

SMARTCARD TERMINAL

Technical Field

This invention concerns a smartcard terminal, that is a device that is able to read information from a smartcard and write information to a smartcard. A smartcard terminal is typically provided as a peripheral for a computer, such as a personal computer, a workstation, a network server or a network computer.

Background Art

For many years communication with a smartcard has involved the use of standalone smartcard terminal devices. The smartcard terminal devices have been used to write information to smartcards and to read information from smartcards. The terminal devices have been connected to computer terminals, and more recently to personal computers, by the standard serial communications port using the standard RS232 interface. The devices have been powered separately by means of a discrete power supply.

Recently smartcard terminals have been integrated into computer keyboards. In one variant the existing legacy keyboard interface remains, and the bidirectional smartcard information is communicated between the keyboard and the computer by an additional standard serial communications port using the standard RS232 interface. There are generally only two serial ports on a personal computer. The first serial port is usually dedicated to a mouse or trackball, and the second to a modem. Consequently in order to connect such a smartcard terminal a third serial port must be added.

In an alternative, the bidirectional smartcard information is communicated using new commands which have been added to the existing half duplex legacy keyboard interface. However, in this variant while bidirectional smartcard transactions are in progress, keyboard strokes may be ignored.

Summary of the Invention

The invention is a smartcard terminal including a smartcard reader and a data assembly and transmission means to receive position data from a computer display pointing device having a serial communications capability, and from the smartcard reader. The data assembly and transmission means is able to transmit position information from the pointing device, and smartcard information from the smartcard reader to an associated computer, and to receive information which is to be written to the smartcard from the computer. The data assembly and transmission means assembles position information and smartcard information into separate frames before transmitting them serially, and it labels each frame depending on whether it contains position information or smartcard information.

In one embodiment the smartcard reader and the data assembly and transmission means are housed together in a housing which includes a serial data port to receive the position data.

The smartcard terminal may include a housing that houses both the smartcard reader and the computer display pointing device. In this case the only port required on the terminal is the port required for communication with the computer.

In this way the invention may provide a solution to the many users who may wish to use a smartcard in conjunction with a computer without requiring an additional serial port, and without compromising the functionality of the keyboard.

Such use may include offline and online access control, offline and online smartcard initialisation, online authentication, online certification, online end to end public and private key encryption, online bidirectional non-financial transactions, online bidirectional financial transactions, including purchase transactions, and withdrawals which reload an electronic purse on the smartcard for later offline payments.

The terminal may be capable of communicating with many smartcards at the same time by being equipped with several smartcard readers which are accessed through respective slots in the housing.

The smartcard terminal may include a keypad for keypad information entry, in which case the data assembly and transmission means may assemble the position information, the smartcard information and any keypad information into separate frames before transmitting them serially

The terminal may incorporate a mouse, but it could also incorporate a trackball, joystick, any other pointing device, a secure PINpad capability, an insecure keypad capability or a modem. The terminal may be in a notebook computer specifically integrated with an electronic stylus pointing device.

The terminal may be used with a personal computer, but it could also be used with a workstation, a network server, a network computer or a computer terminal.

The terminal may employ a cable to communicate with the computer, and the cable could also supply power to the smartcard reader. The cable could also be replaced by an optical transport medium or a radio frequency transport medium.

The communications means may assemble the information into frames which start with a start of text character (STX) to identify the start of the frame. The next byte following the start of frame character may be a Protocol Control Byte (PCB). If the PCB is any other than an X-Y position code, for instance code 0x7F and or below, then the next bytes following are the data length characters (LEN). The data length characters specify the length of the subsequent information bytes. After receiving the LEN characters the personal computer waits to receive the subsequent bytes until the number of bytes received equals to the number of information bytes specified in the LEN byte plus 2 bytes (ETX and BCC). The format is shown in the following Table:

STX	PCB	LEN	Information Block	ETX	BCC
-----	-----	-----	-------------------	-----	-----

