TOSHIBA

Windows POS Terminal

TEC OPOS Control

USB MSR OPOS Application Programmer's Guide for PKBST

2nd Edition : 16 Apr, 2012

TOSHIBA TEC CORPORATION

Revision Record

No. EAA-02151

USB MSR OPOS Application Programmer's Guide for PKBST

Rev. No.	Date	Sections	Description
1	05 Feb, 2010	---	First Release
2	16 Apr, 2012	all	Change format

Table of Contents

1. Magnetic Stripe Reader 1
1.1 USB PKBST MSR CONTROL ["MCRSTUB"] 1
1.1.1 Applicable Models and Operating Systems 1
1.1.2 Software Structure 2
1.1.3 Functions 3
1.1.4 CheckHealth Method Specifications 4
1.1.5 DirectIO Specifications 6
1.1.6 OPOS Registry 7
1.1.7 Limitations and Precautions 10
1.1.8 Usage Example 12
1.1.9 Log 14
1.1.10 Result When Property/Method is Executed 15

© Copyright Toshiba TEC Corporation All rights reserved. It is prohibited to use or duplicate a part or whole of this document without the permission of Toshiba TEC Corporation. This document is subject to change without prior notice.

Trademark Notification

* Microsoft, Windows, Windows XP, WEPOS 1.1, POSReady 2009, Windows 7 and POSReady 7 are registered trademarks of Microsoft Corporation in the United States and/or other countries. The official name of Windows is the "Microsoft Windows Operating System".

* All other product names mentioned in this document are trademarks or registered trademarks of their respective owners.

1. Overview

This specification describes the matters that require attention and the methods Application Programming to utilize TEC OPOS Control.

It is assumed that the reader already possesses some knowledge of:

- General features of the POS peripherals.
- General features of TEC POS Terminal and TEC POS peripherals.
- Terms and Architecture for the OLE Control and OLE Automation.
- The OLE for Retail POS (abbr. OPOS) and the Application Programmer's Guide (abbr. APG).

2. Magnetic Stripe Reader

2.1 USB PKBST MSR CONTROL ["MCRSTUB"]

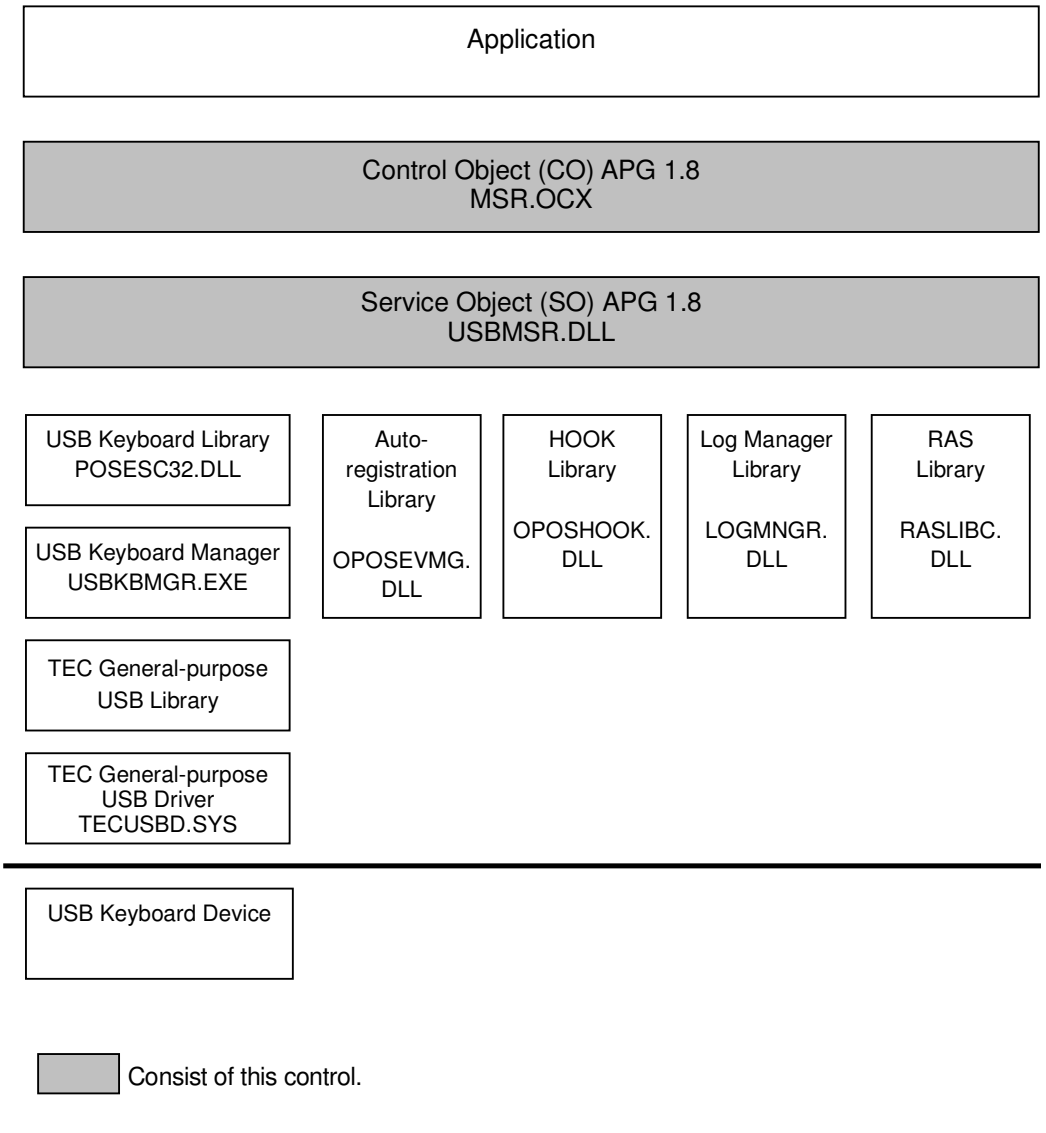
2.1.1 Applicable Models and Operating Systems

