CTAPI Driver
Shared Library

V 1.1

05.02.2004

Jochen Hammer
Handheld Competence

APIs

API - Overview

1. Err **CTAPIInit**(UInt16 refNum, UInt16 baudrate)
2. Err **CTAPIClose**(UInt16 refNum)
3. *Err CTAPISleep(UInt16 refNum)*
4. *Err CTAPIWake(UInt16 refNum)*
5. Err **CTAPISerOpen**(UInt16 refNum)
6. Err **CTAPISerClose**(UInt16 refNum)
7. Err **CTAPISerReConfigure**(UInt16 refNum)
8. Err **CTAPIRead**(UInt16 refNum, CTAPI_ReadWriteDataType* rdt)
9. Err **CTAPISend**(UInt16 refNum, CTAPI_ReadWriteDataType* rdt)
10. Err **CTAPIData**(UInt16 refNum, CTAPI_DataRequestType* drt)
11. Err **CTAPIDataExt**(UInt16 refNum, CTAPI_DataRequestType* drt)
12. Err **CTAPIGetATR**(UInt16 refNum, Char* atr, UInt8* atrLength)
13. Err **CTAPIGetATRHistChars**(UInt16 refNum, Char* atrHistChars, UInt8* atrHistCharsLength)
14. Err **CTAPIGetATR_TSChar**(UInt16 refNum, Char* tsChar)
15. Err **CTAPIReadATRFromCard**(UInt16 refNum,CTAPI_ReadWriteDataType* rdt, UInt8 atrTimeout)
16. UInt8 **CTAPIGetProtocolType**(UInt16 refNum)
17. Boolean **CTAPIIsSerOpen**(UInt16 refNum)
18. Boolean **CTAPIGetATRStatus**(UInt16 refNum)

Description

1. Err **CTAPIInit**(UInt16* refNum, UInt16 baudrate)

Check for presence of library and open it, if available. Set up and initalise global variables for library.

ATTENTION: This function is not integrated in the library, but as an inline function in the CTAPI Header File <CTAPI.h>.

Parameters:

← refNum contains the reference number if the CTAPI has been opened successfully. Use this reference Number for all Library calls.

→ baudrate the baudrate which shall be used for the communication. If this parameter is 0, the default baudrate is used.

2. Err **CTAPIClose**(UInt16 refNum)

Close the serial interface if it is open, close the library and remove from memory. Free the libraries global variables.

ATTENTION: This function is not integrated in the library, but as an inline function in the CTAPI Header File <CTAPI.h>.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

*3. Err **CTAPISleep**(UInt16 refNum)*

Not implemented

*4. Err **CTAPIWake**(UInt16 refNum)*

Not implemented

5. Err **CTAPISerOpen**(UInt16 refNum)

Open the serial interface with the baudrate, which has been passed in the CTAPIOpen call. Returns an error if serial is already off.

The following protocol is used:

8 Data-Bits

Even parity

1 Stop-Bit

RTS/CTS Auto-Mode

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

6. Err **CTAPISerClose**(UInt16 refNum)

Close the serial interface. Returns an error if serial is already off.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

7. Err **CTAPISerReConfigure**(UInt16 refNum)

Flush the send and receive buffer and clear any pending error in the serial interface. Re-Initalise the serial interface to the following protocol:

8 Data-Bits

Even parity

1 Stop-Bit

RTS/CTS Auto-Mode

ATTENTION: The Serial Interface has to be open for this call.

Returns error if serial is not open.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

8. Err **CTAPIRead**(UInt16 refNum, CTAPI_ReadWriteDataType* rdt)

Read data from the serial interface if available. Pass the required reading mode in the data structure. If RAW mode is selected, all received data will be passed to the caller.

This function shall be used to check if there is some data available. If the return status (err) is errNone and rdt.numBytes is 0, no data is available.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

←→ rdt Structure for receiving data:

mode	→ pass the mode which shall be used for receiving data: RAW_MODE or FRAMING_MODE
buffer	→ pass pointer to buffer. This buffer is filled up with the received data
numBytes	← number of received bytes
preselectByte	← contains preselectByte if data has been received (FRAMING_MODE only, 0 otherwise)

9. Err **CTAPISend**(UInt16 refNum, CTAPI_ReadWriteDataType* rdt)

Send data to the serial interface. Pass the required mode in the data structure. If RAW mode is selected, all data from the caller will be send to the serial interface without any modification.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

←→ rdt Structure for sending data:

mode → pass the mode which shall be used for receiving data:
RAW_MODE or FRAMING_MODE
buffer → pass pointer to send buffer. This buffer is send
numBytes → number of bytes which shall be send
preselectByte → contains preselectByte which shall be used in FRAMING mode. Ignore in RAW mode.

