15370 Barranca Parkway
Irvine, CA 92618

OMNIKEY® 5025 CL

SOFTWARE DEVELOPER GUIDE

Contents

Copyright

Trademarks

HID GLOBAL, HID, the HID logo and OMNIKEY are the trademarks or registered trademarks of HID Global Corporation, or its licensors, in the U.S. and other countries.

Revision History

Date	Author	Description	Document Version
11/2013	T. Konieczny, M. Goralczyk	Initial Version	A.0

Contacts

North America	Europe, Middle East and Africa
15370 Barranca Parkway Irvine, CA 92618 USA Phone: 800 237 7769 Fax: 949 732 2120	Phoenix Road Haverhill, Suffolk CB9 7AE England Phone: +44 1440 714 850 Fax: +44 1440 714 840

Asia Pacific
19/F 625 King's Road North Point, Island East Hong Kong Phone: 852 3160 9800 Fax: 852 3160 4809 support.hidglobal.com

About this Guide

Purpose

This Developer Guide is for developers integrating contactless HID PROX storage cards using the OMNIKEY® 5025 CL.

Organization

1 Overview

1.1 Product Description

The OMNIKEY® 5025 CL opens new market opportunities for system integrators seeking simple reader integration and development using standard interfaces, such as CCID (Circuit Card Interface Device). This reader works without installing or maintaining device drivers; only an operating system driver, for example, Microsoft CCID driver is necessary.

The OMNIKEY 5025 CL features include supporting the common low frequency HID Prox card technology.

1.2 Features

- **CCID Support.** Removes the requirement to install drivers on standard operating systems to fully support capabilities of the reader board.
- **HID Prox.** Supports the common low frequency HID Prox card technology.
- **Rapid and Easy Integration.** No special driver installation is required.
- **Advanced Power Management.** Fully compliant to Low Power modes specified by USB include the following.
  - Allows the computer to turn off the reader to save power (while the reader is still able to read cards, with reduced power)
  - Allows the reader to wake the computer
- **OMNIKEY 5325 Legacy Mode.** Emulation of OMNIKEY 5325 reader

2 Getting Started

2.1 Driver Installation

As stated previously, no extra driver installation is necessary and every CCID compliant driver should work with the reader. However, Microsoft's CCID driver prevents the execution of CCID Escape commands. If an application uses those commands, apply the procedure in Appendix *A - Enabling Escape CCID commands*.

If the reader works in legacy mode (5325 mode) it requires installation of the HID driver. For details see *Appendix B - HID Driver* .

3 PC/SC

With the OMNIKEY 5025 CL, access contactless cards through the same framework as ISO7816 contact cards. This makes card integration a snap for any developer who is already familiar with PC/SC. Even valuable PC/SC resource manager functions, such as card tracking, are available for contactless card integration.

The Microsoft® Developer Network (MSDN®) Library contains valuable information and complete documentation of the SCard API within the MSDN Platform SDK.

See http://msdn.microsoft.com/en-us/library/windows/desktop/aa380149(v=vs.85).aspx

You can directly access contactless cards through the PC/SC driver.

3.1 Accessing Contactless Cards or Reader through PC/SC

The following steps provide a guideline to create your first contactless smart card application using industry standard, PC/SC compliant API function calls. The function definitions provided are taken verbatim from the MSDN Library [MSDNLIB]. For additional descriptions of these and other PC/SC functions provided by the Microsoft Windows PC/SC smart card components, reference the MSDN Library.
See http://msdn.microsoft.com/en-us/library/ms953432.aspx.

1. Establish Context
   This step initializes the PC/SC API and allocates all resources necessary for a smart card session. The **SCardEstablishContext** function establishes the resource manager context (scope) within which database operations is performed.

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

2. Get Status Change
   Check the status of the reader for card insertion, removal or availability of the reader.
   This **SCardGetStatusChange** function blocks execution until the current availability of the cards in a specific set of readers **change**. The caller supplies a list of monitored readers and the maximum wait time (in milliseconds) for an action to occur on one of the listed readers.

   ```
   LONG SCardGetStatusChange( IN SCARDCONTEXT hContext,
   IN DWORD dwTimeout,
   IN OUT LPSCARD_READERSTATE rgReaderStates,
   IN DWORD cReaders);
   ```

3. List Readers
   To acquire a list of all PC/SC readers use the `SCardListReaders` function. Look for **HID OMNIKEY 5025-CL** in the returned list. If multiple OMNIKEY Contactless Smart Card readers are connected to your system, they will be enumerated.

   Example: HID OMNIKEY 5025-CL 1, and OMNIKEY CardMan 5x21-CL 2.

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

4. Connect
   Connect to the card. The `SCardConnect` function establishes a connection (using a specific resource manager context) between the calling application and a smart card contained by a specific reader. If no card exists in the specified reader, an error is returned.

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

5. Exchange Data and Commands with the Card or the reader
   Exchange command and data through APDUs. The `SCardTransmit` function sends a service request to the smart card, expecting to receive data back from the card.

```
LONG SCardTransmit( IN SCARDHANDLE hCard,
                    IN LPCSCARD_I0_REQUEST pioSendPci,
                    IN LPCBYTE pbSendBuffer,
                    IN DWORD cbSendLength,
                    IN OUT LPSCARD_IO_REQUEST pioRecvPci,
                    OUT LPBYTE pbRecvBuffer,
                    IN OUT LPDWORD pcbRecvLength);
```

   Note: The application communicates through `SCardControl()` in environments

   - not allowing `SCardTransmit()` without an ICC
   - not allowing `SCardTransmit()` for any other reasons
   - when developers prefer the application communicate through `SCardControl()`.

   The application retrieves the control code corresponding to `FEATURE_CCID_ESC_COMMAND` (see part 10, rev.2.02.07). In case this feature is not returned, the application may try `SCARD_CTL_CODE` (3500) as control code to use.

```
LONG SCardControl( IN SCARDHANDLE hCard,
                   IN DWORD dwControlCode,
                   IN LPCVOID lpInBuffer,
                   IN DWORD nInBufferSize,
                   OUT LPVOID lpOutBuffer,
                   IN DWORD nOutBufferSize,
                   OUT LPDWORD lpBytesReturned);
```

6. Disconnect
   It is not necessary to disconnect the card after the completion of all transactions, but it is recommended. The `SCardDisconnect` function terminates a connection previously opened between the calling application and a smart card in the target reader.

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

7. Release
   This step ensures all system resources are released. The `SCardReleaseContext` function closes an established resource manager context, freeing any resources allocated under that context.

   ```
   LONG SCardReleaseContext( IN SCARDCONTEXT hContext);
   ```

3.2 Contactless PC/SC Commands

The PC/SC command set for contactless cards is defined in section 3.2 of the document *Interoperability Specification for ICCs and Personal Computer Systems - Part 3. Requirements for PC-Connected Interface Devices* and is available from the PC/SC Workgroup website http://www.pcscworkgroup.com. The commands use standard APDU syntax and standard SCardTransmit API, but use the reserved value of the CLA byte of ‘FF’.

Supported Reader Commands

Instruction	Description	Comments
CAh	Get Data	Partially supported (only UID)
70h	Vendor Specific	Fully support for all vendor specific generic commands

Common SW1SW2 return codes

SW1SW2	Meaning
9000h	Operation successful
6700h	Wrong length (Lc or Le)
6A81h	Function not supported
6B00h	Wrong parameter (P1 or P2)
C0XXh	Wrong length (wrong number Le; 'XX' encodes the exact number) if Le is less than the available UID length)
6F00h	Operation failed

3.2.1 Get Data (CAh)

This Get Data command will retrieve the UID of an inserted card. For HID PROX card, returned is the PAC number.

