Panasonic®

C# SDK Library API Reference Manual

C# Software Development Kit

Model No. KX-NCS6100

Thank you for purchasing this Panasonic product.
Please read this manual carefully before using this product and save this manual for future use.

Purpose and Scope

The purpose of this document is to define the API Interface which is provided by Panasonic C# Software Development Kit (SDK) version 1.1. The scope of the document is to describe how a developer will use C# SDK version 1.1 API and its features.

Note

- The main feature of C# SDK version 1.1 is the conversion of TAPI to provide easier interface to Panasonic PBX.
- End users may need to purchase a Windows Server® Client Access License (CAL) for using C# SDK library in end user application.

Trademarks

- Microsoft®, MSDN, Windows® and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
- All other trademarks identified herein are the property of their respective owners.
- Microsoft product screen shots reprinted with permission from Microsoft Corporation.

Table of Contents

1 Preface

1.1 Reference

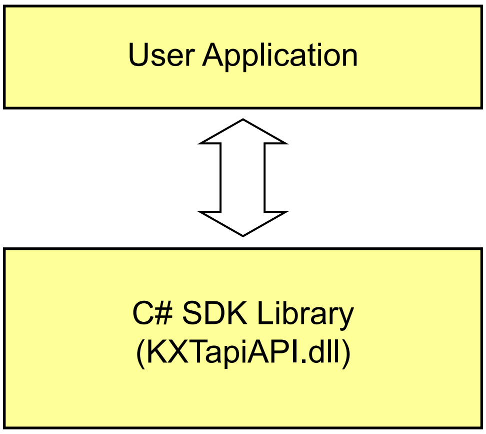
1. Panasonic KX-TDA TAPI Specification Version 3.3
2. TAPI reference from MSDN® 2009

1.2 Definitions, Acronyms, and Terminology

SDK	Software Development Kit
TSP	TAPI Service Provider
CS	Cell Station
PS	Portable Station
PT	Proprietary Telephone
DPT	Digital Proprietary Telephone
SLT	Single Line Telephone
APT	Analog Proprietary Telephone
DLL	Dynamic Link library

2 System Configuration

This section is to define the required files for user application to use Panasonic C# SDK version 1.1. The details of the required files are described in the following figure.
KXTAPIAPI.dll is the DLL library file for C# SDK library API.



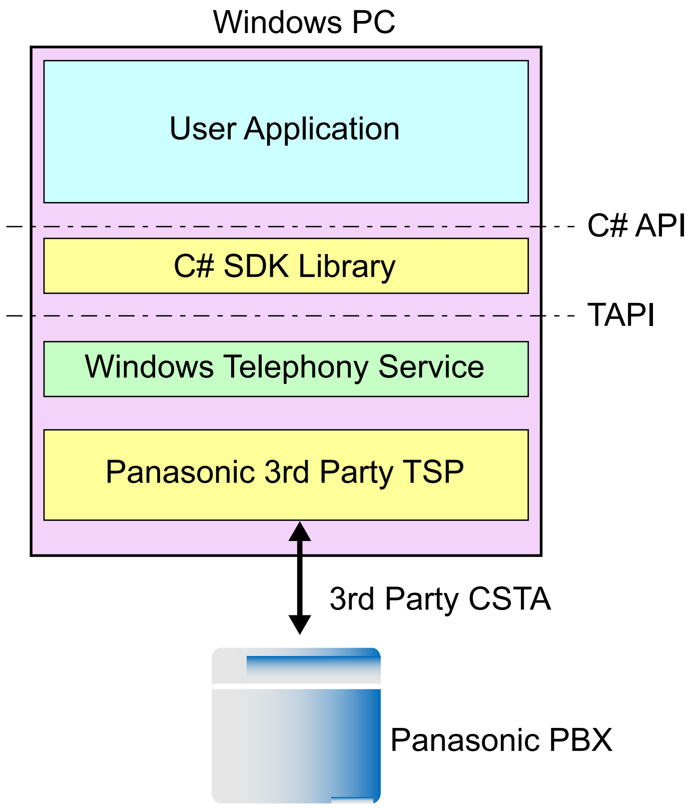
3 C# SDK Architecture

C# SDK is a DLL composed by a C# class library and is a wrapper class located on TAPI. C# SDK provides a simple and easy-to-use programming interface.
To use C# SDK Library, Panasonic TAPI Service Provider (Panasonic 3rd Party TSP) is required. However tapi.h is not required to compile an application with C# SDK Library.

IMPORTANT

C# SDK library assumes that no other TAPI application is running on the same PC. This is to ensure that C# SDK library have full control over the PBX.

The following figure shows the architecture of a C# SDK application.



4 C# SDK Library Members

C# SDK library has the following members.