10. Err **CTAPIData**(UInt16 refNum, CTAPI_DataRequestType* drt)

This function is used to communicate with the processor or memory card. Pass the command to the card and it returns with the response or a 'timeout from card' error. This function allows communication in FRAMING mode and RAW mode.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

←→ drt Structure for requesting data:

preselectByte ←→ pass preselectByte which shall be used in FRAMING mode. Ignore in RAW mode. On return it contains the preselectByte which has been received from the card in framing mode.

mode → pass the mode which shall be used for receiving data:
RAW_MODE or FRAMING_MODE

command → pass pointer to command buffer. This buffer is send to the card

commandLength → number of bytes of the command

response → pass pointer to response buffer. This buffer is filled up with the response data.

numBytes ← number of received bytes from the response

timeout → timeout in ticks. The CTAPI waits for this number of ticks after sending data to check if a response has been received. If 0 the default value is used.

11. Err **CTAPIDataExt**(UInt16 refNum, CTAPI_DataRequestType* drt)

This function is used to communicate with processor cards using the ISO 7816-4 communication (protocol T0/T1). Before this command can be used a vaild ATR has to be received and must be available. If no valid ATR is available, this function returns an error. Based on the ATR this function will use the T0 or T1 protocol. Therefore the *CTAPIReadATRFromCard* has to be called successfully before this function is used the first time.

Pass the command to the card and it returns with the response or a 'timeout from card' error. This function allows communication in FRAMING mode and RAW mode.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

←→ drt Structure for requesting data:

preselectByte ←→ pass preselectByte which shall be used in FRAMING mode. Ignore

in RAW mode. On return it contains the preselectByte which has been received from the card in framing mode.

mode → pass the mode which shall be used for receiving data: RAW_MODE or FRAMING_MODE

command → pass pointer to command buffer. This buffer is send to the card

commandLength → number of bytes of the command

response → pass pointer to response buffer. This buffer is filled up with the response data.

numBytes ← number of received bytes from the response

timeout → timeout in ticks. The CTAPI waits for this number of ticks after sending data to check if a response has been received. If 0 the default value is used.

12. Err **CTAPIReadATRFromCard**(UInt16 refNum,CTAPI_ReadWriteDataType* rdt, UInt8 atrTimeout)

This function reads the ATR from processor cards. If the serial interface is already open, this function closes it first and re-open again to receive the ATR. If the serial interface is off when calling this function, the serial is opened, the ATR is received – if possible – and the serial interface is closed again. The received ATR is stored in the library and passed to the caller.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

←→ rdt Structure for receiving the ATR:

mode → pass the mode which shall be used for receiving data: RAW_MODE or FRAMING_MODE

buffer → pass pointer to receive buffer. IF NULL the ATR will be stored in the library globals only.

numBytes → length of ATR data, which has been received

preselectByte ← contains preselectByte which has been received in FRAMING mode. Ignore in RAW mode.

→ atrTimeout time in ticks to wait for a ATR from the card

13. Err **CTAPIGetATR**(UInt16 refNum, Char* atr, UInt8* atrLength)

This function returns the ATR which is stored in the library. It does not read the ATR from the card. So the *CTAPIReadATRFromCard* has to be called successfully before this function is used the first time.

Parameters:

→ refNum reference number, which has been passed from *CTAPIOpen*

← atr pointer to buffer where the ATR is returned. **ATTENTION**: The buffer must be large enough to contain a full ATR.

← atrLength length of the ATR

14. Err **CTAPIGetATRHistChars**(UInt16 refNum, Char* atrHistChars, UInt8* atrHistCharsLength)

This function returns the ATR Historical characters which are stored in the library. It does not read the ATR from the card. So the *CTAPIReadATRFromCard* has to be called successfully before this function is used the first time.