Command APDU

CLA	INS	P1	P2	Lc	Data In	Le
FFh	CAh	00h	00h	-	-	00h

Response APDU

Data Out	SW1SW2	
PAC number	9000h	Operation successful

For a usage example reference section 6.1 Get UID, page 17.

3.2.2 Vendor Specific Generic Command (70h)

This command allows applications to control OMNIKEY specific features provided by the reader.

Command APDU

CLA	INS	P1	P2	Lc	Data Field	Le
FFh	70h	07h	6Bh	xx	DER TLV coded PDU (Vendor Payload)	xx

In general the IFD supports the INS Byte 70h for vendor specific proprietary commands.

P1 and P2 constitute the vendor ID. For HID OMNIKEY products is the VID = 076Bh.

The Data Field is constructed as ASN.1 objects/items, whereby every OMNIKEY 5025 CL object is identified by a unique Object Identifier (OID).

OIDs are organized as a leaf tree under an invisible root node. The following table shows the first root nodes.

Vendor Command	Tag Value	Vendor Payload Branch
FF 70 07 6C Lc	A0h (constructed)	sioApi
	A2h (constructed)	readerInformationApi
	BCh (constructed)	deviceSpecificCommand
	9Dh (primitive) BDh (constructed)	response
	9Eh (primitive)	errorResponse

Subchapters present all OIDs.

3.2.2.1 Response APDU

For all commands encapsulated in generic 70h APDU, the IFD returns response frame constructed as follows.

Data field	SW1 SW2
DER TLV coded Response PDU	See ISO 7816-4

Two last bytes of response frame are always the return code, SW1SW2.

In cases of an ISO 7816 violation, the return code is according to ISO 7816-4 and the data field may be empty.

In cases of positive processing or internal errors, the IFD returns SW1SW2 = 9000 and the data field is encapsulated in the response TAG (9Dh or BDh) or error response TAG (9Eh).

The response includes more than one leaf, depending on the request. Each leaf is encapsulated in the leaf tag.

3.2.2.2 Error Response

The error response TAG caused by the firmware core is 9Eh (Class Context Specific) + (Primitive) + (1Eh). Length is 2 byte. First byte is the cycle in which the error occurred and the second byte is the exception type.

9E 02 xx yy 90 00	
Value	**Description**
9Eh	Tag = Error Response (0Eh) + (Class Context Specific) + (Primitive)
02h	Len = 2
cycle	Value byte 1: Cycle in which the error is occurred, see Error Cycle
error	Value byte 2: Error code, see Error Code
SW1	90
SW2	00

Error Cycle

First value byte	
Cycle	**Description**
0	HID Proprietary Command APDU
1	HID Proprietary Response APDU
2	HID Read or Write EEPROM Structure
3	RFU
4	RFU
5	RFU

Error Code

Second value byte		
Exception		**Description**
3	03h	NOT_SUPPORTED
4	04h	TLV_NOT_FOUND
5	05h	TLV_MALFORMED
6	06h	ISO_EXCEPTION
11	0Bh	PERSISTENT_TRANSACTION_ERROR
12	0Ch	PERSISTENT_WRITE_ERROR
13	0Dh	OUT_OF_PERSISTENT_MEMORY
15	0Fh	PERSISTENT_MEMORY_OBJECT_NOT_FOUND
17	11h	INVALID_STORE_OPERATION
19	13h	TLV_INVALID_SETLENGTH
20	14h	TLV_INSUFFICIENT_BUFFER
21	15h	DATA_OBJECT_READONLY
31	1F	APPLICATION_EXCEPTION (Destination Node ID mismatch)
42	2Ah	MEDIA_TRANSMIT_EXCEPTION (Destination Node ID mismatch)
43	2Bh	SAM_INSUFFICIENT_MSGHEADER (Secure Channel ID not allowed)

Second value byte		
Exception		**Description**
47	2Fh	TLV_INVALID_INDEX
48	30h	SECURITY_STATUS_NOT_SATISFIED
49	31h	TLV_INVALID_VALUE
50	32h	TLV_INVALID_TREE
64	40h	RANDOM_INVALID
65	41h	OBJECT_NOT_FOUND

4 Objects and Items

The reader presents smart card information as well as reader information as ASN.1 objects/items, whereby every object is identified by a unique Object Identifier (OID).

4.1 The OID Tree

OIDs are organized as a leaf tree under an invisible root node. The following table shows the first root nodes.

Object sub tree	Tag Value	Description
sioApi	A0h (constructed)	SIO API, allows processing of HID Secure Identity Objects
readerInformationApi	A2h (constructed)	Reader information API
deviceSpecificCommand	BCh (constructed)	Device specific command set
response	9Dh (primitive) BDh (constructed)	Response
errorResponse	9Eh (primitive)	Error Response

4.1.1 HID Secure Identity Object

SIO application interface, SIO Api (A0h), supports access to a HID Secure Identity Object.

Reader OK5025 CL does not have SIO processor and does not handle Secure Channel. The only supported SIO Api command is Get PAC Bits call.

Vendor Command	SIO API	Request	ASN1 name of Branches
FF 70 07 6B Lc	Tag = A0h	Get ContentElement [A1h]	ContentElementTag [80h] (read only) implicitFormatPhysicalAccessBits [04h]

For usage example refer to 6.2 Get PAC Bits, page 17.

5 Reader Configuration

All OMNIKEY 5025 CL configuration items are identified by a unique ASN.1 leaf. The root is defined as Reader Information API and is encapsulated in a vendor specific generic command described in 3.2.2 Vendor Specific Generic Command (70h), page 8.

The root tag **readerInformationApi** A2h is reserved for GET and SET of reader specific information and provides access to reader configuration.

For a Reader Information GET requests the Le byte must be present, and the Response Tag (1D) is always CONSTRUCTED.

Under the root and the get/set request are a number of branches, organized as follows:

Reader Information Structure

Vendor Command	Reader Information API	Request	ASN1 name of Branches
FF 70 07 6B Lc	Tag = A2h	Get [A0h] Set [A1h]	readerCapabilities [A0h] (read only) productName [82h] productPlatform [83h] firmwareVersion[85h] numberOfContactSlots[8b] numberOfContactlessSlots[8C] numberOfAntennas[8D] vendorName[8F] serialNumber[92h]
			readerCurrentMode [8Dh] (read/write)
			readerATRConfiguration [8Bh] (read/write)
			readerConfigurationControl [A9h] (write only) restoreFactoryDefaults [81h]
			proxConfiguration [ACh] (read/write) proxTimeScale [90h] proxParamPool [ABh] proxFSKPool [AEh]

Note: After SET requests, restart the reader to apply the changes.

5.1 Reader Capabilities (A0h)

Tag **readerCapabilities A0h** is constructed. One or more primitive sub tags must follow.