4.1 Methods

Name	Description
5.1 Initialize	Initializes C# SDK library at startup.
5.2 Shutdown	Shuts down C# SDK library.
5.3 GetLineInfoList	Gets list of line information.
5.4 GetPBXInfo	Gets PBX information.
5.5 LineOpen	Opens a specified line.
5.6 LineClose	Closes a specified line.
5.7 AnswerCall	Answers a specified offering call.
5.8 DropCall	Drops or disconnects a specified call.
5.9 MakeCall	Places a call on a specified line to a specified destination address.
5.10 SetSMDROn	Enables call accounting information.
5.11 SetSMDROff	Disables call accounting information.
5.12 GetSMDR	Gets a call accounting record.
5.13 SetTwoWayRecording	Executes two-way recording.
5.14 ButtonPress	Simulates activation of a specified button at a specified proprietary telephone.
5.15 SetDisplay	Sets a specified string to a proprietary telephone display.
5.16 ClearDisplay	Clears proprietary telephone display.
5.17 TerminateDisplay	Terminates LCD display.
5.18 SetRingerStatus	Rings a specified extension’s telephone.
5.19 GetLineStatus	Gets current line status for a specified line.
5.20 GetCallInfo	Gets call information for a specified call.
5.21 GetLineInfo	Gets line information for a specified line.

4.2 Event & Type of Messages

4.2.1 Event

Name	Description
6 C# SDK Event and Type of Messages	Event published by C# SDK library to notify user application that is interested in C# SDK messages. User application has to cast the result according to type of message. Type of message that will send by C# SDK library are listed in **4.2.2 Types of Messages**.

4.2.2 Types of Messages

Name	Description
6.1 KX_APPNEWCALL	Sent to inform an application when a new call has been created.
6.2 KX_CALLSTATE	Sent when the status of the specified call has changed.
6.3 KX_CALLINFO	Sent when the call information about the specified call has changed.
6.4 KX_DEVSTATE	Sent when the state of a line device has changed.
6.5 KX_REINIT	Sent to C# SDK application if any connection error between PBX and TSP has occurred.
6.6 KX_RETRIEVE	Sent to C# SDK application if the connection between PBX and TSP has been recovered by connection recovery process of TSP after a connection error occurred.
6.7 KX_SMDR_CREATE	Sent to report that call accounting information is added into account information buffer.
6.8 KX_SMDR_DELETE	Sent to report that call accounting information is deleted from account information buffer.
6.9 KX_RECONFIG	Sent whenever the PBX setting is changed.
6.10 KX_INIT_ERROR	Sent to C# SDK application if initialization fails.
6.11 KX_LICENSE_ERROR	Sent to report activation key validation failed.

4.3 Data Structures

Name	Description
7.1 V_LINEINFO	Describes the line information.
7.2 V_PBXINFO	Describes the PBX information.
7.3 V_CALLACCOUNTINFO	Describes the call account information.
7.4 V_CALLINFO	Contains the call data.
7.5 V_EVENTMESSAGEINFO	Contains the message information.
7.6 V_EVENTCALLINFO	Contains the call information for KX_CALLINFO event.

4.4 Enumerations

Name	Description
8.1 E_LINETYPE	Represents the line type.
8.2 E_LINESTATUS	Represents the current line status.
8.3 E_EXTTYPE	Represents the extension type.
8.4 E_PBXTYPE	Represents the PBX type.
8.5 E_RINGMODE	Represents the ring mode.
8.6 E_EVENTTYPE	Represents the event type for C# SDK events.
8.7 E_KXERROR	Represents the exception.
8.8 E_CALLKIND	Represents the kind of call.
8.9 E_CALLCONDITION	Represents the call condition.
8.10 E_LINECALLINFOSTATE	Represents the call information items.
8.11 E_LINECALLSTATE	Represents the state of a call.
8.12 E_LINECALLORIGIN	Represents the origin of a call.
8.13 E_LINECALLREASON	Represents the reason of a call.
8.14 E_LINEDEVSTATE	Represents the state of a line device.

5 C# SDK Methods

5.1 Initialize

Outline	Initializes the C# SDK library.		
Call form	void Initialize()		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
None			

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_NOLINE
- KX_NOPROVIDER
- KX_FAILEDREINIT
- KX_REQUESTTIMEOUT
- KX_REINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

Initialize is mandatory to start C# SDK library. TSP configuration is needed before calling Initialize. C# SDK application needs to subscribe KXMessageEvent before calling Initialize.

During initialize, C# SDK library will perform activation key for Third Party CTI (KX-NCS3930) validation. In case of activation key validation fails, C# SDK library sends KX_LICENSE_ERROR event to user. KX_LICENSE_ERROR event contains PBX information for the PBX that does not have an activation key for Third Party CTI. In case of multi PBXs connection, C# SDK library will continue with the next PBX in sequence. Each PBX has its own SYSTEM line as the first line in the PBX. Other types of lines follow the SYSTEM line.

C# SDK library will automatically call LineOpen for SYSTEM and TR line for PBX that have an activation key for Third Party CTI during initialization. KX_INIT_ERROR event with provider name of PBX is sent in case C# SDK library fails to open SYSTEM line for the specific PBX. In case of multi PBXs connection, C# SDK library will continue with the next PBX in sequence.

