OMNIKEY AG

SCardI2C Component

Documentation

Version 1.0

Author: MP
Date: 12/14/04

Contents

HOWTO: start using I2C cards

Description

Note

For more information about the parameters, use MSDN.

First of all you have to call the function SCardEstablishContext to establish the resource manager context

```
LONG SCardEstablishContext(
    IN  DWORD dwScope,
    IN  LPCVOID pvReserved1,
    IN  LPCVOID pvReserved2,
    OUT LPSCARDCONTEXT phContext
);
```

If these function returns SCARD_S_SUCCESS, the context has been successfully created. Now you are able to list all installed readers. To do this you need SCardListReaders.

```
LONG SCardListReaders(
    IN SCARDCONTEXT hContext,
    IN LPCTSTR mszGroups,
    OUT LPTSTR mszReaders,
    IN OUT LPDWORD pcchReaders
);
```

After you have listed the readers you have to select one of them. You can to that through a dialog or in your application.
Then you are able to connect to the card with SCardConnect.

```
LONG SCardConnect(
    IN SCARDCONTEXT hContext,
    IN LPCTSTR szReader,
    IN DWORD dwShareMode,
    IN DWORD dwPreferredProtocols,
    OUT LPSCARDHANDLE phCard,
    OUT LPDWORD pdwActiveProtocol
);
```

dwPreferredProtocols has to be SCARD_PROTOCOL_T0 cause of our implementation.

After you finished this function successfully you are connected to the card.
Now you have to initialise it to do anything else.

```
OKERR ENTRY SCardI2CInit(
IN SCARDHANDLE ulHandleCard,
IN SCARD_I2C_CARD_PARAMETERS  * pCardParameters,
IN SCARD_I2C_TYPE Type
);
```

ulHandleCard has to be the cardhandle you got from SCardConnect.
Type is a predefined type of card, you can use a lot of predefined constants, look at

SCardI2CInit function.
If your type is NO_PREDEFINED_CARD_PARAMETERS you have to submit pCardParameters, else you could submit NULL.
pCardParameters is a pointer to a SCARD_I2C_CARD_PARAMETERS structure which contains these members:

```
ucNumberOfAddressBytes (default: 1)
ucPageSize (default: 8)
ulMemorySize (default: 256)
```

Note : A page size = 0 is equal to a page size = 256

Now you are able to read or write data.

If you finished working with the card you should close the connection.

```
LONG SCardDisconnect(
   IN SCARDHANDLE hCard,
   IN DWORD dwDisposition
);
```

Read data from I2C cards:

```
OKERR ENTRY SCardI2CReadData(
   IN SCARDHANDLE ulHandleCard,
   BYTE    * pbReadBuffer,
   ULONG ulReadBufferSize,
   ULONG ulAddress,
   ULONG ulBytesToRead
);
```

ulHandleCard has to contain the cardhandle you got from SCardConnect.
ulBytesToRead indicates how many bytes will be read from card.
ulAddress define the start offset where the function starts to read.
ulReadBufferSize contains the size of **pbReadBuffer**.
pbReadBuffer has to be a pointer to an array (bytearray) and contains the read memory from the card if function was successfull.

The following functions are available in the SCardI2C module included in this library:

- ***SCardI2CInit***
- ***SCardI2CReadData***
- ***SCardI2CWriteData***

SCardI2C - General Overview

Description
The following I2C bus cards are supported by **scardsyn.dll** shared library:

I2C cards from ST-Microelectronics:
- ST14C02C
- ST14C04C
- ST14E32
- M14C04
- M14C16
- M14C32
- M14C64
- M14128
- M14256

I2C cards from GEMplus:
- GFM2K
- GFM4K
- GFM32K

I2C cards from Atmel:
- AT24C01A
- AT24C02
- AT24C04
- AT24C08
- AT24C16
- AT24C164
- AT24C32
- AT24C64
- AT24C128
- AT24C256
- AT24CS128
- AT24CS256
- AT24C512
- AT24C1024

For full understanding of the operation and functions of these cards, please refer to the corresponding I2C bus card technical manual.

If the I2C bus card you intend to use is not among the predefined cards above, you have to allocate and initialize a card parameters structure and submit its address, when calling **SCardI2CInit** function.