Ta	ASN.1 name	Value	Type	Len	Access
Name of product					
82h	productName	OMNIKEY 5025-CL	Null String	16	RO
Get APDU: FF 70 07 6B 08 **A2 06 A0 04 A0 02 82 00** 00 **Response:** BD 12 82 10 4F 4D 4E 49 4B 45 59 20 35 30 32 35 2D 43 4C 00 90 00					
Name of processor platform					
83h	productPlatform	AViatoR	Null String	8	RO
Get APDU: FF 70 07 6B 08 A2 06 A0 04 A0 02 83 00 00 **Response:** BD 0A 83 08 41 56 69 61 74 6F 52 00 90 00					
FwVersionMajor + FwVersionMinor + BuildNr					
85h	firmwareVersion	XX YY ZZ	Octet String	3	RO
Get APDU: FF 70 07 6B 08 A2 06 A0 04 A0 02 85 00 00 **Response:** BD 05 85 03 01 01 02 90 00					
Number of available contact slots					
8Bh	numberOfContactSlots	00h	Uint8_t	1	RO
Get APDU: FF 70 07 6B 08 A2 06 A0 04 A0 02 8B 00 00 **Response:** BD 03 8B 01 00 90 00					
Number of available contactless slots					
8Ch	numberOfContactlessSlots	01h	Uint8_t	1	RO
Get APDU: FF 70 07 6B 08 A2 06 A0 04 A0 02 8C 00 00 **Response:** BD 03 8C 01 01 90 00					
Number of available low frequency antennas					
8Dh	numberOfAntennas	01h	Uint8_t	1	RO
Get APDU: FF 70 07 6B 08 A2 06 A0 04 A0 02 8D 00 00 **Response:** BD 03 8D 01 01 90 00					
Vendor name					
8Fh	vendorName	HID Global	Null String	11	RO
Get APDU: FF 70 07 6B 08 A2 06 A0 04 A0 02 8F 00 00 **Response:** BD 0D 8F 0B 48 49 44 20 47 6C 6F 62 61 6C 00 90 00					
Unique IFD serial number					
85h	serialNumber	01h	Octet String	16	RO
Get APDU: FF 70 07 6B 08 A2 06 A0 04 A0 02 92 00 00 **Response:** BD 12 92 10 10 03 4F 00 4B 00 31 00 32 00 33 00 34 00 35 00 90 00					

5.2 HID Prox Configuration (ACh)

The tag **proxConfiguration ACh** is constructed. It must be followed by one primitive or constructed tag. Multi-leaf request is not allowed.

5.2.1 Prox Time Scale (9Ch)

The **proxTimeScale** tag **90h** is primitive and class specific. It defines the polling delay between card scanning.

Tag	ASN.1 name	Value	Type	Len	Access
90h	proxTimeScale	Default: 0014h Allowed: 0000h – 00FFh	Uint16_t	2	RW
Get APDU: FF 70 07 6B 08 **A2 06 A0 04 AC 02 90 00** 00 Response: BD 04 90 02 00 14 90 00 Set APDU: FF 70 07 6B 0A A2 08 A1 06 AC 04 90 02 00 14 00 Response: 9D 00 90 00					

The scaling rule is following:

$$delay = delay_{default} \times \frac{timeScale}{10}$$

As the default polling delay ($delay_{default}$) is 100 ms, actual polling delay is 200 ms.

5.3 ATR Configuration (8Bh) in Native CCID Mode

The tag **readerATRConfiguration 8Bh** is primitive, and defines ATR format in Native CCID Mode – see 5.4 Reader Mode (8Dh), page 15.

Tag	ASN.1 name	Value	Type	Len	Access
8Bh	readerATRConfiguration	00h - ATR is formatted according to PC/SC part 3 01h – ATR contains card data (PAC) in historical bytes	Octet String	1	RW
Get APDU: FF 70 07 6B 06 A2 04 A0 02 8B 00 00 **Response:** BD 03 8B 01 01 90 00 Set APDU: FF 70 07 6B 07 A2 05 A1 03 8B 01 00 00 **Response:** 9D 00 90 00					

ATR Format according to PC/SC part 3

The ATR of storage cards (for example HID Prox cards) is composed as described in PC/SC - **Part 3: Requirements for PC-Connected Interface Devices, section 3.1.3.2.3.2 Contactless Storage Cards, Table 3.6.** For the host application to identify a storage and card type properly, its standard and card name is mapped according to PC/SC 2.01 - Part 3: Requirements for PC - Connected Interface Devices - Supplemental Document.

Note: The Registered Application Provider Identifier (RID) returned by the OMNIKEY Contactless Smart Card reader for storage cards (cards without a CPU) is A0 00 03 06 0A, indicating a PC/SC compliant ATR generation.

Example:

```
ATR
3B 8F 80 01 80 4F 0C A0 00 00 03 06 40 00 00 00 00 00 00 28
    ^       ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^
    |       HISTORICAL BYTES                             ||
    |                                                    ||
    + F - number of historical bytes                     ++ TCK

HISTORICAL BYTES
80 4F 0C A0 00 00 03 06 40 00 00 00 00 00 00
^^ ^^ ^^ **  -RID-   ** ^^    ^^
|| || || ||             ||    ||
|| || || ||             ||    ++ NN - Bytes for Card Name (No information
given)
|| || || ||             ||
|| || || ||             ++ SS - Byte for Standard (Low frequency contactless
cards)
|| || || ||
|| || || ++ Registered application provider identifier (RID) for PC/SC
Workgroup
|| || ||
|| || ++ Length
|| ||
|| ++ Application identifier Presence indicator
||
++ Category indicator - status indicator may be present
```

ATR Format containing PAC Bits in Historical Bytes

In this setting the OMNIKEY 5025 CL reader returns PROX card data in an answer to reset (ATR) commonly used in PC/SC – based smart card systems. For HID PROX cards, the first byte of the ATR is always 3B hex. It is followed by a byte that indicates in its LSB nibble how many historical bytes will follow. The historical bytes hold the card's ProxFormat coded as follows:

```
00 03 SS NN DATA[SS – 1]
```

-	00	-	PROX format = Wiegand Raw
-	03	-	PAC BIT STRING TAG
-	SS	-	number of subsequent bytes
-	SS	-	number of trailing insignificant bits
-	DATA[SS]	-	PAC bits, where NN least significant bits in last byte are invalid

Example:

```
ATR
3B 88 01 00 03 05 06 80 86 98 C0 D7
    ^    ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^
    |    HISTORICAL BYTES        ||
    |                            ++ TCK
    |
    + number of historical bytes (LSB nibble)
HISTORICAL BYTES
00 03 05 06 80 86 98 C0
^^ ^^ ^^ ^^ ^^ ^^ ^^ ^^
|| || || || *CARD DATA* - PAC bits, where NN least significant bits in last
byte are invalid
|| || || ||
|| || || ++ NN - number of trailing insignificant bits
|| || ||
|| || ++ SS - number of subsequent bytes
|| ||
|| ++ PAC BIT STRING TAG
||
++ PROX format = Wiegand Raw
```

5.4 Reader Mode (8Dh)

The tag **readerCurrentMode 8Dh** is primitive and defines reader operating mode:

Tag	ASN.1 name	Value	Type	Len	Access
8Dh	readerCurrentMode	00h – Native CCID mode 01h – Legacy mode, OK5325 emulation	Octet String	1	RW
Get APDU: FF 70 07 6B 05 A2 03 A0 01 8D 00 **Response:** BD 03 8D 01 00 90 00 Set APDU: FF 70 07 6B 07 A2 05 A1 03 8D 01 01 00 Response: 9D 00 90 00					

Note: In Native CCID mode the OMNIKEY 5025 CL reader does not require a dedicated driver – native CCID driver from the operating system is used. The ATR format is configured according to the `readerATRConfiguration` setting. See 5.3 ATR Configuration (8Bh) in Native CCID Mode, page 13.

In Legacy mode the OMNIKEY 5325 behavior is emulated. HID driver is required. The ATR format is the same as for the OMNIKEY 5325 and is described in 6.3 ATR Format in Legacy Mode, page 18.