Model	Interface	Device Name (*1)
PKBST-52 (USB) MSR	USB I/F	"MCRSTUB"
Operating System		
Windows XP Professional Windows Embedded for Point of Service(WEPOS) Windows Embedded POSReady 2009 Windows 7 Professional Windows Embedded POSReady 7		

(*1) Device names are used by the Open method.

2.1.2 Software Structure

The software structure of the USB PKBST MSR Control is as shown below.



2.1.3 Functions

Functions supported	Functions not supported
Power notification	Reading of JIS-I track
Error report about card level	Collection of statistics
Error report about track level	Reset of statistics
Reading of ISO, JIS-II track	Change of statistics
Reading of start and end sentinels (*1)	

*1: This function is not supported if “DataType”=”MCRST” is described in the registry.

USB PKBST MSR Control – Functions

The following table shows only the device-dependent properties.

Common property	Value
ControlObjectDescription	“TEC OPOS MSR Control Object”
ControlObjectVersion	“1008XXX” (*2)
ServiceObjectDescription	“TEC OPOS USB Keyboard MSR Service Object”
ServiceObjectVersion	“1008XXX” (*2)
DeviceDescription	“MSR on PKBST-5xU POS Keyboard”
DeviceName	“MCRST UB”
CapPowerReporting	OPOS_PR_STANDARD
CapStatisticsReporting	FALSE
CapUpdateStatistics	FALSE
Exclusive property	**Value**
CapISO	TRUE
CapJISOne	FALSE
CapJISTwo	TRUE
CapTransmitSentinels	TRUE (*1)

(*1) The value will become FALSE if “DataType”=”MCRST” is described in the registry.
(*2) Build version is indicated as “XXX” because this document may not be revised every time the module is updated.

USB PKBST MSR Control – Property Values (in part)

2.1.4 CheckHealth Method Specifications

1) Internal Level (OPOS_CH_INTERNAL)

Return Value (ResultCode)	CheckHealthText	Meaning
OPOS_E_ILLEGAL	“Internal Hcheck:Illegal”	Not supported.

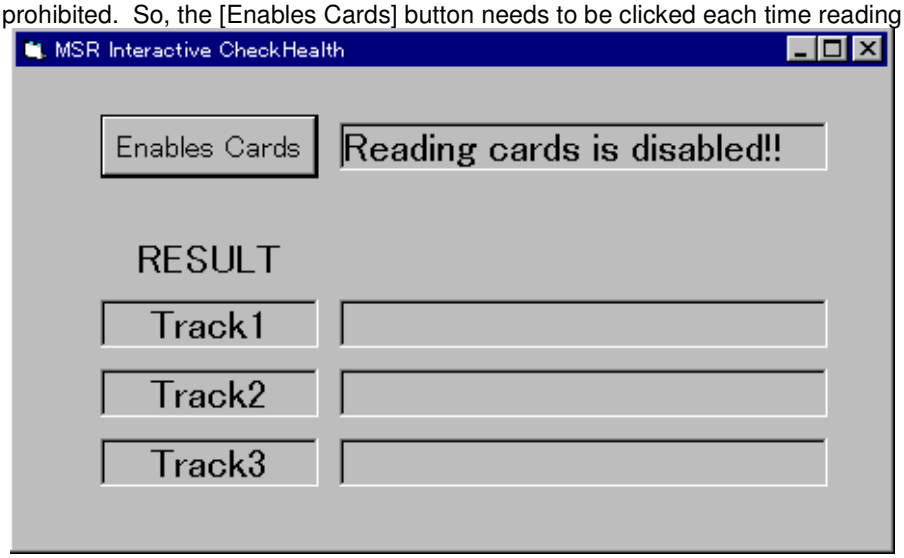
2) External Level

This requests the device to enable reading a card. Reading the card is prohibited if this request succeeds.

Return Value (ResultCode)	CheckHealthText	Meaning
OPOS_SUCCESS	“External HCheck: Successful”	Successfully completed.
OPOS_E_NOHARDWARE	“External HCheck: Not Support”	Not supported.
OPOS_E_ILLEGAL	“External HCheck:Driver Error”	Driver error
OPOS_E_DISABLED	“HCheck: Disabled”	DeviceEnabled=TRUE has not been executed.
OPOS_E_NOTCLAIMED	“Hcheck: Exclusive”	ClaimDevice has not been executed.

3) Interactive Level (OPOS_CH_INTERACTIVE)

It will show following dialog box. After the [Enables Cards] button is clicked and a card is swiped, the data in the card is checked and its result is stored in CheckHealthText. Once a card is read, subsequent reading is prohibited. So, the [Enables Cards] button needs to be clicked each time reading a card.



Enable MSR Error Messages

Message	Description
Reading card is disabled!!	Reading card is prohibited as default. Once a card is read, subsequent swiping is prohibited.
Read Card!!	Reading card is enabled.
Missed enabling reading card!!	Enabling reading card failed.