I2C cards are supported by following CardMan:
- Cardman 2011
- Cardman 2020
- Cardman 4000
- Cardman 6020
- Cardman 3121
- Cardman 4040
- Cardman 3111
- Cardman 3621
- Cardman Smart@Link

Cardman Smart@Key
Cardman Smart@Bus
Serial Smart Card Reader
PC-Card Smart Card Reader
USB CCID Smart Card Reader

The following functions are available in the SCardI2C module included in this library:

- ***SCardI2CInit***
- ***SCardI2CReadData***
- ***SCardI2CWriteData***

Function SCardI2CInit

Prototype

OKERR ENTRY SCardI2CInit
    (
    IN SCARDHANDLE *ulHandleCard*,
    IN SCARD_I2C_CARD_PARAMETERS * *pCardParameters*,
    IN SCARD_I2C_TYPE *Type*
)

Description

The function **SCardI2CInit** initializes the card and protocol specific parameters of the driver for communication with the I2C bus card. It has to be called once before any use of **SCardI2CReadData** or **SCardI2CWriteData**.
There is no corresponding function in the card itself.
The card is specified with **Type**. In this case, the card parameters are internaly initialized according the corresponding manufacturer specification and **pCardParameters** pointer is not evaluated.
When **Type = NO_PREDEFINED_CARD_PARAMETERS**, each card parameter is defined in a *CardParameters* structure which has to be allocated and initialized by the calling application. Its address is submitted as **pCardParameters** pointer in the call of this function. *Note:* It is expected, that the card is already connected.

The following **Type** constants can be used :

ST-Microelectronics:
ST14C02C
ST14C04C
ST14E32
M14C04
M14C16
M14C32
M14C64
M14128
M14256

GEMplus:
GFM2K
GFM4K
GFM32K

Atmel:
AT24C01A
AT24C02
AT24C04
AT24C08
AT24C16
AT24C164
AT24C32
AT24C64
AT24C128
AT24C256
AT24CS128

AT24CS256
AT24C512
AT241024

Parameters

The following parameters need to be provided:

Parameter	Type	Description
ulHandleCard	in	Handle to a I2C bus card, provided from the "smart card resource manager" after connecting the card (SCardConnect)
pCardParameters	in	Pointer to a structure holding I2C card parameters. Used only if **Type = NO_PREDEFINED_CARD_PARAMETERS**
Type	in	Predefined type of the used I2C card.

Function SCardI2CReadData

Prototype

OKERR ENTRY SCardI2CReadData
(
IN SCARDHANDLE *ulHandleCard*,
BYTE * *pbReadBuffer*,
ULONG *ulReadBufferSize*,
ULONG *ulAddress*,
ULONG *ulBytesToRead*
)

Description

Reads number of bytes starting from the specified memory address **ulAddress** and saves the content in the read buffer pointed by **pbReadBuffer**.
The function uses internal I2C bus sequential read for number of bytes. A better speed performance will be acheived, when a block of bytes is read with one call of this function instead calling it in a loop reading 1 byte.

Parameters

The following parameters need to be provided:

Parameter	Type	Description
ulHandleCard	in	Handle (get from SCardConnect)
pbReadBuffer	in	Pointer to the buffer, where the data read from the card are to be stored
ulReadBufferSize	in	Size of the read buffer
ulAddress	in	Memory address to read from
ulBytesToRead	in	Number of bytes to be read from the address above

Return Values

This function returns the following:

Value	Description
OK Standard Error Codes	see header file ok.h

Function SCardI2CWriteData

Prototype

OKERR ENTRY SCardI2CWriteData
 (
 IN SCARDHANDLE *ulHandleCard*,
 BYTE * *pbWriteBuffer*,
 ULONG *ulWriteBufferSize*,
 ULONG *ulAddress*,
 ULONG *ulBytesToWrite*
)

Description

Writes number of bytes (**ulBytesToWrite**) to the specified card memory starting from **ulAddress**.
The data to be written is taken from the buffer pointed by **pbWriteBuffer**.
The function uses internal I2C bus sequential write of number of bytes. A better speed performance will be acheived, when a block of bytes is written with one call of this function instead calling it in a loop writing 1 byte.

Parameters

The following parameters need to be provided:

Parameter	**Type**	**Description**
ulHandleCard	in	Handle (get from SCardConnect)
pbWriteBuffer	in	Pointer to the write buffer, holding the data to be written
ulWriteBufferSize	in	Size of the write buffer
ulAddress	in	Memory address in the I2C card where to start writing
ulBytesToWrite	in	Number of bytes to be written in the address above

Return Values

This function returns the following:

Value	**Description**
OK Standard Error Codes	see header file ok.h