In case all connected PBXs are returning errors, then C# SDK library throws KX_NOLINE error to user. In case no Panasonic 3rd Party TSP is found, KX_NOPROVIDER is thrown to the user. If at least one PBX is successfully connected and used by C# SDK library, user can still proceed by using C# SDK library; no exception is received.

5.2 Shutdown

Outline	Shutdowns C# SDK library.		
Call form	void Shutdown()		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
None			

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

If the application is not monitoring and not using any devices, it should call Shutdown so that memory resources allocated can be released if needed.

If this method is called when the application has lines open or calls active, the call resources are deleted and C# SDK library automatically performs the equivalent of a LineClose on each opened line.

5.3 GetLineInfoList

Outline	Gets list of line information.		
Call form	List<V_LINEINFO> GetLineInfoList()		
Return value	Type	Description	Remarks
	List <V_LINEINFO>	List of line information for all lines	For details of the V_LINEINFO structure, please refer to **7.1 V_LINEINFO**.
Parameter	Type	Description	Remarks
None			

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method is called after initialization to obtain list of available lines and its information.
Line ID in V_LINEINFO structure is used in subsequent operation on the line device.

5.4 GetPBXInfo

Outline	Gets PBX information.		
Call form	V_PBXINFO GetPBXInfo(uint lineID)		
Return value	Type	Description	Remarks
	V_PBXINFO	PBX Information	Refer to **7.2 V_PBXINFO**.
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_SDKNOTINIT
- KX_INVALLINEID

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method is called to obtain PBX information of the specific line.

5.5 LineOpen

Outline	Opens a specified line device for providing subsequent monitoring and/or control of the line.		
Call form	void LineOpen(uint lineID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID for the specified line to open	Provided by GetLineInfoList method

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_REINIT
- KX_NODRIVER
- KX_NODEVICE
- KX_SDKNOTINIT
- KX_INUSE

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

LineOpen will register the application as having an interest in monitoring calls or receiving ownership of calls that are of the specified media types.

No error is returned if other applications are running. Call control features might be limited in this case.

Opening a line entitles the application to perform subsequent operation on the line device.

5.6 LineClose

Outline	Closes a specified opened line.		
Call form	void LineClose(uint lineID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID to the opened line to be closed	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

If an application calls LineClose while it still has active calls on the opened line, the application's ownership of these calls is revoked.
C# SDK library does not drop active calls when the line is closed.

5.7 AnswerCall

Outline	Answers a specified offering call.		
Call form	void AnswerCall(uint callID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
callID	uint	Call ID to the call to be answered	– Provided by KX_CALLSTATE message. – The call state of callID must be offering or accepted.

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALCALLSTATE
- KX_INVALCALLID
- KX_NOTOWNER
- KX_SDKNOTINIT
- KX_REQUESTTIMEOUT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

When a new call arrives, applications with an interest in the call are sent a KX_CALLSTATE message to provide the new call ID and to inform the application about the call's state. After the call has been successfully answered, the call typically transitions to the connected state. If other applications are running, this method may fail.

5.8 DropCall

Outline	Drops or disconnects a specified call.		
Call form	void DropCall(uint callID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
callID	uint	Call ID for the specified call to be dropped	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALCALLSTATE
- KX_INVALCALLID
- KX_NOTOWNER
- KX_SDKNOTINIT
- KX_REQUESTTIMEOUT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

When invoking DropCall, related calls can sometimes be affected as well. For example, dropping a conference call can drop all individual participating calls.
KX_CALLSTATE messages are sent to the application for all calls whose call state is affected. A dropped call typically transitions to the idle state. Invoking DropCall on a call in the offering state is not supported.

PBX Behavior

Dropping the conference owner will drop all the parties in the conference. Dropping a party from a 3 party conference will result the moving of the conference into a normal 2 party call.

5.9 MakeCall

Outline	Places a call on a specified line to a specified destination address.		
Call form	uint MakeCall(uint lineID, string destAddress)		
Return value	Type	Description	Remarks
	uint	callID	This callID is void if method throws an exception.
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line on which a call is to be originated	
destAddress	string	The destination to be dialed using the standard dialable number format	This follows the standard dialable numbers format same as TAPI.

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_CALLUNAVAIL
- KX_INUSE
- KX_SDKNOTINIT
- KX_REQUESTTIMEOUT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

After dialing has completed, several KX_CALLSTATE messages are usually sent to the application to notify it about the progress of the call. No generally valid sequence of call-state transitions is specified, as no single fixed sequence of transitions can be guaranteed in practice. A typical sequence can cause a call to transition from dial tone, dialing, proceeding, ring back, to connected. With non-dialed lines, the call can typically transition directly to connected state.