Track Result Messages

Message	Description
Normal End	Reading data succeeded.
Start Error	Reading data failed due to Start Error.
LCR Error	Reading data failed due to LRC Error.
No Data	No data

Return Value (ResultCode)	CheckHealthText	Meaning
OPOS_SUCCESS	“Interactive Hcheck:Successful”	Successfully completed.
OPOS_E_FAILURE	“Interactive Hcheck:Data Error”	Terminated with an error (Start Error/LRC Error).
OPOS_E_ILLEGAL	“External HCheck:Driver Error”	Driver error
OPOS_E_NOHARDWARE	“External Hcheck: Not Support”	Not supported.
OPOS_E_DISABLED	“HCheck: Disabled”	DeviceEnabled=TRUE has not been executed.
OPOS_E_NOTCLAIMED	“Hcheck:Exclusive”	ClaimDevice has not been executed.

Note: For ResultCode/CheckHealthText, the last value entered will be retained.

2.1.5 DirectIO Specifications

The following table shows the extended functions supported by DirectIO method of this control.

Command	Function
TMSR_CMD_SET_CHECKCARD	Specifies the check function for improper data in the card.
TMSR_CMD_GET_CHECKCARD	Obtains the card data check setting.

The constants used for the DirectIO method of this control are defined in *TECMSR.H* provided by TOSHIBA TEC. Please note that this file may be revised when the module is updated. We would recommend you to use the appropriate file for the module to be used.

(1) Specifying the card data check function

Function Generates an LRC error if improper data is contained in card data.

Format

Parameter	Description
Command	TMSR_CMD_SET_CHECKCARD
pData	0: Card data check is not performed. 1: Card data check is performed.
pString	Not used. (Specify a null character string ““.)

Description Mandatory requirements are Open and Claim.
This function checks whether or not improper data is contained in card data, and if yes, generates an LRC error.
When the card data check is enabled, reading a card containing improper data causes ErrorEvent, which indicates an LRC error, to be generated.
Effective data ranges from 0x20 to 0x7F when decoded into ASCII code. Other data is considered as improper data.

Note This function checks whether character code before a null (0x20) is contained in the card data, and if yes, such data is handled as error data. In the case the program processes error data properly, it is not necessary to use this function.

Return value One of the following values will be returned and also stored in the ResultCode property.

Value	Meaning
OPOS_SUCCESS	The processing succeeded.
OPOS_E_CLOSED	The device has been closed.
OPOS_E_NOTCLAIMED	Exclusive access has not been granted.
OPOS_E_ILLEGAL	The command or pData is wrong. Or pData is NULL.

(2) Obtaining the card data check setting

Function Obtains the setting specified by using the TMSR_CMD_SET_CHECKCARD command.

Format

Parameter	Description
Command	TMSR_CMD_GET_CHECKCARD
pData	The current setting is stored in the field specified by this parameter. 0: Card data check is not performed. 1: Card data check is performed.
pString	Not used. (Specify a null character string ““.)

Description Mandatory requirement is Open.
This is used to obtain the setting specified by using the TMSR_CMD_SET_CHECKCARD command.

Note

Return value One of the following values will be returned and also stored in the ResultCode property.

Value	Meaning
OPOS_SUCCESS	The processing succeeded.
OPOS_E_CLOSED	The device has been closed.
OPOS_E_ILLEGAL	The command or pData is wrong. Or pData is NULL.

2.1.6 OPOS Registry

The OPOS registry contains the following configuration information:

HKEY_LOCAL_MACHINE\SOFTWARE\OLEforRetail\ServiceOPOS\MSR\MCRSTUB

General	“TEC.USBMSR1”
Service	“C:\OPOS\TEC\USBMSR.DLL”
Description	“TEC MSR on PKBST-5xU Keyboard”
Version	“1.8”
DataType	“MCSRST” | “Standard”
Vkey	“VK_F13” to “VK_F24” (“VK_F16”)
DataType	“MCRST” | “Standard” | ”TransmitSentinels”
StartSentinel	Character string which defines the start sentinel to be added.
Track2AnalysisType	“0”|”1”
MSRDataCheck	“0”|”1”
DebugLogLevel	“0” | ”1” | ”2”
DebugLogFile	“C:\OPOS\TEC\LOG\MCRSTUB.LOG”
ErrorReportingType	“CARD” | “TRACK”
ErrorMaskLevel	“0” | “1”

Service	Filename of Service Object
Description	Brief explanation of Service Object
Version	Version number of Service Object
DataType	Specifies the data type of the card. There are two options as follows: MCRST: Data format exclusively for the MCRST.