The following definitions apply:

STX	ASCII 02H
PCB	The protocol control byte of the message
LEN	The length of the information block in bytes
Information Block	Command data to the integrated smartcard terminal device
ETX	ASCII 03H
BCC	The block check of the information bytes

- 5 The Protocol Control Byte in the frames between the terminal and the computer contains a one byte value chosen from those available shown in the following table:

PCB	Type	Description
0000000Sb	D-Frame,	Data frame
0001000Sb	L-Frame,	Link control frame
0010000Sb	N-Frame,	NACK frame
0011000Sb	E-Frame,	Event frame
01xxxxxxb	M-Frame	X-Y Position Information

- 10 The information block in the frame may contain the terminal current status, data from the smartcard, secure keypad, keypad or X-Y position information.

Brief Description of the Drawings

- 15 An example of the invention will now be described with reference to the accompanying drawings, in which:

Figure 1 is a pictorial view of a smartcard terminal.

Figure 2 is a pictorial view of the underside of the terminal of Figure 1. And

Figure 3 is a schematic diagram of an alternative terminal.

The same reference numerals have been used throughout the
5 drawings to refer to corresponding features.

Best Modes for Carrying out the Invention

The integrated smartcard terminal device 1 has an external plastics casing 2 shaped like a mouse with two buttons 3 and 4 on the top and a roller
10 ball 5 underneath. An horizontal slot 6 along the front can receive a smartcard.

The roller ball 5 engages two roller bearings with optical shaft encoders and an opposing spring loaded roller bearing. The shaft encoders are orthogonal to each other perpendicular in the same plane to record movement
15 of the roller ball in conventional mouse fashion.

Within the device 1 there are also conventional smartcard reading and writing heads 7 together with associated circuitry. The information being read from the smartcard is transmitted to a personal computer together with X-Y position input control information along a cable 8. Information to be written
20 to the smartcard is also transmitted back along the same cable 8.

A bicolour, red and green, LED 9 is provided for indicating the logical status of the smartcard interface of the terminal device 1.

A keypad 10 is positioned in the underside of the casing 2 to enable key entry. A thumbscanner 11 is also incorporated into the side of the housing
25 to verify the identity of the user.

A bidirectional communications transport protocol is duplicated at either end of the cable, both in the terminal device 1 and at the device driver level of the personal computer (not shown).

The core of the bidirectional communications transport protocol
30 comprises of a set of control commands. By use of the control commands the

terminal device is capable of reading from a smartcard, writing to a smartcard and providing X-Y position input control information to the personal computer.

The communications protocol is defined in two sections. The first
5 section consists of frame level processing between the personal computer (PC) and the terminal device 1 (TERMINAL). The second section comprises application messages transmitted between the TERMINAL and the PC.

Message frames can originate at either the PC or the TERMINAL.
Message frames originating from the PC are either D-frames (data frames), or L-
10 frames (link control frames). Message frames originating from the TERMINAL are either D-frames (data frames), N-frames (negative acknowledgment frames), E-frames (event frames) or M-frames (mouse event frames). All message frames originating from the PC are acknowledged by the TERMINAL with a corresponding message frame.

15 All frames start with a start of text character (STX) to identify the start of the frame. Any other characters received by the terminal device 1 are ignored and are flushed from the receive buffer. The next byte following the start of text character is the Protocol Control Byte (PCB). The following bytes are the data length characters (LEN). The data length characters specify the
20 length of the subsequent information bytes. After receiving the LEN bytes the terminal device waits to receive the subsequent bytes until the number of bytes received is equal to the number of information bytes specified in the LEN byte plus 2 bytes (ETX and BCC). All message frames originating at either device (except for M-frames) contain a block checksum. A message sequence is
25 included in D-frames. The format is shown in the following Table 1:

STX	PCB	LEN	Information Block	ETX	BCC
-----	-----	-----	-------------------	-----	-----

The following definitions apply:

STX	ASCII 02H
-----	-----------

PCB	The protocol control byte of the message
LEN	The length of the information block in bytes
Information Block	Command data to the integrated smartcard terminal device
ETX	ASCII 03H
BCC	The block check of the information bytes

Table 1.