5.5 Reader Configuration Control (A9h)

The **`readerConfigurationControl`** tag A9h is constructed, and the SET branches control the reader's behavior.
There is only one supported function.

Tag	ASN.1 name	Function	Len	Access
81h	**`restoreFactoryDefaults`**	Restore Factory Defaults This means that any custom settings will be lost	1	command
Set APDU: **`FF 70 07 6B 09 A2 07 A1 05 A9 03 81 01 00 00`** Response: **`9D 00 90 00`**				

6 Migration Scenarios

6.1 Get UID

This Get Data command will retrieve the UID of an inserted card. For HID PROX card, PAC number will be returned.

Example: Reading HID PROX UID

Command:

```
FF CA          // Get Data
      00 00    // Get UID
            00 // Le
```

Response:

```
PACS        // PAC Bits
     90 00 // Success
```

6.2 Get PAC Bits

The OMNIKEY 5325 generates an ATR which contains the PACS bits. The OMNIKEY 5025 CL introduces a new method to retrieve those bytes in a card independent way.

The command Get PAC Bits returns the Physical Access Control bits of the inserted media. It is processed by the IFD firmware reading the HID PROX media. The GET PAC Bits command is coded as follows:

DER TLV PDU:

```
A0 05                 // CHOICE SioAPI
      A1 03           // CHOICE SamCommandGetContentElement
            80 01     // Sequence ContentElementTag
                  04 // Value = implicitFormatPhysicalAccessBits
```

The complete APDU for LF media is: `FF 70 07 6B 07 A0 05 A1 03 80 01 04 00`

Get PAC Bits response:

```
9D                          // Tag = CHOICE Response
   LL                       // number of subsequent bytes
      03                    //PAC BIT STRING TAG
         SS                 //number of subsequent bytes
            NN              //number of trailing insignificant bits
               DATA[SS - 1] //PAC bits, where NN least significant bits
                              in last byte are invalid
```

Example response for 35bit PAC bit string:

```
9D 08 03 06 05 81 ED BE 15 60
```

6.3 ATR Format in Legacy Mode

The OMNIKEY 5025 CL can be configured to emulate OK5325 behavior. In such a scenario the HID driver is required. The operating mode is controlled by the **`readerCurrentMode`** – see 5.4 Reader Mode (8Dh), page 15 for details.

The OMNIKEY 5025 CL reader in legacy mode returns PROX card data in an answer to reset (ATR) commonly used in PC/SC - based smart card systems. For HID PROX cards, the first byte of the ATR is always 3B hex. It is followed by a byte that indicates in its LSB nibble how many PROX data bytes will follow. The third byte holds the card's ProxFormat.

6.3.1 ATR Example

The following ATR was generated by a card that returned the Wiegand raw data 00 02 25 64 hex =

100010010101100100 bin.

```
ATR
3B 05 00 00 02 25 64
    ^ ^^ ^^ ^^ ^^ ^^
    | HISTORICAL BYTES
    |
    5 number of historical bytes (LSB nibble)
HISTORICAL BYTES
00 00 02 25 64
^^ ^^ ^^ ^^ ^^
|| *card data*
||
Prox Format (here 0, meaning Wiegand raw)
```

For more information regarding Prox Format Settings in Legacy mode see to the *OMNIKEY Contactless Smart Card Readers Developer Guide, 5321-903*.

7 Appendix A - Enabling Escape CCID commands

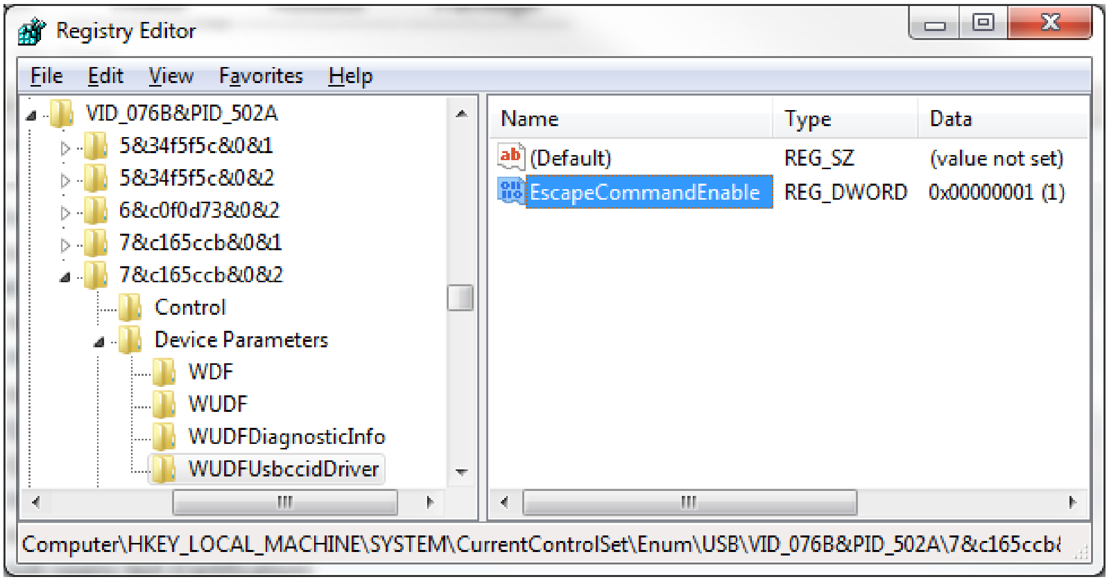
In order to send or receive an Escape command to a reader using Microsoft's CCID driver, add the DWORD registry value **EscapeCommandEnable** and set to a non-zero value under one of the following keys:

- **Windows 7 and 8**
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\VID_076B&PID_502A \<serial number>\Device Parameters\WUDFUsbccidDriver
- **Prior Windows 7**
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\VID_076B&PID_502A \<serial number>\ Device Parameters

Then the vendor IOCTL for the Escape command is defined as follows:

```
#define IOCTL_CCID_ESCAPE SCARD_CTL_CODE(3500).
```

For details see http://msdn.microsoft.com/en-us/windows/hardware/gg487509.aspx .



Name	Type	Data
(Default)	REG_SZ	(value not set)
EscapeCommandEnable	REG_DWORD	0x00000001 (1)

Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\VID_076B&PID_502A\7&c165ccb(

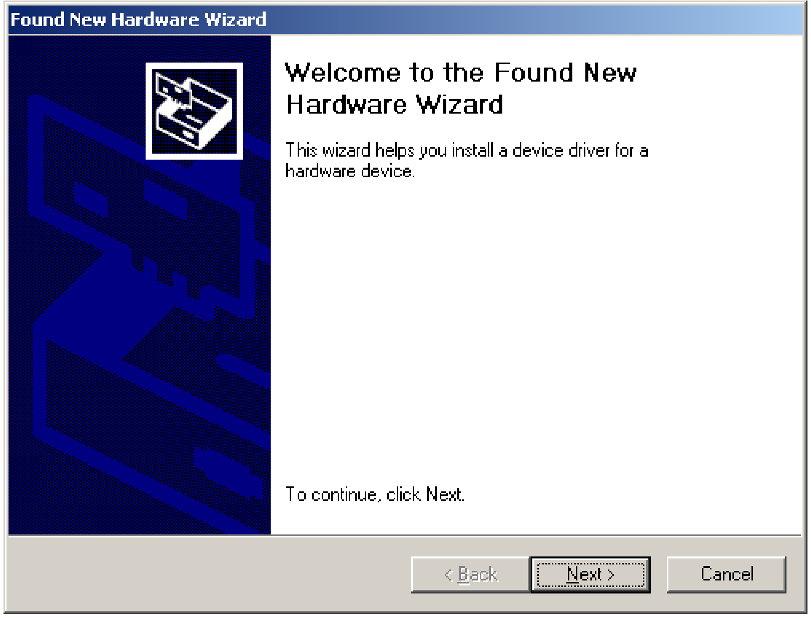
If reader with different serial number is connected to computer the operation must be repeated.

