: FFFFFFFF)

**1.5.25.7 SLE\_WriteProtectionMemory**

Command:	[1F06][Byte: <i>Channel</i> ][Byte: <i>Address</i> ][Byte: <i>ProtMemData</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1F062800FF (Channel: 28, Address: 00, ProtMemData: FF)
Response:	0001 (Result: true)

**1.5.25.8 SLE\_CompareVerificationData**

Command:	[1F07][Byte: <i>Channel</i> ][Byte: <i>Address</i> ][Byte: <i>VerificationData</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	1F072800FF (Channel: 28, Address: 00, VerificationData: FF)
Response:	0001 (Result: true)

**1.5.26 API NTAG****1.5.26.1 NTAG\_Read**

Command:	[2000][Byte: <i>Page</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>Page</i> ]
Example	
Command:	200004 (Page: 04)
Response:	000103B691028C537091016855016E78702E (Result: true, Page: 03B691028C537091016855016E78702E)

**1.5.26.2 NTAG\_Write**

Command:	[2001][Byte: <i>Page</i> ][Byte Array(4): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	20010400000000 (Page: 04, Data: 00000000)
Response:	0001 (Result: true)

**1.5.26.3 NTAG\_FastRead**

Command:	[2002][Byte: <i>StartPage</i> ][Byte: <i>NumberOfPages</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>Data</i> ]
Example	
Command:	20020401 (StartPage: 04, NumberOfPages: 01)
Response:	00010403B69102 (Result: true, Data: 03B69102)

**1.5.26.4 NTAG\_ReadCounter**

Command:	[2003]
Response:	[00][Bool: <i>Result</i> ][UInt32: <i>CounterValue</i> ]
Example	
Command:	2003
Response:	000101000000 (Result: true, CounterValue: 1)

**1.5.26.5 NTAG\_ReadSig**

Command:	[2004]
Response:	[00][Bool: <i>Result</i> ][Byte Array(32): <i>ECCSig</i> ]
Example	
Command:	2004
Response:	0001A9AC15AFB52080BA26A45B1DA442F363E31B41271AB12B3E6F67- 864615B05321 (Result: true, ECCSig: A9AC15AFB52080BA26A45B1DA442F363E31B41271AB12B3E6F67864615B05321)

**1.5.26.6 NTAG\_GetVersion**

Command:	[2005]
Response:	[00][Bool: <i>Result</i> ][Byte Array(8): <i>Version</i> ]
Example	
Command:	2005
Response:	00010004040502011503 (Result: true, Version: 0004040502011503)

**1.5.26.7 NTAG\_PwdAuth**

Command:	[2006][Byte Array(4): <i>Password</i> ][Byte Array(2): <i>PwdAck</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	2006FFFFFFFF0000 (Password: FFFFFFFF, PwdAck: 0000)
Response:	0001 (Result: true)

**1.5.26.8 NTAG\_SectorSelect**

Command:	[2007][Byte: <i>Sector</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	200700 (Sector: 00)
Response:	0001 (Result: true)

**1.5.27 API SRX****1.5.27.1 SRX\_ReadBlock**

Command:	[2100][Byte: <i>Block</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(4): <i>Data</i> ]
Example	
Command:	210000 (Block: 00)
Response:	000100000000 (Result: true, Data: 00000000)

**1.5.27.2 SRX\_WriteBlock**

Command:	[2101][Byte: <i>Block</i> ][Byte Array(4): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	21010000000000 (Block: 00, Data: 00000000)
Response:	0001 (Result: true)





**1.5.30 API SPI****1.5.30.1 SPINinit**

Command:	[2400][Byte: <i>Mode</i> ][Byte: <i>CPOL</i> ][Byte: <i>CPHA</i> ][Byte: <i>ClockRate</i> ][Byte: <i>BitOrder</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	240001000000000 (Mode: 01, CPOL: 00, CPHA: 00, ClockRate: 00, BitOrder: 00)
Response:	0001 (Result: true)

**1.5.30.2 SPIDelinit**

Command:	[2401]
Response:	[00]
Example	
Command:	2401
Response:	00

**1.5.30.3 SPIMasterBeginTransfer**

Command:	[2402]
Response:	[00]
Example	
Command:	2402
Response:	00

**1.5.30.4 SPIMasterEndTransfer**

Command:	[2403]
Response:	[00]
Example	
Command:	2403
Response:	00

**1.5.30.5 SPITransmit**

Command:	[2404][Byte Array(Var), 2 LB: <i>TXData</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	2404010000 (TXData: 00)
Response:	0001 (Result: true)