<table>
<tr><td></td><td>Start sentinel is deleted, and the LRC is converted into 2-byte format.
Standard: OPOS APG standard data format
Start sentinel, end sentinel, and LRC are deleted.
The data type can be selected through the control panel.</td></tr>
<tr><td>Vkey</td><td>Specifies the virtual key code which is used for controlling the entry sequence.
Selectable from “VK_F13” to “VK_F24”
The USB MSR Control uses “VK_F16” as standard, but this is changeable through the control panel.</td></tr>
<tr><td>DataType</td><td>Specifies the card data formats from the following three types:
MCRST: Data format exclusively for the MCRST
Start Sentinel is deleted.
Standard: OPOS APG standard data format
Start/End sentinels and LRC are deleted.
TransmitSentinels: Start sentinel is added according to the StartSentinel parameter, and the LRC is deleted.
MCRST and Standard can be selected through the control panel.</td></tr>
<tr><td>StartSentinel</td><td>In order to handle a card including special ISO track, it is possible to specify an optional character as the start sentinel in the following format.
<dlm> track 1 <dlm> track 2 <dlm> track 3
<dlm> Character string used as a separator
Track 1 Character to be used as the start sentinel for track 1
Track 2 Character to be used as the start sentinel for track 2
Track 3 Character to be used as the start sentinel for track 3

[Example 1]
Start sentinel setting: Track 1 = ‘%’, Track 2 = ‘;’, Track 3 = ‘;’ (Same as standard).
Separator: ‘#’
“StartSentinel”=”#%#;#;”
[Example 2]
Start sentinel setting: Track 2 = ‘<’
Separator: ‘,’
“StartSentinel”=”,,<”
[Example 3]
Start sentinel setting: Track 1 = ‘!’, Track 3 = ‘>’
Separator: ‘,’
“StartSentinel”=”,!,,>”
[Note]
Characters applicable to the start sentinel are limited as follows: Describe as a value after decoding. Only ISO track is applicable.
<table>
<tr><td>Track 1</td><td>0x20 to 0x5F (ASCII code ‘ ’ to “_”)</td></tr>
<tr><td>Track 2 and Track 3</td><td>0x30 to 0x3F (ASCII code ‘0’ to ‘?’)</td></tr>
</table></td></tr>
</table>

Track2AnalysisType	Specifies how to process ISO Track 2. 0 On the expiration date, obtains the service code only from track 1. 1 On the expiration date, obtains the service code from track 1 in priority to other tracks. If no service data exists in track 1, then obtain it from track 2.
MSRDataCheck	Specifies whether to check the card data for any improper data. Proper card data ranges from 0x20 to 0x7F, and if the data is outside of this range, an LRC error results. 0 Card data check is not performed. 1 Card data check is performed. This parameter can be temporarily changed from the application by using DirectIO method. The registry cannot be changed with this method.
DebugLogLevel	Specifies the extent of the log data to be output. There are three options: "0" The log is not output. "1" Basic log is output. "2" Detailed log is output.
DebugLogFile	Specifies the full pathname of the file where the log is written. If the specified directory is not found, it will not be created automatically.
ErrorReportingType	Specifies the initial value of the error detection level. If this parameter is omitted or an invalid value is set, "CARD" will be effective. The value set here will be the initial value of the ErrorReportingType property. "CARD": Detects a card level error. Track data is notified only when reading the all track data in the card succeeded. If reading even one track data failed, no data is notified. "TRACK": Detects a track level error. Track data is notified when reading any one of the track data in the card succeeded. If reading every track data failed, no track data is notified.
ErrorMaskLevel	Specifies the processing when the track of which data reading succeeded and the one failed are mixed in a card. If this parameter is omitted or an invalid value is set, "1" will be effective. This parameter is effective only when the ErrorReportingType property is set to "TRACK". "0": Notifies of the information of the track, of which data reading failed, by the ErrorEvent event, and the information of the track, of which data reading succeeded, by DataEvent event. "1": Notifies only of the information of the track, of which data reading succeeded, by the DataEvent event. The information of the track, of which data reading failed, is not notified.

USB PKBST MSR Control – Registries

2.1.7 Limitations and Precautions

1) JIS Type-II Card

When a JIS Type-II card is read, the data will be processed as Track 1 data. ISO Track 1 data will be also stored in the same property. Decoding and division of the field are automatically determined and processed. Although the PKBST-50/51 is equipped with two reading heads, these two heads will not read data at the same time. From the OPOS APG1.5, data of JIS Type-II card is stored in Track4Data property, and also in Track1Data to maintain the compatibility with the lower version of the OPOS APG.

2) Format of TrackxData property

In the case DataType="MCRST" is specified in the registry, the format of the TrackxData property will be as follows. In the case of DataType="Standard", the end sentinel and LRC can be deleted.

(1) Data X characters
(2) End sentinel One character
(3) LRC Two characters

The start sentinel is not stored. The LRC is not one character, but converted 2 characters.
When the LRC is 0x1b on the card, DecodeData=TRUE or Track1Data is expressed as 0x31,0x62('1','b').
In the case of DecodeData=FALSE and non Track1Data, the LRC will be the value which cannot be expressed as a character string, but expressed with two characters.

3) Precaution in use of TransmitSentinels property

This property properly operates only when DataType="Standard" is specified in the registry. In the case of DataType="MCRST", CapTransmitSentinels property is set to FALSE because, in the first place, the start sentinel cannot be notified.
Setting TransmitSentinels property to TRUE enables attaching the start sentinel and end sentinel to the track data stored in Track*n*Data property. However, doing this causes the data check not to be performed, and eventually the data including an LRC error or end sentinel error could be stored in Track*n*Data property. The application requires checking the data stored in the property for any errors.
Check that the data stored in Track*n*Data property starts with the start sentinel and ends with the end sentinel because erroneous data sometimes does not.

4) Notification of Track Level error

The MSR OPOS control may return OPOS_EL_INPUT_DATA as ErrorLocus of ErrorEvent event in case of an input error, only when the notification of the track level error is specified (ErrorReportingType=MSR_ERT_TRACK).
In the case the notification of card level error (ErrorReportingType=MSR_ERT_CARD) is specified, when any queuing data exists, ErrorEvent cannot be received until queued data is input even if an input error occurs. However, State property turns to OPOS_S_ERROR at the time ErrorEvent is put into queue, so it is possible to detect an error state.