The Protocol Control Byte in the frames between the terminal and the computer contains a one byte value chosen from those available shown in

5 Table 2

PCB	Type	Description
0000000Sb	D-Frame,	Data frame
0001000Sb	L-Frame,	Link control frame
0010000Sb	N-Frame,	NACK frame
0011000Sb	E-Frame,	Event frame
01xxxxxxb	M-Frame	X-Y Position Information

Table 2

D-frames

D-frames are data frames and are sent in both directions between the PC and the TERMINAL. D-frames sent by the PC to the TERMINAL contain commands. The command code and any relevant data is contained within the I-block. A D-frame sent by the TERMINAL to the PC is always in direct response to a PC to TERMINAL D-frame. The TERMINAL never sends unsolicited D-frames. The I-block of D-frames sent by the TERMINAL contains the response code and any relevant data.

The s bit is the message sequence number as shown in the following table, and the value of this bit sent for a particular D-frame is equal to the

current message sequence number as maintained internally by the sender's data link software. This internal message sequence number is set to 0 on protocol initialisation. This sequence number is incremented module 2 on receipt of a D-frame with a sequence number equal to the current internal
5 sequence number. For every D-frame received, the sequence number bit in the PCB byte is compared against the receiver's internally maintained sequence number. If the sequence numbers are the same, the D-frame is considered valid and processed by the receiver. Also, the receiver's internally maintained sequence number is incremented module 2.

PCB (binary)	Description
0000 000Sb	D-frame, data frame

10

L-frames

L-frames are link control frames used by the data link layer for managing the communications link between the PC and TERMINAL. At present there is a single L-frame command defined:

PCB (binary)	Description
0001 0000b	Data link version

15

N-frames

N-frames are negative acknowledgment frames or NAK frames, signalling a negative acknowledgment of a received frame. The TERMINAL
20 sends this frame after receiving an invalid frame (incomplete or bad checksum) from the PC. The LEN for such frames is 0.

E-frames

E-frames are frames containing event messages from the TERMINAL.
25 The I-block consists of a type byte and any relevant data.

M-frames

- M-frames are event frames and are sent by the TERMINAL when it detects a change in the state of its buttons or when sufficient X-Y movement has occurred. M-frames are the only frames not to conform to the frame structure above. Each M-frame is a fixed 4 bytes in length, and has the following format:
- 5

STX	PCB	X Position	Y Position
-----	-----	------------	------------

D-Frame Command Full Functional Definitions

GetDeviceCharacteristics

This command queries the TERMINAL for device. This enables the PC to query for such parameters as the TERMINAL serial number and default ICC

5 clock rate.

Command Format:

I-block
Command
Tag

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	GET_DEVICE_CHARACTERISTICS, 00h
Tag	DWORD	Identifies the data element being queried. The TERMINAL must respond for all defined tag values, with the exception of 0110h. The TERMINAL will not be queried on this data element

Response Format:

I-block
ResultCode Data

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_ERROR_TAG	Invalid tag value

ITEM	TYPE	DESCRIPTION
Data	Variable length	Only present if ResultCode is RESULT_SUCCESS. Note that this is a variable length data item, and the I-block length is used to determine this item's length

GetICCState

This command queries the **TERMINAL** for the ICC. This enables the PC to query for such parameters as card presence and the last ATR string.

Command Format:

I-block
Command Tag

5 Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	GET_ICC_STATE, 01h
Tag	DWORD	Identifies the data element being queried.

Response Format:

I-block
ResultCode Data

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_ERROR_TAG	Invalid tag value

ITEM	TYPE	DESCRIPTION
Data	Variable length	Only present if ResultCode is RESULT_SUCCESS Note that this is a variable length data item, and the I-block length is used to determine this item's length

GetProtocolState

This command queries the **TERMINAL** for current protocol. This enables the PC to query for such parameters as the current IFSC and IFSD.