An application can call MakeCall without specifying the DestAddress, this will create a call with KX_DIALTONE state. If the application continues call MakeCall a second time with a specific DestAddress, dialing on the existing call appearance.
MakeCall is not affected by existence of other TAPI application.

5.10 SetSMDROn

Outline	Enables call accounting information.		
Call form	void SetSMDROn(uint systemID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
systemID	uint	Line ID for specific SYSTEM Line	Provided by GetLineInfoList method

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

For this function to take effect, SMDR must be turned on at the PBX. After SetSMDROn, KX_SMDR_CREATE message is generated for SYSTEM line. If account information queue buffer is full, then KX_SMDR_DELETE message will be generated as well.

Panasonic 3rd Party TSP Note

After starting SMDR On, PBX starts to send stored account information data to TSP.
When Panasonic 3rd Party TSP receives new account information data, a new account information record is created and stored in the account information queue buffer.

5.11 SetSMDROff

Outline	Disables call accounting information.		
Call form	void SetSMDROff(uint systemID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
systemID	uint	Line ID for specific SYSTEM Line	Provided by GetLineInfoList method

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

After SetSMDROff, C# SDK application will not receive any call accounting information messages.

Panasonic 3rd Party TSP Note

After SMDR off, PBX stops sending stored account information data to TSP.

5.12 GetSMDR

Outline	Gets call accounting information.		
Call form	V_CALLACCOUNTINFO GetSMDR(uint systemID, uint accountID)		
Return value	Type	Description	Remarks
	V_CALLACCOUNTINFO	Call accounting information	Refer to **7.3 V_CALLACCOUNTINFO**.
Parameter	Type	Description	Remarks
systemID	uint	Line ID for specific SYSTEM Line	Provided by GetLineInfoList method
accountID	uint	Call accounting information ID for a record of call accounting information to be obtained	Provided by KX_SMDR_CREATE message

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

To enable this function:

- SMDR must be turned on at the PBX.
- SetSMDROn must be called on specified SYSTEM line.

Call accounting information for all lines can be obtained on SYSTEM Line. Content of call accounting information is built based on SMDR in the PBX. After SetSMDROn is called, Panasonic 3rd Party TSP starts generating call accounting information.

The call accounting information is generated when;

- SetSMDROn is called and if there is already stored information inside the PBX.
- A call is dropped. This generating trigger is the same as SMDR.

Panasonic 3rd Party TSP Note

Call accounting information is stored in a queue buffer of the given size. The size can be changed by Panasonic 3rd Party TSP Configuration dialog. Available range of buffer size is from 100 to 10000 (Default is 500). In case the number of stored records reaches the configured buffer size, the oldest record is overwritten by cyclic operating.

Each call account ID has a unique value. When the value comes to be 0xffffffff, it returns to 0. A record of call account information can be obtained by specifying the call account information ID.

5.13 SetTwoWayRecording

Outline	Uses to execute Two-Way Recording. The message is stored in the requesting device’s mailbox.		
Call form	void SetTwoWayRecording(uint lineID, uint callID, string vm_groupNo)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line on which a call is to be recorded	DPT type extension line only
callID	uint	Call ID for the specified call to record	The call state of callID must be connected.
vm_groupNo	string	Voice Mail Group number	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_INVALCALLID
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method is not available for SIP extension line. Voice mail setting is needed for using this method.

5.14 ButtonPress

Outline	Simulates the activation of a specified button at a specified proprietary telephone.		
Call form	void ButtonPress(uint lineID, uint buttonID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line	DPT type extension line only
buttonID	uint	Button ID to be pressed	Refer to KXTDA_Tapi_Specification document - Annex D Button Map.

Exceptions (if an error occurs)

Possible exceptions are:
- KX_FAILED
- KX_INVALLINEID
- KX_REQUESTTIMEOUT
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method is available only for DPT extension line because of PBX limitation. In old MPR software, software upgrade is needed to use this feature. Software upgrade is not required by KX-NCP Series PBXs (PBMPR Software File Version 2.0000 or later) and KX-TDE Series PBXs (PMMPR/PGMPR Software File Version 3.0000 or later).

5.15 SetDisplay

Outline	Sets specified string to a proprietary telephone display.		
Call form	void SetDisplay(uint lineID, uint row, uint column, string text)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line	For PT(DPT and IP-PT), and PS extension line only
row	uint	The row position on the display where the new text is to be displayed	Available rows is obtained from GetLineInfoList.
column	uint	The column position on the display where the new text is to be displayed	Available columns is obtained from GetLineInfoList.
text	string	Content to be displayed	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_NOTOWNER
- KX_REQUESTTIMEOUT
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

The specified display information is written to the phone's display, starting at the specified positions. This operation overwrites previously displayed information.

If the amount of information exceeds the size of the display, the information is truncated. The amount of information that can be displayed is at most (Rows * Columns) elements in size. Available rows and columns on LCD depend on type of PT or PS. Rows and columns are available in the V_LINEINFO structure, which is returned by GetLineInfoList.