Parameters:

→ refNum	reference number, which has been passed from *CTAPIOpen*
← atrHistChars	pointer to buffer where the ATR Historical characters are returned. **ATTENTION**: The buffer must be large enough to contain all data.
← atrHistCharsLength	length of the ATR

15. Err **CTAPIGetATR_TSChar**(UInt16 refNum, Char* tsChar)

This function returns the ATR TS character which is stored in the library. It does not read the ATR from the card. So the *CTAPIReadATRFromCard* has to be called successfully before this function is used the first time.

Parameters:

→ refNum	reference number, which has been passed from *CTAPIOpen*
← tsChar	pointer to char where the TS character is stored.

16. UInt8 CTAPIGetProtocolType(UInt16 refNum)

This function returns the ATR protocol type which is stored in the library. It does not read the ATR from the card. So the *CTAPIReadATRFromCard* has to be called successfully before this function is used the first time.

Parameters:

→ refNum	reference number, which has been passed from *CTAPIOpen*
Returns:	0 or 1 indicating the protocol or 0xFF if error

17. Boolean **CTAPIIsSerOpen**(UInt16 refNum)

This function returns a boolean value indicating if the serial interface is on or off:

Returns:

True Serial interface is on

False Serial interface is off

Parameters:

→ refNum	reference number, which has been passed from *CTAPIOpen*

18. Boolean **CTAPIGetATRStatus**(UInt16 refNum)

This function returns a boolean value indicating if an ATR is available in the library globals. It does not read the ATR from the card. So the *CTAPIReadATRFromCard* has to be called successfully before this function is used the first time.

Returns:
True ATR available
False ATR available

Parameters:
→ refNum reference number, which has been passed from *CTAPIOpen*

Data structures

struct **CTAPI_ReadWriteDataType**

```
{
        UInt8  mode;
        UInt8* buffer;
        UInt16 numBytes;
        UInt8  preselectByte;
}
```

Used for sending or receiving data to/from the card reader. Pass RAW_MODE or FRAMING_MODE in structures mode variable to communicate with the desired communication protocol. The preselectByte variable is used to indicate the type of card in FRAMING_MODE. If sending data, pass a pointer to a databuffer which shall be send and the length of the buffer in numBytes. If receiving data, pass a pointer to a data buffer which shall be filled up with the received data. If receiving, the numBytes variable contains the number of received bytes.

struct **CTAPI_DataRequestType**

```
{
        UInt8   preselectByte;
        UInt8   mode;
        UInt8*  command;
        UInt16  commandLength;
        UInt8*  response;
        UInt16  responseLength;
        UInt8   timeout;
}
```

Used for general communication with the card reader. Pass RAW_MODE or FRAMING_MODE in structures mode variable to communicate with the desired communication protocol. The preselectByte variable is used to indicate the type of card in FRAMING_MODE. The timeout value specifies the time the driver waits to receive a response from the card.

Error Codes

CTAPIErrParam
CTAPIErrStillOpen
CTAPIErrSevere
CTAPIErrWrongOS
CTAPIErrRefNumIllegal
CTAPIErrNullPointer
CTAPIErrLowMemory
CTAPIErrSerialAlreadyOn
CTAPIErrSerialIsOff
CTAPIErrNoCommunication
CTAPIErrFramingChksum
CTAPIErrModeUnknown
CTAPIErrTimeout
CTAPIErrNoATRavailable
CTAPIErrATRTimeout
CTAPIErrATRProtocolUnsupported
CTAPIErrATRInvalid
CTAPIErrCommunication

The serial interface responds in error conditions with one of the following error codes:

serLineErrorParity
serLineErrorHWOverrun
serLineErrorFraming
serLineErrorBreak
serLineErrorHShake
serLineErrorSWOverrun
serLineErrorCarrierLost

The reader itself responds in error conditions with one of the following error codes:
CTAPI_READER_TIMEOUT
CTAPI_READER_PARITY
CTAPI_READER_XOR
CTAPI_READER_PRESELECT
CTAPI_READER_BUFFEROVERFLOW
CTAPI_READER_CARD_MISSING
CTAPI_READER_CARD_NOT_SUPPORTED

CTAPI_READER_COMMAND_UNKNOWN

CTAPI_READER_INVALID_RANGE

CTAPI_READER_OUT_OF_ADDRESS