Command Format:

I-block
Command Tag

5 Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	GET_PROTOCOL_STATE, 02h
Tag	DWORD	Identifies the data element being queried.

Response Format:

I-block
ResultCode Data

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_ERROR_TAG	Invalid tag value

ITEM	TYPE	DESCRIPTION
Data	Variable length	Only present if ResultCode is RESULT_SUCCESS. Note that this is a variable length data item, and the I-block length is used to determine this item's length

SetProtocolState

This command is used to set protocol parameters used in the TERMINAL's interaction with the smartcard. This enables the PC to change such parameters as the current IFSD.

5 Command Format:

I-block
Command Tag Data

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	SET_PROTOCOL_STATE, 03h
Tag	DWORD	Identifies the data element being set.
Data	Variable length	Value of parameter. Note that this is a variable length data item, and the I-block length is used to determine this item's length

Response Format:

I-block
ResultCode

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_ERROR_TAG	Invalid tag value
RESULT_ERROR_SET_FAILURE	Operation failed
RESULT_ERROR_READ_ONLY	The parameter can not be set

SetTerminalMode

This command may be used to set the mode of the integrated smartcard terminal device. The integrated smartcard terminal device may operate in either standard mode, or full-integrated smartcard terminal device mode. This

- 5 command may also set a new communications baud rate.

Command Format:

I-block
Command Mode Baudrate

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	SET_TERMINAL_MODE, 04h
Mode	BYTE	01h = Standard mode 02h = Full integrated smartcard terminal device mode
Baudrate	BYTE	01h=1200bps 02h=4800bps 03h=9600bps 04h=19,200bps 05h=33,600bps

Response Format:

I-block
ResultCode

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_INVALID_VALUE	Invalid mode or baudrate

DefineCardType

This command may be used to set the integrated smartcard terminal device to operate on a specific smartcard type. By default, an asynchronous card type is selected.

5 Command Format:

I-block
Command CardType

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	DEFINE_CARD_TYPE, 05h
CardType	BYTE	Current card type. The following card families are supported: 02h = asynchronous card (T=0 and T=1) 03h = 256 family 04h = 416 family 81h = 101 family 90h = ISO 2-Wire Protocol A0h = I2C Protocol

Response Format:

I-block
ResultCode

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_INVALID_VALUE	Invalid CardType

EnableSmartcardPort

All commands received by the TERMINAL are either port specific or non-port specific.

- 5 This command enables one of the smartcard ports on the TERMINAL. For all commands received after this command that are port specific, the TERMINAL will interact with the smartcard in the port specified by this command. On reset, smartcard port 0 is enabled.

Command Format:

I-block
Command Port

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	ENABLE_SMARTCARD_PORT, 06h
Port	BYTE	00h = Port 0, Full size card 01h = Port 1, First ID000 size card 02h = Port 2, Second ID000 size card 03h = Port 3, Third ID000 size card 04h = Port 4, Fourth ID000 size card

10 **Response Format:**

I-block
ResultCode

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_INVALID_VALUE	Invalid Port

PowerOnICC

This command powers on and resets the card in the currently selected smartcard port. The reset function is described in ResetICC.

Command Format:

I-block
Command

5 Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	POWER_ON_ICC, 07h

Response Format:

I-block
ResultCode ICCType CurrentProtocolType ATRString

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful

ITEM	TYPE	DESCRIPTION
ICCType	BYTE	Same coding as for GetICCState parameter ICCType (tag 0304h)
CurrentProtocolType	BYTE	Same coding as for GetProtocolState parameter CurrentProtocolType (tag 0201h)
ATRString	Variable length	Same coding as for GetICCState parameter ATRString (tag 0303h). Note that this is a variable length data item (max 32 bytes), and is therefore the last data item in the I-block. The I-block length is used to determine this item's length

ResetICC

This command resets the card in the currently selected smartcard port.