PS requires an activation key for software upgrade in order to use this function. Call state of a PS must be idle to set display. Otherwise, KX_FAILED is returned.

5.16 ClearDisplay

Outline	Clears proprietary telephone display.		
Call form	void ClearDisplay(uint lineID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line	For PT(DPT and IP-PT), and PS only

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_NOTOWNER
- KX_REQUESTTIMEOUT
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method clears all display information written on the phone's display.
PS requires an activation key for software upgrade in order to use this function.

5.17 TerminateDisplay

Outline	Terminates LCD display control.		
Call form	void TerminateDisplay(uint lineID)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line	For PT(DPT and IP-PT), and PS only

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_NOTOWNER
- KX_REQUESTTIMEOUT
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method unlocks the phone's display.
PS requires an activation key for software upgrade in order to use this function.

5.18 SetRingerStatus

Outline	Rings the specified extension's telephone using the specified ring mode.		
Call form	void SetRingerStatus(uint lineID, E_RINGMODE ringMode)		
Return value	Type	Description	Remarks
	None		
Parameter	Type	Description	Remarks
lineID	uint	Line ID of an opened line	For extension line only
ringMode	E_RINGMODE	0: KX_SILENCE 1: KX_SINGLE 2: KX_DOUBLE 3: KX_TRIPLE 4: KX_SDOUBLE 5: KX_OPTION1 6: KX_OPTION2	Refer to **8.5 E_RINGMODE**.

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_NOTOWNER
- KX_REQUESTTIMEOUT
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

SetRingerStatus will not set the ring mode permanently; it will only ring the specified extension telephone with the selected ringMode.
PS requires an activation key for software upgrade in order to use this function.

5.19 GetLineStatus

Outline	Gets current line status for specified line.		
Call form	E_LINESTATUS GetLineStatus(uint lineID)		
Return value	Type	Description	Remarks
	E_LINESTATUS	0: UNKNOWN 1: INS 2: OUS	UNKNOWN when line is not opened. Refer to **8.2 E_LINESTATUS**.
Parameter	Type	Description	Remarks
lineID	uint	Line ID of a line	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALLINEID
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method returns the current line status (INS / OUS) for opened lines. If a line is not opened, UNKNOWN is returned.

5.20 GetCallInfo

Outline	Gets call information for specified existing call.		
Call form	V_CALLINFO GetCallInfo(uint callID)		
Return value	Type	Description	Remarks
	V_CALLINFO		Refer to **7.4 V_CALLINFO**.
Parameter	Type	Description	Remarks
callID	uint	Call ID to the call	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_INVALCALLID
- KX_SDKNOTINIT

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method enables an application to obtain fixed information about the specified call.

A separate V_CALLINFO structure exists for every incoming or outgoing call. The structure contains primarily fixed information about the call.

An application would typically be interested in checking this information when it receives its handle for a call by the KX_CALLSTATE message, or each time it receives notification by a KX_CALLINFO message that parts of the call information structure have changed. These messages supply the ID for the call as a parameter.

5.21 GetLineInfo

Outline	Gets line information for specified line.		
Call form	V_LINEINFO GetLineInfo(uint lineID)		
Return value	Type	Description	Remarks
	V_LINEINFO	Line information for a specified line	For details of the V_LINEINFO structure, please refer to **7.1 V_LINEINFO**.
Parameter	Type	Description	Remarks
lineID	uint	Line ID of a line	

Exceptions (if an error occurs)

Possible exceptions are:

- KX_FAILED
- KX_SDKNOTINIT
- KX_INVALLINEID

For details of each exception, please refer to **8.7 E_KXERROR**.

Description

This method is called to update line information for a specified line only.

6 C# SDK Event and Type of Messages

This section contains events published and types of messages in the C# SDK library.
The event always contains an ID to the relevant object (line, or call). The application can determine the type of the ID from the message type.

User application that are interested in C# SDK events need to subscribe KXMessageEvent:

Outline	Event published by C# SDK library to notify user application that are interested in C# SDK messages.
Call form	event KXMessage KXMessageEvent

6.1 KX_APPNEWCALL

Outline	Sent to inform an application when a new call ID has been spontaneously created on its behalf (other than through an API call from the application, in which case the ID would have been returned through a parameter passed in the function).		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID on which the call has been created
e_EventType	KX_APPNEWCALL	E_EVENTTYPE	New call created event
objResult	callID	uint	Call ID to a new call

Description

The KX_APPNEWCALL message is always immediately followed by a KX_CALLSTATE message indicating the initial state of the call.

6.2 KX_CALLSTATE

Outline	Sent when the status of the specified call has changed. Typically, several such messages are received during the lifetime of a call.		
Message structure	Parameter	Type	Description
uiMsgID	callID	uint	Call ID
e_EventType	KX_CALLSTATE	E_EVENTTYPE	Call state changed event
objResult	callState	E_LINECALLSTATE	Call state

Description

When the CallState is KX_IDLE, C# SDK library will call TAPI lineDeallocateCall to release internal resources related to the call.