**1.5.30.6 SPIReceive**

Command:	[2405][UInt16: <i>ByteCount</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>RXData</i> ]
Example	
Command:	24050100 (ByteCount: 0100)
Response:	000101005A (Result: true, RXData: 5A)

**1.5.30.7 SPITransceive**

Command:	[2406][Byte Array(Var), 2 LB: <i>TXData</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var), 2 LB: <i>RXData</i> ]
Example	
Command:	2406010000 (TXData: 00)
Response:	000101005A (Result: true, RXData: 5A)

### 1.5.31 API BLE

#### 1.5.31.1 BLEPresetConfig

Command:	[2500][Byte Array(17): <i>BLEConfig</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	2500881300000A01A0000702020000D2040000 (BLEConfig: 881300000A01A0000702020000D2040000)
Response:	0001 (Result: true)

#### 1.5.31.2 BLEPresetUserData

Command:	[2501][Byte: <i>ScanResp</i> ][Byte Array(Var): <i>UserData</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	2501001E0201061AFF4C000215E2C56DB5DFFB48D2B060D0F5A71096- E000000000C3 (ScanResp: 00, UserData: 0201061AFF4C000215E2C56DB5DFFB48D2B060D0F5A71096E000000000C3)
Response:	0001 (Result: true)

#### 1.5.31.3 BLEInit

Command:	[2502][Byte: <i>Mode</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	250200 (Mode: 00)
Response:	0001 (Result: true)

**1.5.31.4 BLECheckEvent**

Command:	[2503]
Response:	[00][UInt32: <i>Event</i> ]
Example	
Command:	2503
Response:	0072000000 (Event: 114)

**1.5.31.5 BLEGetAddress**

Command:	[2504]
Response:	[00][Bool: <i>Result</i> ][Byte Array(6): <i>DeviceAddress</i> ][Byte Array(6): <i>RemoteAddress</i> ][Byte Array(1): <i>RemoteType</i> ]
Example	
Command:	2504
Response:	000122F60B570B000000000000000000 (Result: true, DeviceAddress: 22F60B570B00, RemoteAddress: 000000000000, RemoteType: 00)

**1.5.31.6 BLEGetVersion**

Command:	[2505]
Response:	[00][Bool: <i>Result</i> ][Byte Array(16): <i>HWVersion</i> ][Byte Array(12): <i>BootString</i> ]
Example	
Command:	2505
Response:	000156312E30332C31342E31312E3230313601000000040031040000-0100 (Result: true, HWVersion: 56312E30332C31342E31312E32303136, BootString: 010000000400310400000100)

**1.5.31.7 BLEGetEnvironment**

Command:	[2506]
Response:	[00][Bool: <i>Result</i> ][Byte Array(1): <i>DeviceRole</i> ][Byte Array(1): <i>SecurityMode</i> ][Byte Array(1): <i>Rssi</i> ]
Example	
Command:	2506
Response:	0001000000 (Result: true, DeviceRole: 00, SecurityMode: 00, Rssi: 00)

**1.5.31.8 BLEGetGattServerAttributeValue**

Command:	[2507][Byte: <i>AttrHandle</i> ][Byte: <i>MaxLen</i> ]
Response:	[00][Bool: <i>Result</i> ][Byte Array(Var): <i>Data</i> ]
Example	
Command:	25071314 (AttrHandle: 13, MaxLen: 14)
Response:	000100 (Result: true, Data: )

**1.5.31.9 BLESetGattServerAttributeValue**

Command:	[2508][Byte: <i>AttrHandle</i> ][Byte: <i>Offset</i> ][Byte Array(Var): <i>Data</i> ]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	2508170005000000000000 (AttrHandle: 17, Offset: 00, Data: 000000000000)
Response:	0001 (Result: true)

**1.5.31.10 BLERequestRssi**

Command:	[2509]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	2509
Response:	0001 (Result: true)

**1.5.31.11 BLERequestEndpointClose**

Command:	[250A]
Response:	[00][Bool: <i>Result</i> ]
Example	
Command:	250A
Response:	0001 (Result: true)

## 2 Disclaimer

Elatec reserves the right to change any information or data in this document without prior notice. The distribution and the update of this document is not controlled. Elatec declines all responsibility for the use of product with any other specifications but the ones mentioned above. Any additional requirement for a specific custom application has to be validated by the customer himself at his own responsibility. Where application information is given, it is only advisory and does not form part of the specification.

All referenced brands, product names, service names and trademarks mentioned in this document are the property of their respective owners.