Command Format:

I-block
Command

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	RESET_ICC, 08h

5 Response Format:

I-block
ResultCode [ICCType CurrentProtocolType ATRString]

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful
RESULT_NO_ICC_POWER	The ICC has not yet been powered on

The following data items are only present if ResultCode is RESULT_SUCCESS:

ITEM	TYPE	DESCRIPTION
ICCType	BYTE	Same coding as for GetICCState parameter ICCType (tag 0304h). For values 01h (async) or 02h (sync), an implicit DefineCardType will have been performed by ResetICC
CurrentProtocolType e	BYTE	Same coding as for GetProtocolState parameter CurrentProtocolType (tag 0201h)

ATRString	Variable length	Same coding as for GetICCState parameter ATRString (tag 0303h). Note that this is a variable length data item (max 32 bytes), and is therefore the last data item in the I-block. The I-block length is used to determine this item's length
-----------	--------------------	---

PowerOffICC

This command will power off the card in the currently selected smartcard port. It is good practice to always power off the card before removing it from the TERMINAL device.

5 Command Format:

I-block
Command

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	POWER_OFF_ICC, 09h

Response Format:

I-block
ResultCode

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Operation successful

ISOInput

- On receipt of this command, the **TERMINAL** sends a command to the inserted smartcard together with any optional data. The **TERMINAL** does not expect any data to be returned from the card except for the two **ME** result bytes for
- 5 async cards. This command can be used for synchronous cards and **T=0** asynchronous cards. Use the **ISOT1** command for **T=1** asynchronous cards.

Command Format:

I-block
Command Class Instruction P1 P2 Length Data

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	ISO_INPUT, 0Ah
Class	BYTE	ISO command class byte
Instruction	BYTE	ISO command instruction byte
P1	BYTE	ISO command P1 byte
P2	BYTE	ISO command P2 byte
Length	BYTE	ISO command length byte. May be 0 for ISO commands not requiring any data. In that case the Data item will be null
Data	Variable length	Data to be sent to card. Number of bytes in this field is specified in Length parameter.

Response Format:

I-block
ResultCode [ME1 ME2]

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Successful communication with card. Success of card operation can be determined by examining ME1, ME2
RESULT_NO_ICC_POWER	The ICC has not yet been powered on
RESULT_BYTE_ERROR	Error reading byte, eg parity error
RESULT_BAD_PB	Bad procedure byte returned from card
RESULT_CARD_REMOVED	ICC has been removed whilst power was applied
RESULT_NO_CARD	No card is inserted into the TERMINAL
RESULT_NO_RESPONSE	No response from the card
RESULT_WRONG_PROTOCOL	Card uses T=1 protocol, cant use ISOInput
RESULT_CARD_TYPE_NOT_SE T	On last card reset, no ATR was read, and no subsequent DefineCardType command was performed

The following data items are only present if ResultCode is RESULT_SUCCESS:

ITEM	TYPE	DESCRIPTION
ME1	BYTE	Returned by ICC as result of command.
ME2	BYTE	Returned by ICC as result of command.

ISOOutput

On receipt of this command, the **TERMINAL** sends a command to the inserted smartcard and reads back the resulting data from the card. This command can be used for synchronous cards and T=0 asynchronous cards. Use the **ISOT1**

- 5 command for T=1 asynchronous cards.