5) Auto sensing of the connected port

The USB keyboard is automatically sensed by the Service Object determined according to the open name. However, when two or more units of identical keyboard are connected, which device is sensed is unknown.

6) Plug-and-Play for USB Keyboard in operation

The USB keyboard is a plug-and-play device, but a disconnection and reconnection of the connector during operation is not recommended.

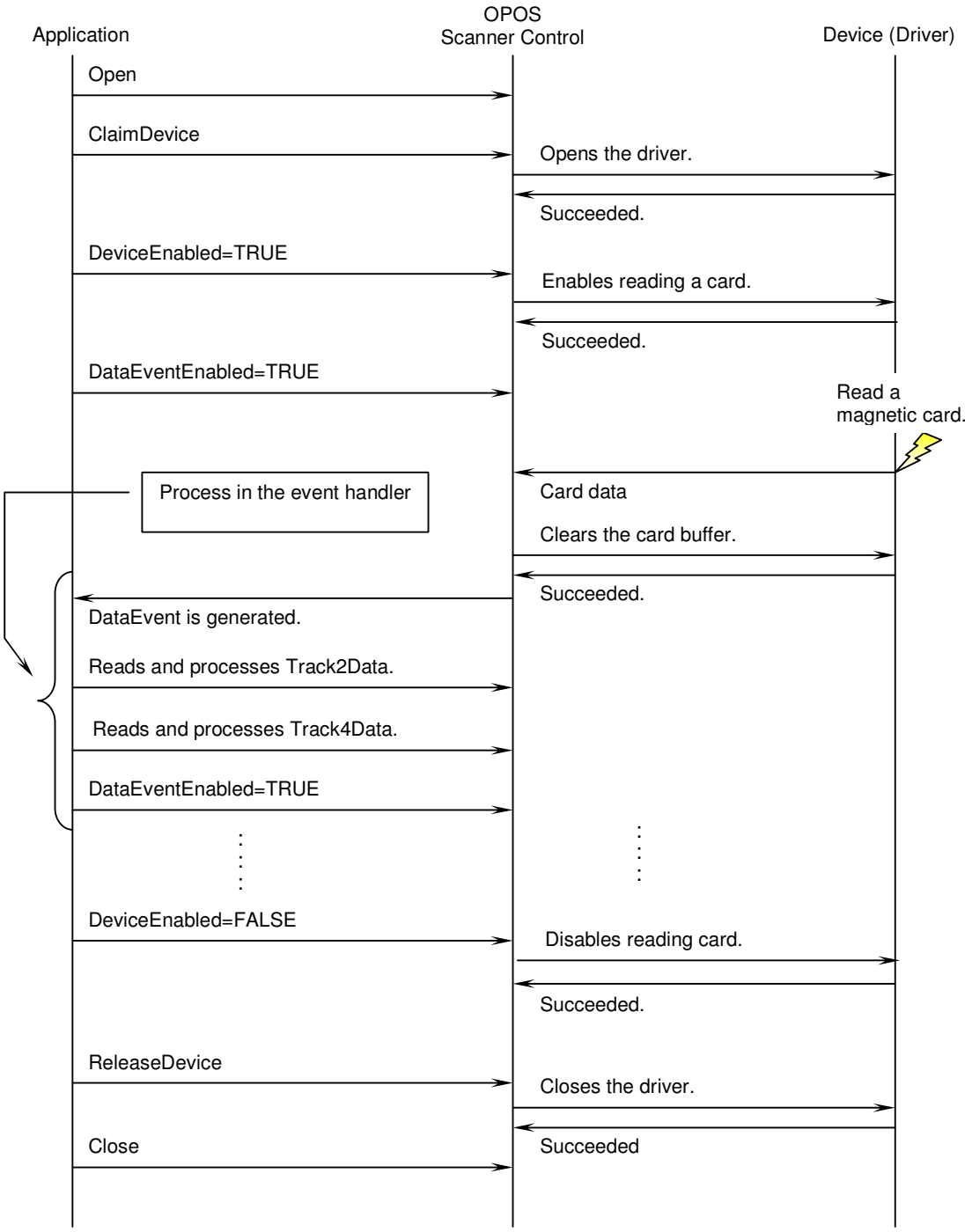
7) Notification of the power status
This control determines whether the power status of the keyboard is in “ON LINE” or “OFF or OFFLINE”, and notifies of the result. This is substantially corresponding to the disconnection and reconnection of the USB keyboard connector. Though ClaimDevice for unconnected USB device succeeds, subsequent setting of DeviceEnabled property to TRUE results in OPOS_E_NOHARDWARE error. Accordingly, the notification of the power status can be started only while the device is connected. The subsequent operations follow the PowerNotify property.

8) Operation of TracksToRead property
Reading accuracy and reading speed cannot be controlled by this property.