Panasonic 3rd Party TSP Note

The KX_CALLSTATE message sent to notify applications that monitor calls on a line about the existence and state of outbound calls established by other applications or manually by the user (for example, on an attached phone device). The call state of such calls reflects the actual state of the call, which is not offering. When the KX_CALLSTATE is offering means the call is offered to the station and signaling the arrival of a new call. The offering state is not the same as causing a phone or computer to ring. By examining the call state, the application can determine whether the call is an inbound call that needs to be answered or not.

6.3 KX_CALLINFO

Outline	Sent when the call information about the specified call has changed. This message will provide current call information of the specified call.		
Message structure	Parameter	Type	Description
uiMsgID	callID	uint	Call ID for specified call whose information has changed
e_EventType	KX_CALLINFO	E_EVENTTYPE	Call information changed event
objResult	eventCallInfo	V_EVENTCALLINFO	Refer to **7.6 V_EVENTCALLINFO**.

Description

A KX_CALLINFO message with a KX_NUMOWNERINCR, KX_NUMOWNERDECR, and/or KX_NUMMONITORS indication is sent to applications that already monitor the call. This can be the result of another application changing ownership or monitorship to a call with LineOpen, LineClose, Shutdown and some other TAPI APIs.

These KX_CALLINFO messages are not sent when a notification of a new call is provided in a KX_CALLSTATE message, because the call information already reflects the correct number of owners and monitors at the time the KX_CALLSTATE messages are sent. KX_CALLINFO messages are also suppressed in the case where a call is offered by TAPI to monitors through the KX_UNKNOWN mechanism.

No KX_CALLINFO messages are sent for a call after the call has entered the idle state. Specifically, changes in the number of owners and monitors are not reported as applications deallocate their handles for the idle call.

6.4 KX_DEVSTATE

Outline	Sent when the state of a line device has changed.		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID of which the state has changed. This parameter is 0 when DeviceState is KX_REINIT.
e_EventType	KX_DEVSTATE	E_EVENTTYPE	Line device state changed event
objResult	deviceState	E_LINEEDEVSTATE	Refer to **8.14 E_LINEDEVSTATE**.

Description

If any connection error between PBX and TSP has occurred and TSP failed to recover the connection, KX_DEVSTATE - KX_REINIT is sent to the application. The lineID is 0 for this state change as it applies to any of the lines in the system. In this case, Windows Telephony Service should be restarted to recover the connection. This normally requires Windows reboot.

6.5 KX_REINIT

Outline	Sent to C# SDK application if any connection error between PBX and TSP has occurred.		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID of opened line
e_EventType	KX_REINIT	E_EVENTTYPE	Connection error occurred event
objResult	None		

Description

This message is sent if any connection error between PBX and TSP has occurred. In such case KX_DEVSTATE - KX_OUTSERVICE message is sent as well as this message.
After this message, TSP tries to recover the connection several times.

- If it succeeds, KX_RETRIEVE message and KX_DEVSTATE - KX_INSERVICE are sent to the application.
- If it fails as a result of retry, KX_DEVSTATE - KX_REINIT message is sent to C# SDK application. In this case Windows Telephony Service should be restarted to recover the connection. This normally requires Windows reboot.

Panasonic 3rd Party TSP Note

This message is sent for all line devices serviced by Panasonic 3rd Party TSP.
This message is generated only when “Send KX_REINIT and KX_RETRIEVE” check box in configuration GUI is checked and TSP works in Standard Mode.

6.6 KX_RETRIEVE

Outline	Sent to C# SDK application if the connection between PBX and TSP has been recovered by connection recovery process of TSP after a connection error occurred.		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID of opened line
e_EventType	KX_RETRIEVE	E_EVENTTYPE	Connection retrieved event
objResult	None		

Description

This message is sent to C# SDK application if the connection between PBX and TSP has been recovered by connection recovery process of TSP after a connection error occurred. In such cases a KX_DEVSTATE − KX_INSERVICE message is sent as well as this message.

Panasonic 3rd Party TSP Note

This message is sent for all line devices serviced by Panasonic 3rd Party TSP.
This message is generated only when “Send KX_REINIT and KX_RETRIEVE” check box in configuration GUI is checked and TSP works in Standard Mode.

6.7 KX_SMDR_CREATE

Outline	Sent to report that call accounting information is added into call account information buffer.		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID for SYSTEM line
e_EventType	KX_SMDR_CREATE	E_EVENTTYPE	Call accounting information created event
objResult	accountID	uint	Call account Information ID that had been added

Description

Before this message is generated call account information record is changed into a call account info buffer, therefore after this message is generated it is possible to obtain a call account information structure for call account information ID specified as AccountID parameter of the message by using GetSMDR.