Command Format:

I-block
Command Class Instruction P1 P2 Length

Command Description:

ITEM	TYPE	DESCRIPTION
Command	BYTE	ISO_OUTPUT, 0Bh
Class	BYTE	ISO command class byte
Instruction	BYTE	ISO command instruction byte
P1	BYTE	ISO command P1 byte
P2	BYTE	ISO command P2 byte
Length	BYTE	ISO command length byte. Number of bytes expecting to read from the card

Response Format:

I-block
ResultCode [[Data] ME1 ME2]

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Successful communication with card, data read
RESULT_NO_ICC_POWER	The ICC has not yet been powered on
RESULT_BYTE_ERROR	Error reading byte, eg parity error
RESULT_BAD_PB	Bad procedure byte returned from card
RESULT_CARD_REMOVED	ICC has been removed whilst power was applied
RESULT_NO_CARD	No card is inserted into the TERMINAL
RESULT_NO_RESPONSE	No response from the card
RESULT_WRONG_PROTOCOL	Card uses T=1 protocol, cant use ISOInput
RESULT_CARD_TYPE_NOT_SET	On last card reset, no ATR was read, and no subsequent DefineCardType command was performed
RESULT_DATA_NOT_AVAILABLE	Successful communication with card, no data read

The following data item is only present if ResultCode is RESULT_SUCCESS:

ITEM	TYPE	DESCRIPTION
Data	Variable length	Data read from the card. Number of bytes in this field is equal to the Length parameter specified in the ISOOutput command.

The following data items are only present if ResultCode is RESULT_SUCCESS

5 or RESULT_DATA_NOT_AVAILABLE:

ITEM	TYPE	DESCRIPTION
ME1	BYTE	Returned by ICC as result of command.
ME2	BYTE	Returned by ICC as result of command.

ISOT1

- On receipt of this command, the **TERMINAL** sends a command to the inserted smartcard together with any optional data using the T=1 protocol. The **TERMINAL** reads the response from the card, which may include data. Due to the nature of the T=1 protocol, it is not necessary to distinguish between data going to the card or data coming out of the card. The protocol is capable of handling data in either or both directions in the same command cycle. This command can only be used for asynchronous cards using the T=1 asynchronous protocol. Use the **ISOInput/ISOOutput** commands for all other cards.

Command Format:

I-block
Command T1Data

Command Description:

ITEM	TYPE	DESCRIPTION
T1Data	Variable length	Data to be sent to card. No interpretation is made on the contents of T1Data by the TERMINAL . To be accepted by a 7816-3 compliant card, it should be of the form: Class Instruction P1 P2 Length [Data]. Note that this is a variable length data item, and the I-block length is used to determine this item's length.

Response Format:

I-block
ResultCode [T1Data]

Response Description:

RESULTCODE (BYTE)	DESCRIPTION
RESULT_SUCCESS	Successful communication with card.
RESULT_NO_ICC_POWER	The ICC has not yet been powered on
RESULT_BLOCK_CHECK_ERROR	Unrecoverable errors reading T1 frame
RESULT_CARD_REMOVED	ICC has been removed whilst power was applied
RESULT_NO_CARD	No card is inserted into the TERMINAL
RESULT_NO_RESPONSE	No response from the card
RESULT_WRONG_PROTOCOL	Card doesn't use T=1 protocol, cant use ISOT1

The following data item is only present if ResultCode is RESULT_SUCCESS:

ITEM	TYPE	DESCRIPTION
T1Data	Variable length	Data returned from the card. No interpretation is made on the contents of T1Data by the TERMINAL. Sent from a 7816-3 compliant card, it will be of the form: [Data] ME1 ME2. Note that this is a variable length data item, and the I-block length is used to determine this item's length.

L-Frame Command Full Functional Definitions

For such commands, the LEN is 2 and the I-block is

Command I-block format:

I-block
Major Minor

Data Description:

ITEM	TYPE	DESCRIPTION
Major	BYTE	Major version number
Minor	BYTE	Minor version number

5 Response I-block format:

I-block
Major Minor

Data Description:

ITEM	TYPE	DESCRIPTION
Major	BYTE	Major version number
Minor	BYTE	Minor version number

E-Frame Command Full Functional Definitions

All application layer event notifications sent by the TERMINAL to the PC are placed within E-frames. These are frames that are unsolicited by the PC, and are sent by the TERMINAL when certain events occur. The format of the I-

- 5 block of each E-frame is:

EVENTCODE	DATA
1 byte	0 or more bytes, event dependent

This section defines each event notification.