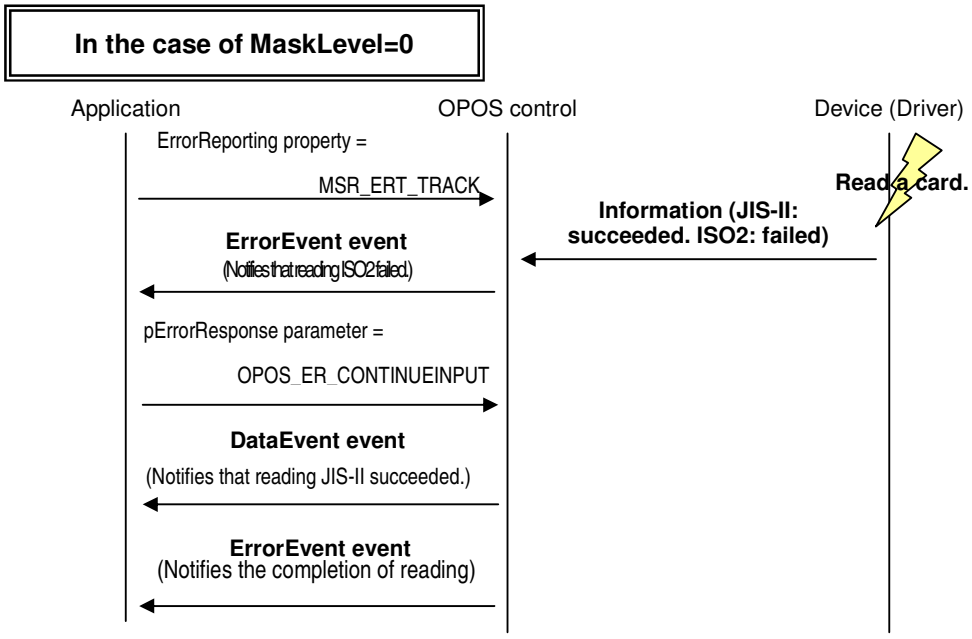
9) DeviceEnabled Property
If DeviceEnabled property is operated while a card is read, OPOS_E_BUSY is stored in ResultCode property.

2.1.8 Usage Example

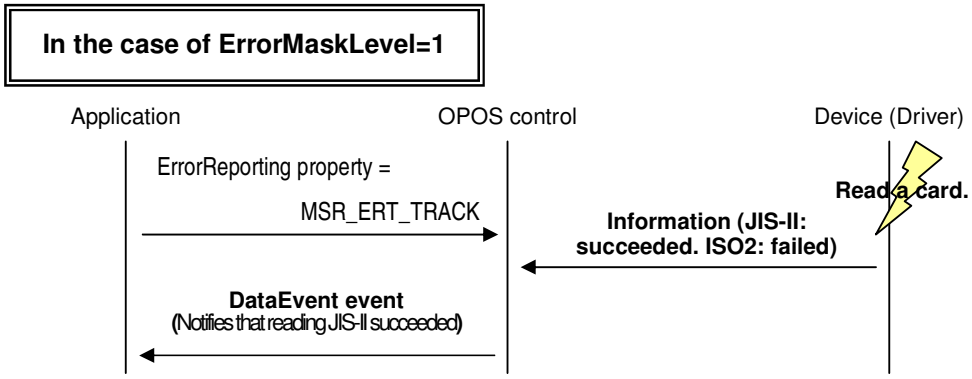
This section gives a sequence diagram to show an operation flow from reading data on JIS-II track and ISO Track 2 of a magnetic card to a close operation.



This section describes the difference of the operation depending on the ErrorMaskLevel setting in the registry. The following diagram shows an example where a card including JIS-II track and ISO track 2 is read and reading the JIS-II track succeeds and ISO track 2 fails.



In the case of ErrorMaskLevel=0, it is possible to know that reading the ISO2 failed. However, the processing is a bit difficult due to generation of more than one ErrorEvent event.



In the case of ErrorMaskLevel=1, it is impossible to know that reading the ISO2 failed because the notification of the ErrorEvent is not generated. However, the processing is simple since only the information of success in reading the JIS-II is notified by the DataEvent event.

2.1.9 Log

This Control outputs the trace log and error log according to the registry.
The specification of the log is shown below.

(1) File name
PKBSTUB.LOG

(2) Size
Max. 4MB
After the data size reached the maximum, the new data afterwards will overwrite the chronologically oldest one.

(3) Format
Date, Time, [Function]<Error information>

(4) Log level
The extent of log data to be output depends on the following log levels.
0: No log is output.
1: Minimum log is output. Error logs are always output when the log level is set to “1”.
2: Detailed log is output.

(5) Log data
No explanation is made in this document.

2.1.10 Result When Property/Method is Executed

The OPOS Control notifies the user of a result when a property/method is executed.

The table below shows the ResultCode Property values and OpenResult Property values described in this document.

ResultCode	Value
OPOS_SUCCESS	0
OPOS_E_CLOSED	101
OPOS_E_CLAIMED	102
OPOS_E_NOTCLAIMED	103
OPOS_E_NOSERVICE	104
OPOS_E_DISABLED	105
OPOS_E_ILLEGAL	106
OPOS_E_NOHARDWARE	107
OPOS_E_OFFLINE	108
OPOS_E_NOEXIST	109
OPOS_E_EXISTS	110
OPOS_E_FAILURE	111
OPOS_E_TIMEOUT	112
OPOS_E_BUSY	113
OPOS_E_EXTENDED	114

OpenResult	Value
OPOS_OR_ALREADYOPEN	301
OPOS_OR_REGBADNAM	302
OPOS_OR_REGPROGID	303
OPOS_OR_CREATE	304
OPOS_OR_BADIF	305
OPOS_OR_FAILEDOPEN	306
OPOS_OR_BADVERSION	307
OPOS_ORS_NOPORT	401
OPOS_ORS_NOTSUPPORTED	402
OPOS_ORS_CONFIG	403
OPOS_ORS_SPECIFIC	450
OPOS_ORS_BADCO	451
OPOS_ORS_RESOURCEFAIL	452
OPOS_ORS_ALREADYOPENED	453

1) Results When Property is Executed
The table below gives the results common to all available properties and those unique to certain property.