6.8 KX_SMDR_DELETE

Outline	This message is sent to reports that call accounting info is deleted from call account information buffer.		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID for SYSTEM line
e_EventType	KX_SMDR_DELETE	E_EVENTTYPE	Call accounting information deleted event
objResult	accountID	uint	Call account Information ID that had been deleted

Description

Before this message is generated call account information record is deleted from call account information buffer, therefore after this message is generated it is not possible to obtain a call account information structure for call account information ID specified as AccountID parameter of the message by using GetSMDR.

6.9 KX_RECONFIG

Outline	This message is sent to C# SDK application if PBX setting is changed.		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID for SYSTEM line
e_EventType	KX_RECONFIG	E_EVENTTYPE	PBX setting changed event
objResult	None		

Remarks

This message is supported only in Standard Mode.
This message may be sent if PC Maintenance Console is exited with some system data change.

6.10 KX_INIT_ERROR

Outline	This message is sent to C# SDK application at initialization error event.		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID for SYSTEM line
e_EventType	KX_INIT_ERROR	E_EVENTTYPE	Initialization Error Event
objResult	providerName	string	TSP Name

Remarks

This message is sent when any SYSTEM line failure is found during initialization. The TSP name of the SYSTEM line is provided in this message.

6.11 KX_LICENSE_ERROR

Outline	This message is sent to C# SDK application if PBX does not contain activation key for Third Party CTI (KX-NCS3930).		
Message structure	Parameter	Type	Description
uiMsgID	lineID	uint	Line ID for SYSTEM line
e_EventType	KX_LICENSE_ERROR	E_EVENTTYPE	License validation failed event
objResult	PBXInfo	V_PBXINFO	PBX information

Remarks

This message sent when activation key validation failed for the specific PBX. PBX information for the PBX which failed is sent along with this message.

7 C# SDK Data Structures

7.1 V_LINEINFO

Outline	Line information		
Related section	**5.3 GetLineInfoList**		
Members	Type	Description	Remarks
uiLineID	uint	Line ID	
strLineName	string	Line Name	
e_LineType	E_LINETYPE	Line Type	Refer to **8.1 E_LINETYPE**.
e_LineStatus	E_LINESTATUS	Line Status	Can be obtained for opened line only. Refer to **8.2 E_LINESTATUS**.
strExtNumber	string	Extension Number	
strExtName	string	Extension Name	Can be obtained for opened line only.
e_ExtType	E_EXTTYPE	Extension Type	Refer to **8.3 E_EXTTYPE**.
uiLCDRows	uint	Number of rows in the phone display	
uiLCDColumns	uint	Number of columns in the phone display	

Remarks

Member strExtName is available for following line type only.

- KX_EXT
- KX_CO
- KX_SVM
- KX_SENSOR
- KX_GROUP

For line type other than mentioned above, strExtName in V_LINEINFO is empty.

Line status of a line is obtainable for opened line only. For un-opened line, e_LineStatus is KX_UNKNOWN. Line status for portable station is always KX_INS, once extension number is assigned.

Extension type is obtainable when e_LineStatus is KX_INS. Extension type for Cell station is always KX_UNKNOWN regardless of INS/OUS status.

7.2 V_PBXINFO

Outline	PBX information		
Related section	**5.4 GetPBXInfo**		
Members	Type	Description	Remarks
e_PBXType	E_PBXTYPE	PBX Type	Refer to **8.4 E_PBXTYPE**.
strName	string	PBX Name	
strMPRVersion	string	MPR Version	
strCTIVersion	string	CTI Version	
strIPAddress	string	PBX IP Address	

7.3 V_CALLACCOUNTINFO

Outline	Call Accounting Information		
Related section	**5.12 GetSMDR**		
Members	Type	Description	Remarks
e_CallKind	E_CALLKIND	Kind of call	Refer to **8.8 E_CALLKIND**.
e_CallCondition	E_CALLCONDITION	Call condition	Refer to **8.9 E_CALLCONDITION**.
dtDateTime	DateTime	Date and time	
iRingingTime	int	Ringing time	
iDuration	int	Duration time	
strOriNumber	string	Original number	
strDestNumber	string	Destination number	
strDialledNumber	string	Dialed number	
strCallerID	string	Caller ID	
strCallerIDName	string	Caller name	
strAccountCode	string	Account code	
strCurrency	string	Currency	
strTelephoneCharge	string	Telephone charge	

7.4 V_CALLINFO

Outline	Call information		
Related section	**5.20 GetCallInfo**		
Members	Type	Description	Remarks
uiLineID	uint	Line ID	
e_CallState	E_LINECALLSTATE	Call state	Refer to **8.11 E_LINECALLSTATE**.
e_Origin	E_LINECALLORIGIN	Identifies where the call originated.	Refer to **8.12 E_LINECALLORIGIN**[*1].
e_Reason	E_LINECALLREASON	Reason why the call occurred	Refer to **8.13 E_LINECALLREASON**[*1].
uiTrunk	uint	Number of trunk over which the call is routed. 0xFFFFFFFF if it is unknown.	
strCallerID	string	Caller party ID number information	[*2]
strCallerIDName	string	Caller party name information	[*2]
strCalledID	string	Called party ID number information	[*2]
strCalledIDName	string	Called party name information	[*2]
strConnectedID	string	Connected party ID number information. Contains actual connected address.	[*2]
strConnectedIDName	string	Connected party name information	[*2]
strRedirectionID	string	Redirection party ID number information	[*2]
strRedirectionIDName	string	Redirection party name information	[*2]
strRedirectingID	string	Redirecting party ID number information	[*2]
strRedirectingIDName	string	Redirecting party name information	[*2]

[*1] KX_UNKNOWN if specified call exists before LineOpen and PBX connection recovery process.