Smartcard Insertion

- 10 This event occurs when the TERMINAL detects a smartcard insertion into the specified smartcard port.

Event format:

EVENTCODE	DATA
00h	Port

Data Description:

ITEM	TYPE	DESCRIPTION
Port	BYTE	Currently set to 0

Smartcard Removal

- 15 This event occurs when the TERMINAL detects a smartcard removal from the specified smartcard port.

Event format:

EVENTCODE	DATA
00h	Port

Data Description:

ITEM	TYPE	DESCRIPTION
Port	BYTE	Currently set to 0

M-Frame Command Full Functional Definitions

- M-frames are event frames and are sent by the **TERMINAL** when it detects a change in the state of its buttons or when sufficient X-Y movement has occurred. M-frames are the only frames not to conform to the frame structure above. Each M-frame is a fixed 4 bytes in length, and has the following format:

STX	PCB	X Position	Y Position
-----	-----	------------	------------

The following definitions apply:

	Bit							
Byte	7	6	5	4	3	2	1	0
PCB	0	1	L-BTN	R-BTN	Y7	Y6	X7	X6
X-Position	R	R	X5	X4	X3	X2	X1	X0
Y-Position	R	R	Y5	Y4	Y3	Y2	Y1	Y0

Where:

L-BTN	The current state of the left mouse button, 1 = pressed
R-BTN	The current state of the right mouse button, 1 = pressed
X	X7... X0; 8 bit signed number reflecting the mouse movement along the horizontal axis since the last M-frame. The positive X axis is to the right
Y	Y7... Y0; 8 bit signed number reflecting the mouse movement along the vertical axis since the last M-frame. The positive Y axis is upwards

The personal computer arbitrates, at the device driver level, between the reception of the X-Y position input control information and the smartcard information which is transmitted from the terminal device, and the smartcard information which is received by the terminal device. The arbitration ensures correct extraction of the relevant information and correct passing of the information to the appropriate higher level.

The key mechanism which enables the device driver of the personal computer to extract the relevant information from the terminal device communications protocol is the addition of a distinct per frame slave command field. This mechanism allows the device driver of the personal computer to distinguish between commands which are intended for a higher level smartcard application, or commands which are intended to control the X-Y position of the cursor of the personal computer.

For example Table 5 below shows an example of a method by which the personal computer arbitrates messages between smartcard terminal commands and X-Y position information commands. If the command is less than a certain value, then the command is a smartcard terminal command. If the command is greater than or equal to a certain value, then the command is an X-Y position information command.

PCB	Command Type
<Value	Smartcard Terminal Command
>=Value	X-Y Position Information Command

Table 5.

Power is provided to the terminal device by utilising the existing power supplies of the personal computer. The source of power for the terminal device is the keyboard port of the personal computer. A special cable is used that routes the raw keyboard power source, to be then stepped down, and back up to power the smartcards.

Although the invention has been described with reference to a preferred embodiment it should be appreciated that it could be embodied in other forms. For instance, the terminal device may be capable of
5 communicating with many smartcards at the same time by being equipped with several smartcard readers which are accessed through respective slots in the housing. The mouse may also include a biometric sensor, such as a thumbscanner, built into the side of the housing.

The invention has been described with reference to a mouse, but it
10 could also be incorporated in a trackball, joystick or any other pointing device. The invention has been described with reference to a personal computer, but it could also be used with a workstation, a network server, a network computer or a computer terminal. The cable could also be replaced by an optical transport medium or a radio frequency transport medium.

15 It is also envisaged that the terminal could be constructed without the mouse (or other pointing device) being integrated into the same housing as the remainder of the terminal. This arrangement will allow the terminal to be retrofitted to existing systems at lower cost. The arrangement is shown in Figure 3.