Property	ResultCode	Meaning	Error Handling
Common properties	OPOS_SUCCESS	Property setting was completed successfully.	–
	OPOS_E_CLOSED	The device has been closed.	Open the device using the Open method, then perform a setting again.
BinaryConversion	OPOS_E_ILLEGAL	An invalid value was specified.	Specify a valid value.
DeviceEnabled	OPOS_E_NOTCLAIMED	Exclusive access has not been granted.	Execute the Claim method, then perform a setting again.
	OPOS_E_NOHARDWARE	The device is not connected or its power is not turned on.	Make sure the device is connected and its power is on, then perform a setting again. If the error occurs again, install the OPOS control and the USB driver again. When the OPOS control and the USB driver have been properly installed, re-install the driver kit suitable for the model of the POS terminal, then perform a setting again. If the error still persists, then investigate the error.
	OPOS_E_FAILURE	Enabling reading a card, or setting the beep sound for card reading or read mode failed.	Make sure the device is connected and its power is on, then perform a setting again. If the error occurs again, install the OPOS control and the USB driver again. When the OPOS control and the USB driver have been properly installed, re-install the driver kit suitable for the model of the POS terminal, then perform a setting again. If the error still persists, then investigate the error.
	OPOS_E_BUSY	An attempt was made to change the property while a card is being read.	Perform a setting again in the idle state.
PowerNotify	OPOS_E_ILLEGAL	An invalid value was specified.	Specify a valid value.
		The device is already enabled.	Unless CapPowerReporting property is OPOS_PR_NONE, the device is already enabled. Set DeviceEnabled property to FALSE, then perform a setting again.
		Power notification function is not supported.	In the case the CapPowerReporting property is OPOS_PR_NONE, the power notification function is not supported.
	OPOS_E_NOSERVICE	The version of the service object is so old that this property is not supported.	Install the latest service object.
TracksToRead	OPOS_E_ILLEGAL	An invalid value was specified.	Specify a valid value.
ErrorReportingType	OPOS_E_ILLEGAL	An invalid value was specified.	Specify a valid value.

Property	ResultCode	Meaning	Error Handling
TransmitSentinels	OPOS_E_ILLEGAL	Notification of the start and end sentinels is not supported.	When CapTransmitSentinels property is FALSE, the notification of the start and end sentinels is not supported.
	OPOS_E_NOHARDWARE	The device is not connected or its power is not turned on.	Make sure the device is connected and its power is on, then perform a setting again. If the error occurs again, install the OPOS control and the USB driver again. When the OPOS control and the USB driver have been properly installed, re-install the driver kit suitable for the model of the POS terminal, then perform a setting again. If the error still persists, then investigate the error.
	OPOS_E_FAILURE	Changing the card read mode of the keyboard failed.	Make sure the device is connected and its power is on, then perform a setting again. If the error occurs again, install the OPOS control and the USB driver again. When the OPOS control and the USB driver have been properly installed, re-install the driver kit suitable for the model of the POS terminal, then perform a setting again. If the error still persists, then investigate the error.
	OPOS_E_BUSY	An attempt was made to change the property while a card is being read.	Perform a setting again in the idle state.
	OPOS_E_NOSERVICE	The version of the service object is so old that this property is not supported.	Install the latest service object.

2) Results When Open Method is Executed
The Open method differs from other methods and is separately described.

Return Value	ResultCode	OpenResult	Meaning	Error Handling
OPOS_SUCCESS	OPOS_SUCCESS	OPOS_SUCCESS	The device was successfully opened.	—
OPOS_E_ILLEGAL	—	OPOS_OR_ALREADYOPEN	The device has already been open.	Make sure the name of the device to be opened is correct.
OPOS_E_NOEXIST	OPOS_E_CLOSED	OPOS_OR_REG_BADNAME	The specified device name does not exist in the registry.	Make sure the name of the device to be opened is correct.
		OPOS_OR_REGPROGID	The service object has not been registered correctly.	Register the driver or service object again.
OPOS_E_NOSERVICE	OPOS_E_CLOSED	OPOS_OR_CREATE	The service object has not been registered correctly.	Register the driver or service object again.
		OPOS_OR_REGPROGID	The service object has not been registered correctly.	Register the driver or service object again.
		OPOS_OR_BADIF	The service object does not support the methods required.	Register the service object again.
		OPOS_OR_FAILEDOPEN	An error occurred in the service object, but the service object does not support the OpenResult property.	Register the service object and add the registry again.
		OPOS_OR_BADVERSION	The version of the service object is invalid.	Register the service object again.
		OPOS_ORS_CONFIG	The OPOS registry for the MSR on the USB keyboard does not exist or is not correct.	Register the service object and add the registry again
			Reference to the registry failed.	If the registry for the MSR on the USB keyboard has been correctly registered, investigate the error.
		OPOS_ORS_RESOURCEFAIL	Creation of a resource to serialize the events failed. Creation of a window to process the events failed.	Investigate the error.
		OPOS_ORS_BADCO	The control object does not support the methods required.	Register a correct control object again.