8 Appendix B - HID Driver Installation

This procedure is only valid for OMNIKEY 5025 CL reader running in legacy mode (when OMNIKEY 5325 reader is emulated).

1. Go to http://www.hidglobal.com/products/readers/omnikey. Click the Download OMNIKEY drivers link and select **5325 USB Prox** in the product combo box. Download the latest OMNIKEY Contactless Smart Card driver installation package for Windows.
2. Run the installation package and follow the instructions. The installation package extracts all the necessary driver files to your hard drive.
   Take note of the location to which the files were copied.

   At this time you have only extracted, not installed the driver files.
3. Connect the reader to your computers USB port.
4. The **Found New Hardware Wizard** appears. To continue the driver installation, click **Next**.



Found New Hardware Wizard

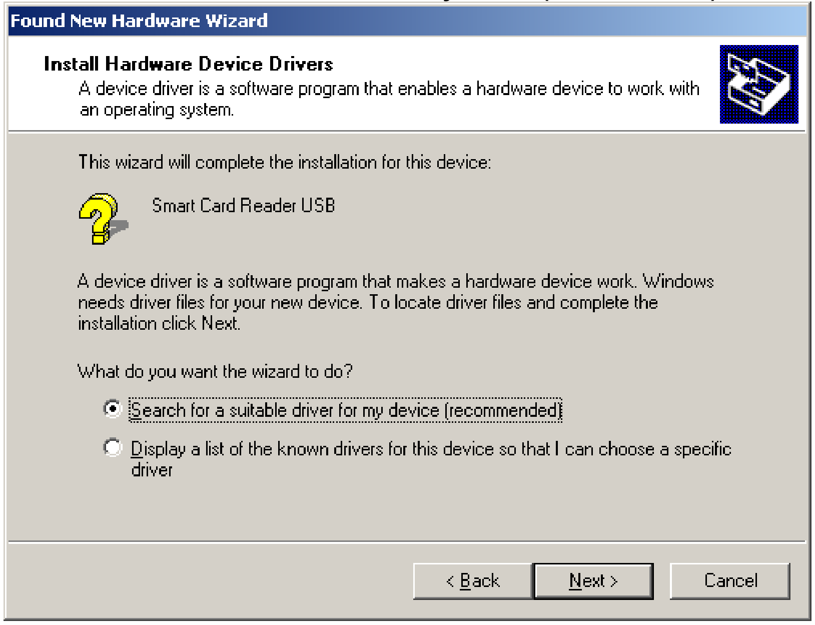
Welcome to the Found New Hardware Wizard

This wizard helps you install a device driver for a hardware device.

To continue, click Next.

Note: On Windows XP systems, the Microsoft Windows CCID Class driver may be activated without showing the **Found New Hardware Wizard**. If this is the case, replace the Microsoft PC/SC driver manually with the OMNIKEY proprietary PC/SC driver using the Device Manager. See Section 9 Appendix C - Manual Driver , page 24.

5. Select **Search for a suitable driver for my device (recommended)** and click **Next**.



Found New Hardware Wizard

Install Hardware Device Drivers

A device driver is a software program that enables a hardware device to work with an operating system.

This wizard will complete the installation for this device:

Smart Card Reader USB

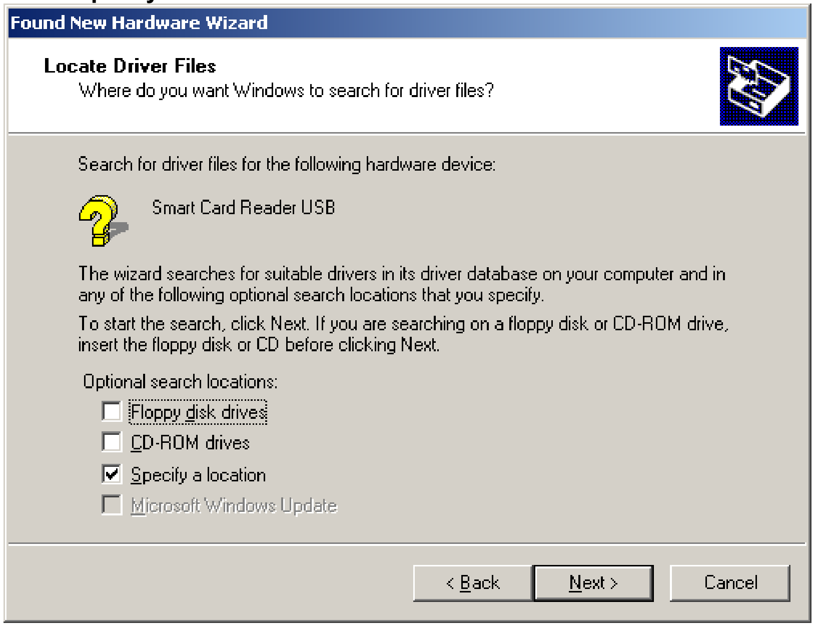
A device driver is a software program that makes a hardware device work. Windows needs driver files for your new device. To locate driver files and complete the installation click Next.

What do you want the wizard to do?

● Search for a suitable driver for my device (recommended)

○ Display a list of the known drivers for this device so that I can choose a specific driver

6. Select **Specify a location** and click **Next**.



Found New Hardware Wizard

Locate Driver Files

Where do you want Windows to search for driver files?

Search for driver files for the following hardware device:

Smart Card Reader USB

The wizard searches for suitable drivers in its driver database on your computer and in any of the following optional search locations that you specify.

To start the search, click Next. If you are searching on a floppy disk or CD-ROM drive, insert the floppy disk or CD before clicking Next.

Optional search locations:

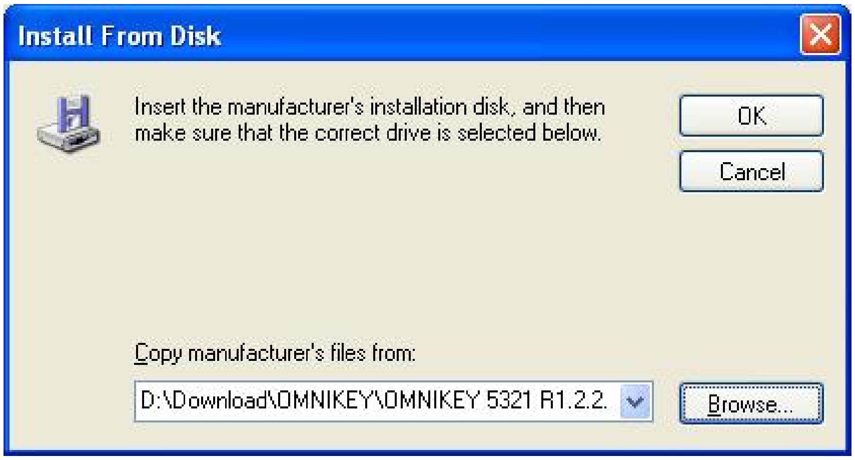
☐ Floppy disk drives

☐ CD-ROM drives

☑ Specify a location

☐ Microsoft Windows Update

7. Click **Browse** and go to the location where you saved the driver package. To continue, click **OK**.



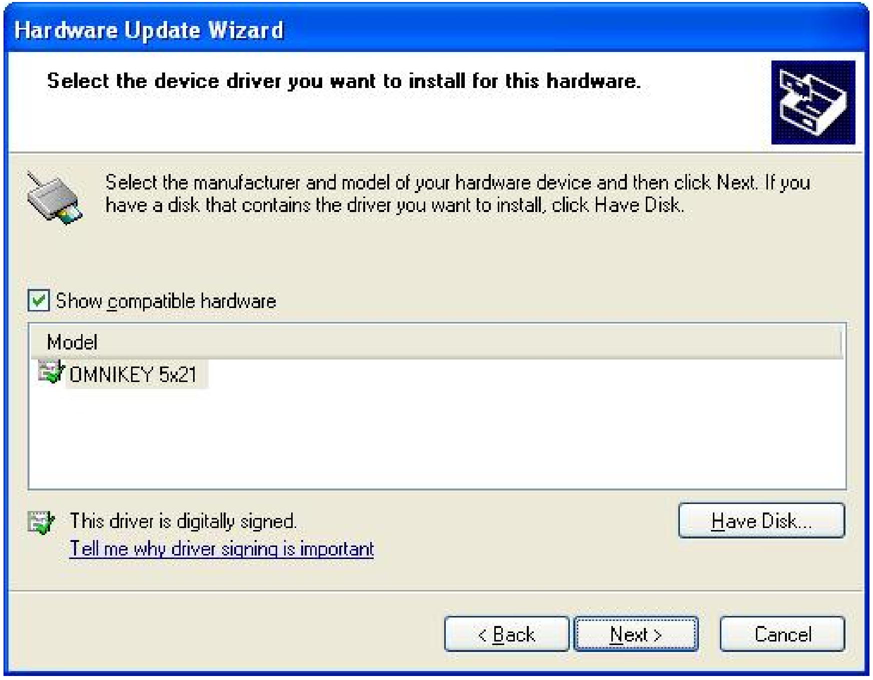
Install From Disk

Insert the manufacturer's installation disk, and then make sure that the correct drive is selected below.

Copy manufacturer's files from:

D:\Download\OMNIKEY\OMNIKEY 5321 R1.2.2.

8. If the driver was found, click **Next**.



Hardware Update Wizard

Select the device driver you want to install for this hardware.

Select the manufacturer and model of your hardware device and then click Next. If you have a disk that contains the driver you want to install, click Have Disk.

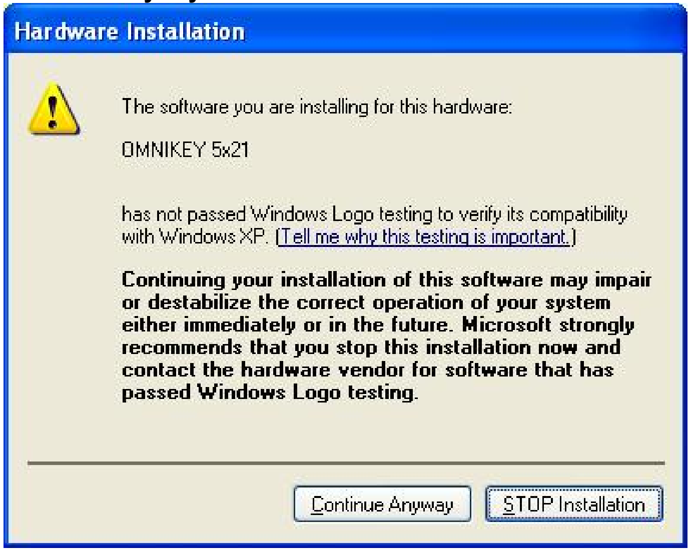
☑ Show compatible hardware

Model
OMNIKEY 5x21

This driver is digitally signed.

Tell me why driver signing is important

9. If the driver is a beta driver and not digitally signed, the following dialogue appears. Click **Continue Anyway**.



Hardware Installation

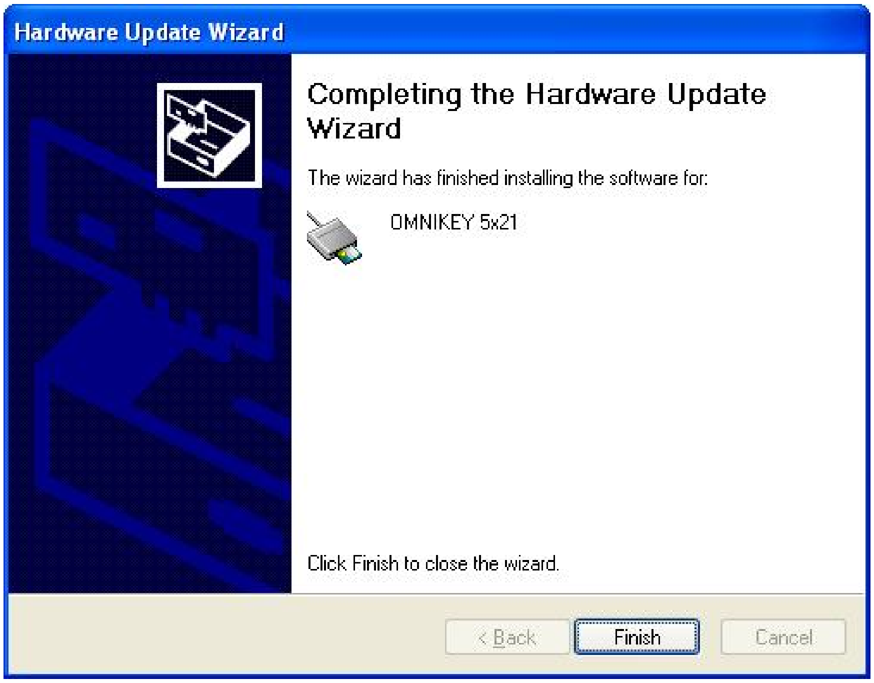
The software you are installing for this hardware:

OMNIKEY 5x21

has not passed Windows Logo testing to verify its compatibility with Windows XP. (Tell me why this testing is important.)

Continuing your installation of this software may impair or destabilize the correct operation of your system either immediately or in the future. Microsoft strongly recommends that you stop this installation now and contact the hardware vendor for software that has passed Windows Logo testing.

10. The following message appears and the green LED illuminates on the OMNIKEY Contactless Smart Card reader.



Hardware Update Wizard

Completing the Hardware Update Wizard

The wizard has finished installing the software for:

OMNIKEY 5x21

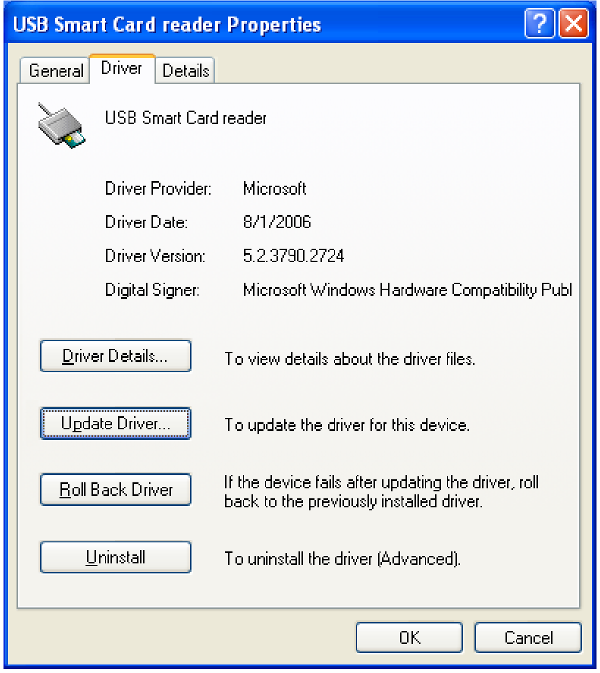
Click Finish to close the wizard.