20 In Figure 3 the integrated smartcard terminal device 12 comprises an external plastics casing 2. An horizontal slot 6 along the front can receive a smartcard. A socket 13 will accept a plug from a standard mouse.

Within the device 12 there are also conventional smartcard reading and writing heads (not shown) together with associated circuitry. The information
25 being read from the smartcard is transmitted to a personal computer together with X-Y position input control information along a cable 8. Information to be written to the smartcard is also transmitted back along the same cable 8.

The device 12 has a keypad 10 which may include a standard keyscanning mechanism or a secure keyscanning mechanism.

A bicolour, red and green, LED 9 is provided for indicating the logical status of the smartcard interface of the terminal device 12.

A bidirectional communications transport protocol is duplicated at either end of the cable 8, both in the terminal device 12 and at the device
5 driver level of the personal computer (not shown).

In addition the example arbitration mechanism described is not necessarily the only way to arbitrate the signals, and they could for instance be arbitrated by having a code allocated to them.

It will be appreciated by persons skilled in the art that numerous
10 variations and/or modifications may be made to the invention as shown in the specific embodiments without departing from the spirit or scope of the invention as broadly described. The present embodiments are, therefore, to be considered in all respects as illustrative and not restrictive.

**CLAIMS**

1. A smartcard terminal, including:
a smartcard reader; and
a data assembly and transmission means to receive position data from
5 a computer display pointing device having a serial communications capability
and from the smartcard reader, and to transmit position information from the
pointing device, and smartcard information from the smartcard reader to an
associated computer, and to receive information which is to be written to the
smartcard from the computer;
10 wherein the data assembly and transmission means assembles the
position information and the smartcard information into separate frames
before transmitting them serially, and it labels each frame depending on
whether it contains position information or smartcard information.
- 15 2. A smartcard terminal according to claim 1, wherein the smartcard
reader and the data assembly and transmission means are housed together in a
housing which includes a serial data port to receive the position data.
3. A smartcard terminal according to claim 1, wherein the terminal
20 includes a housing that houses both the smartcard reader and the computer
display pointing device.
4. A smartcard terminal according to any preceding claim, when used for
offline and online access control, offline and online smartcard initialisation,
25 online authentication, online certification, online end to end public and
private key encryption, online bidirectional transactions, including purchase
transactions and withdrawals which reload an electronic purse on the
smartcard for later offline payments.



5. A smartcard terminal according to any preceding claim, able to communicate with many smartcards at the same time by being equipped with several smartcard readers.
- 5 6. A smartcard terminal according to any preceding claim 1, that includes keypad for keypad information entry, and wherein the data assembly and transmission means assembles the position information, the smartcard information and any keypad information into separate frames before transmitting them serially
- 10 7. A smartcard terminal according to claim 3, wherein the terminal incorporates a mouse.
8. A smartcard terminal according to any preceding claim, wherein the
- 15 terminal is used with a personal computer.
9. A smartcard terminal according to any preceding claim, wherein the terminal employs a cable to communicate with the computer.
- 20 10. A smartcard terminal according to any preceding claim, wherein the communications means assembles the information into frames which start with a start of text character (STX) to identify the start of the frame.
11. A smartcard terminal according to claim 10, wherein the next byte
- 25 following the start of frame character is a protocol control byte (PCB).
12. A smartcard terminal according to claim 11, wherein if the protocol control byte (PCB) is any other than an X-Y position then the next byte following is a data length character (LEN) which specifies the length of the
- 30 subsequent information bytes.



13. A smartcard terminal according to claim 12, wherein the format is shown in the following Table:

STX	PCB	LEN	Information Block	ETX	BCC
-----	-----	-----	-------------------	-----	-----

5

The following definitions apply:

STX	ASCII 02H
PCB	The protocol control byte of the message
LEN	The length of the information block in bytes
Information Block	Command data to the terminal
ETX	ASCII 03H
BCC	The block check of the information bytes