3) Results when DirectIO Method is Executed
Since the result of the DirectIO method differs depending on the commands, they are described separately from the other methods.

Property	Return value/ResultCode	Meaning	Error Handling
Common properties	OPOS_SUCCESS	Processing was completed successfully.	–
	OPOS_E_CLOSED	The device has been closed.	Open the device using the Open method, then perform a setting again.
TMSR_CMD_SET_CHECKCARD	OPOS_E_ILLEGAL	The command is wrong. Or, pData is NULL or the value in the field designated by pData is improper.	Check the command and pData.
	OPOS_E_NOTCLAIMED	Exclusive access has not been granted.	Execute the Claim method, then perform a setting again.
TMSR_CMD_GET_CHECKCARD	OPOS_E_ILLEGAL	The command is wrong. Or, pData is NULL.	Check the command and pData.

4) Results When A Method Is Executed
The table below describes the result when each method other than the Open method is executed.

Method	Value/ResultCode	ResultCodeExtended	Meaning	Error Handling
Close	OPOS_SUCCESS	–	The device was successfully closed.	–
	OPOS_E_CLOSED	–	The device has already been closed.	–
Claim ClaimDevice	OPOS_SUCCESS	–	Exclusive access has been granted.	–
	OPOS_E_CLOSED	–	The device has been closed.	Open the device using the Open method, then execute the method again.
	OPOS_E_ILLEGAL	–	An invalid timeout parameter value was specified.	Specify a valid timeout value.
		–	An event thread was not created. Device input/output thread was not created.	Investigate the error.
	OPOS_E_TIMEOUT	–	While waiting for another application, which has an exclusive access to the device, to release the device, a timeout was called.	Execute the method again after another application releases the exclusive access to the device.
Release ReleaseDevice	OPOS_SUCCESS	–	Exclusive access has been released.	–
	OPOS_E_CLOSED	–	The device has been closed.	–
	OPOS_E_ILLEGAL	–	The application does not have an exclusive access to the applicable device.	–
		–	The CheckHealth method is being executed.	Execute the method again after the CheckHealth method is completed.

Method	Value/ResultCode	ResultCodeExtended	Meaning	Error Handling
CheckHealth	OPOS_SUCCESS	—	A series of health check normally terminated.	—
	OPOS_E_CLOSED	—	The device has been closed.	Open the device using the Open method, then execute the method again.
	OPOS_E_NOTCLAIMED	—	Exclusive access has not been granted.	Execute the method again after the Claim method is completed.
	OPOS_E_DISABLED	—	The device has been disabled.	Set the DeviceEnabled property to TRUE, then execute the method again.
	OPOS_E_ILLEGAL	—	Unsupported health check level was specified. Or, unexpected result was returned from the driver.	Check the health check level. If there is no problem with the health check level, investigate the error.
	OPOS_E_FAILURE	—	An error was found during a series of health check. In the case of the interactive level, this indicates the result of a user's visual check.	After eliminating the cause of the error, execute the method again. If there is no problem with the device, investigate the error.
	OPOS_E_NOHARDWARE	—	No response has been returned from the device or the device is not connected.	Check the connection status. If there are no problems with the connection, re-install the driver kit suitable for the model of the POS terminal, then execute the method again. If the error still persists, investigate the error.
ClearInput	OPOS_SUCCESS	—	Buffered input of the device was cleared.	—
	OPOS_E_CLOSED	—	The device has been closed.	Open the device using the Open method, then execute the method again.
	OPOS_E_NOTCLAIMED	—	Exclusive access has not been granted.	Execute the Claim method, then execute the method again.
ResetStatistics	OPOS_E_CLOSED	—	The device has been closed.	Open the device using the Open method, then execute the method again.
	OPOS_E_ILLEGAL	—	Either CapStatisticsReporting property or CapUpdateStatistics property is FALSE. Or, the all specified statistics are undefined or non-resettable.	In the case CapStatisticsReporting property is set to FALSE, the statistics function is not supported. In the case CapUpdateStatistics property is set to FALSE, the statistics information is non-changeable and non-resettable. Otherwise, specify the statistics having the defined name or resettable statistics.
	OPOS_E_NOSERVICE	—	The version of the service object is so old that the method is not supported.	Install the latest service object.

Method	Value/ResultCode	ResultCodeExtended	Meaning	Error Handling
RetreiveStatistics	OPOS_E_CLOSED	–	The device has been closed.	Open the device using the Open method, then execute the method again.
	OPOS_E_ILLEGAL	–	The CapStatisticsReporting property is set to FALSE. Or, the all specified statistics are undefined.	In the case the CapStatisticsReporting property is set to FALSE, the statistics function is not supported. Otherwise, specify the statistics having the defined name.
	OPOS_E_NOSERVICE	–	The version of the service object is so old that the method is not supported.	Install the latest service object.
UpdateStatistics	OPOS_E_CLOSED	–	The device has been closed.	Open the device using the Open method, then execute the method again.
	OPOS_E_ILLEGAL	–	Either CapStatisticsReporting property or CapUpdateStatistics property is FALSE. Or, the all specified statistics are undefined or non-changeable.	In the case CapStatisticsReporting property is set to FALSE, the statistics function is not supported. In the case CapUpdateStatistics property is set to FALSE, the statistics information is non-changeable and non-resettable. Otherwise, specify the statistics having the defined name or changeable statistics.
	OPOS_E_NOSERVICE	–	The version of the service object is so old that the method is not supported.	Install the latest service object.