If the installation was successful, the LED on the reader illuminates and the reader is listed in the OMNIKEY Workbench as OMNIKEY Contactless Smart Card reader.

9 Appendix C - Manual Driver Update

This procedure describes how to manually choose the driver used by the operating system.

1. In the **Device Manager** find the OMNIKEY 5025 CL reader and choose its **Properties**.
2. In the **Driver** tab, click **Update Driver…**.



USB Smart Card reader Properties

USB Smart Card reader

Driver Provider: Microsoft

Driver Date: 8/1/2006

Driver Version: 5.2.3790.2724

Digital Signer: Microsoft Windows Hardware Compatibility Publ

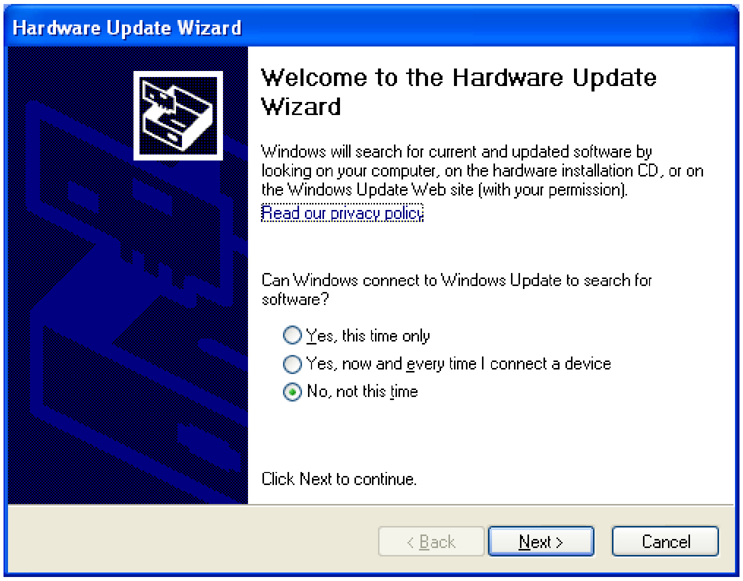
To view details about the driver files.

To update the driver for this device.

If the device fails after updating the driver, roll back to the previously installed driver.

To uninstall the driver (Advanced).

3. To install the driver already available in your system in **Hardware Update Wizard** choose **No, not this time** and click **Next**.



Hardware Update Wizard

Welcome to the Hardware Update Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

Read our privacy policy

Can Windows connect to Windows Update to search for software?

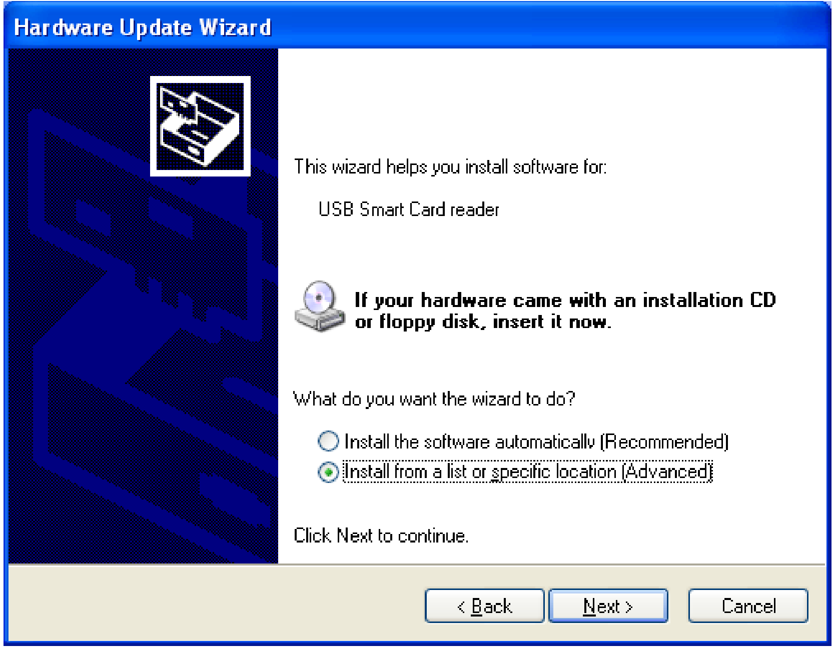
○ Yes, this time only

○ Yes, now and every time I connect a device

● No, not this time

Click Next to continue.

4. Select **Install from a list or specific location (Advanced)**. Click **Next**.



Hardware Update Wizard

This wizard helps you install software for:

USB Smart Card reader

If your hardware came with an installation CD or floppy disk, insert it now.

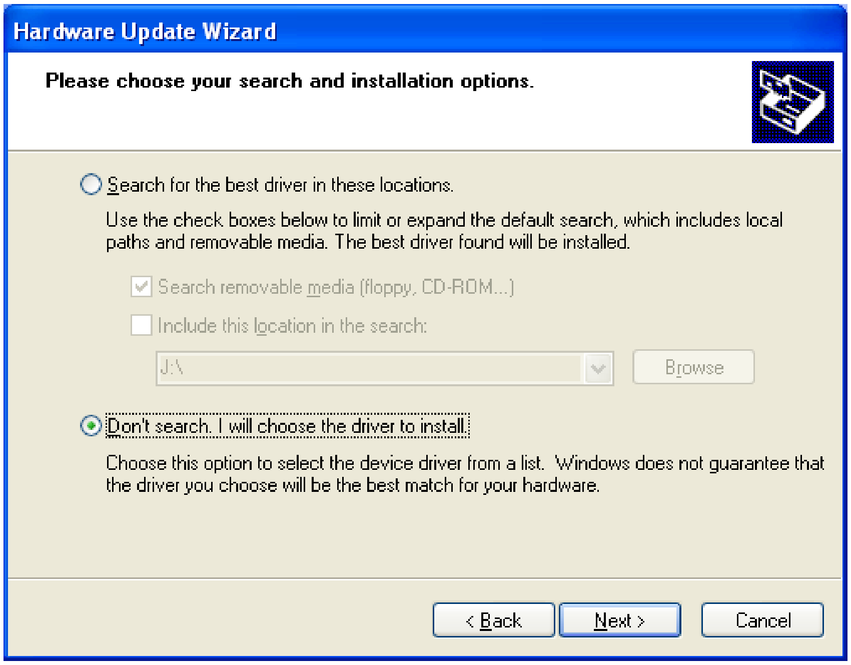
What do you want the wizard to do?

○ Install the software automatically (Recommended)

● Install from a list or specific location (Advanced)

Click Next to continue.

5. And then **Don't search. I will choose the driver to install**. Click **Next**.



Hardware Update Wizard

Please choose your search and installation options.

○ Search for the best driver in these locations.

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

☑ Search removable media (floppy, CD-ROM...)

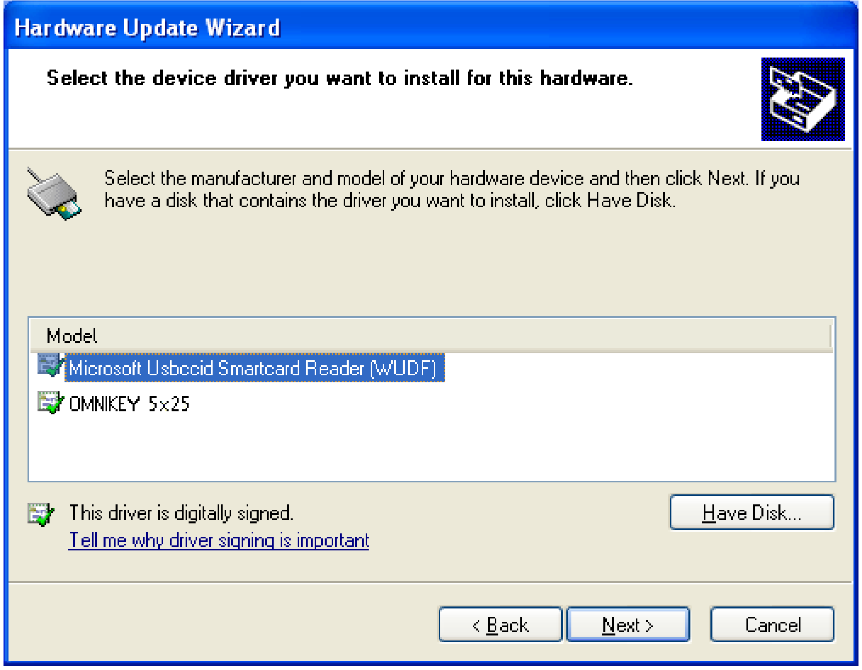
□ Include this location in the search:

J:\

● Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

6. Select the driver for your device and click **Next**.



Hardware Update Wizard

Select the device driver you want to install for this hardware.

Select the manufacturer and model of your hardware device and then click Next. If you have a disk that contains the driver you want to install, click Have Disk.

Model
Microsoft Usbccid Smartcard Reader (WUDF)
OMNIKEY 5x25

This driver is digitally signed.
Tell me why driver signing is important

10 Appendix D - Definitions, Abbrev and Symbols

AES	Advanced Encryption Standard
APDU	Application Protocol Data Unit
API	Application Programming Interface
ASN.1	Abstract Syntax Notation One
BER	Basic Encoding Rules
CLA	Class byte of an APDU
DER	Distinguished Encoding Rules
MAC	Message Authentication Code
MSDN	Microsoft® Developer Network
OID	Object Identifier
PAC	Physical Access Control
PACS	PAC Physical Access Control Services
PDU	Protocol Data Unit
PC/SC	Personal Computer/Smart Card
SIO	Secure Identity Object
CSN	Card Serial Number
IFD	Interface Device (for accessing ICC card)

11 Appendix E - References

[ISO 7816-4]	ISO 7816-4 Identification cards — Integrated circuit cards - Part 4: Organization, security and commands for Interchange Second edition - 2005-01-15
[ISO 8825]	ISO/IEC8825 ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) Fourth edition 2008-12-15 or X.690 Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)
[PCSC-3-Sup-CL]	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Supplemental Document for Contactless ICCs Revision 2.02.00
[PCSC-3]	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Requirements for PC-Connected Interface Devices Revision 2.01.09

[PCSC-3-Sup]	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Supplemental Document Revision 2.01.08
[PCSC-3-AMD]	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Requirements for PC-Connected Interface Devices - AMENDMENT 1 Revision 2.01.09
[OMNIKEY DEV GUIDE]	OMNIKEY® Contactless Smart Card Readers Developer Guide Paragraph 12. Driver Configuration via ProxFormat Paragraph 13. ProxFormat Settings Revision B.2

12 Appendix F - Sample Code

This sample displays PROX card ATR and data.

```
// Sample C code that displays ATR and data from PROX card.
// Link with winscard.lib
// Copyright 2013, HID Global

#include <stdio.h>
#include <winscard.h>

int main(int argc, char* argv[])
{
    LONG lResult;
    SCARDCONTEXT hContext;
    SCARDHANDLE hCard;
    DWORD dwActiveProtocol;
    DWORD dwLen;
    SCARD_IO_REQUEST pIoReq;
    BYTE pBuffer[32];
    USHORT SW;
    DWORD i;

    //APDU to get data from card
    BYTE GET_DATA[] = {0xFF, 0xCA, 0x00, 0x00, 0x00};

    // First 5025 reader name
    WCHAR szReader[] = L"HID OMNIKEY 5025-CL 0";

    // Establish context
    lResult = SCardEstablishContext(SCARD_SCOPE_USER, NULL, NULL, &hContext);
    if( SCARD_S_SUCCESS != lResult )
    {
        printf("SCardEstablishContext failed. Error code 0x%08X.\n", lResult );
        return 1;
    }

    //Connect to card
    lResult = SCardConnect( hContext,
                            szReader,

                            SCARD_SHARE_SHARED,
                            SCARD_PROTOCOL_T0 | SCARD_PROTOCOL_T1,
                            &hCard,
                            &dwActiveProtocol);

    if( SCARD_S_SUCCESS != lResult )
    {
        //release context
        SCardReleaseContext(hContext);

        printf("Can not detect card. Error code 0x%08X.\n", lResult );
        return 1;
    }

    //Select protocol T=1 or T=0
    if( SCARD_PROTOCOL_T1 == dwActiveProtocol )
    {
        pIoReq = *SCARD_PCI_T1;
    }
    else
    {
        pIoReq = *SCARD_PCI_T0;
    }

    //get ATR
    dwLen = sizeof(pBuffer);
    lResult = SCardStatus(  hCard,
                            NULL,
                            NULL,
                            NULL,
                            NULL,
                            pBuffer,
                            &dwLen);

    if( SCARD_S_SUCCESS != lResult )
    {
        //disconnect card
        SCardDisconnect(hCard, SCARD_LEAVE_CARD);

        //release context
        SCardReleaseContext(hContext);

        printf("Can not get card status. Error code 0x%08X.\n", lResult );
        return 1;
    }

    //display ATR
    printf("ATR: ");
    for(i=0;i<dwLen-2;i++)
    {
        printf(" %02X", pBuffer[i]); //print hex digits
    }
    printf("\n"); //end of line

    dwLen = sizeof(pBuffer);
    lResult = SCardTransmit(hCard,
                            &pIoReq,
                            GET_DATA,
                            sizeof(GET_DATA),
                            NULL,
                            pBuffer,
                            &dwLen);

    if( SCARD_S_SUCCESS != lResult )
    {
        //release context
        SCardReleaseContext(hContext);

        printf("Card not detected. Error code 0x%08X.\n", lResult );
        return 1;
    }

    SW = pBuffer[dwLen-2] << 8 | pBuffer[dwLen-1];

    //response code
    if( SW != 0x9000 )
    {
              //disconnect card
              SCardDisconnect(hCard, SCARD_LEAVE_CARD);

              //release context
           SCardReleaseContext(hContext);

        printf("Command not accepted. Error code 0x%04X.\n", SW );
        return 1;
    }

    //display response
    printf("Data:");
    for(i=0;i<dwLen-2;i++)
    {
        printf(" %02X", pBuffer[i]); //print hex digits
    }
    printf("\n"); //end of line

    //disconnect card
    SCardDisconnect(hCard, SCARD_LEAVE_CARD);

    //release context
    SCardReleaseContext(hContext);
    return 0;
}
```

HID Global Headquarters:

North America: +1 949 732 2000

Toll Free: 1 800 237 7769
Europe, Middle East, Africa: +49 6123 791 0

Asia Pacific: +852 3160 9800

Latin America: +52 477 779 1492

support.hidglobal.com

hidglobal.com