[*2] Empty if specified call exists before LineOpen and PBX connection recovery process.

7.5 V_EVENTMESSAGEINFO

Outline	C# SDK library message structure		
Related section	**6 C# SDK Event and Type of Messages**		
Members	Type	Description	Remarks
uiMsgID	uint	Message ID	
e_EventType	E_EVENTTYPE	Event Type	Refer to **8.6 E_EVENTTYPE**.
objResult	object	Result structure. Data type depending on e_EventType.	

7.6 V_EVENTCALLINFO

Outline	Call information for KX_CALLINFO event		
Related section	**6.3 KX_CALLINFO**		
Members	Type	Description	Remarks
e_CallInfoState	E_LINECALLINFOSTATE	Call information items that have changed	Refer to **8.10 E_LINECALLINFOSTATE**.
v_CallInfoDetail	V_CALLINFO	Call detailed information	Refer to **7.4 V_CALLINFO**.

8 C# SDK Enumerations

8.1 E_LINETYPE

Outline	Represents the line type.	
Related section	**7.1 V_LINEINFO**	
Members	Value (int)	Description
KX_DOOR	1	Door phone
KX_DISA	2	DISA
KX_PGG	3	Paging group
KX_VU	4	Voice mail unit
KX_GROUP	5	Group
KX_PSG	6	PS group
KX_VMG	7	VM group
KX_PARK	8	Park device
KX_EXPG	9	External pager
KX_CO	10	Central office (CO)
KX_EXT	11	Extension
KX_SENSOR	12	Sensor
KX_SYSTEM	13	System
KX_VSYSTEM	14	Virtual system (TR)
KX_MODEM	15	Modem
KX_HDLC	16	HDLC
KX_SVM	17	SVM

8.2 E_LINESTATUS

Outline	Represents the current line status.	
Related section	**7.1 V_LINEINFO**	
Members	Value (int)	Description
KX_UNKNOWN	0	Status unknown when line is not opened.
KX_INS	1	In Service
KX_OUS	2	Out Service

8.3 E_EXTTYPE

Outline	Represents the extension type.	
Related section	**7.1 V_LINEINFO**	
Members	Value (int)	Description
KX_SLT	0	SLT
KX_APT	1	APT
KX_DPT	3	DPT
KX_WIRELESS	4	Wireless
KX_ISDN	5	ISDN
KX_IPPT_NONP2P	12	IP-PT (non P2P)
KX_IPPT_P2P	13	IP-PT (P2P)
KX_SIP	14	SIP
KX_NONE	100	Not extension type
KX_UNKNOWN	101	Extension type is unknown if the extension is out of service.

Remarks

Extension type for Cell station is always KX_UNKNOWN regardless of INS/OUS status.

8.4 E_PBXTYPE

Outline	Represents the PBX type.	
Related section	**7.2 V_PBXINFO**	
Members	Value (int)	Description
Unknown	0	Unknown type
TDE100	8	KX-TDE100
TDE200	9	KX-TDE200
TDE600	10	KX-TDE600
NCP500	11	KX-NCP500
NCP1000	12	KX-NCP1000

8.5 E_RINGMODE

Outline	Represents the ring mode.	
Related section	**5.18 SetRingerStatus**	
Members	Value (uint)	Description
KX_SILENCE	0	Silence
KX_SINGLE	1	Single
KX_DOUBLE	2	Double
KX_TRIPLE	3	Triple
KX_SDOUBLE	4	S-Double
KX_OPTION1	5	Option 1
KX_OPTION2	6	Option 2

8.6 E_EVENTTYPE

Outline	Represents the event type for C# SDK events.	
Related section	**6 C# SDK Event and Type of Messages**	
Members	Value (uint)	Description
KX_APPNEWCALL	1	New call event
KX_CALLSTATE	2	Call state changed event
KX_CALLINFO	3	Call information changed event
KX_DEVSTATE	4	Device state changed event
KX_SMDR_CREATE	5	SMDR created event
KX_SMDR_DELETE	6	SMDR deleted event
KX_REINIT	7	Connection error event
KX_RETRIEVE	8	Connection retrieved event
KX_RECONFIG	9	PBX system reconfiguration event
KX_INIT_ERROR	10	C# SDK library initialization failed event
KX_LICENSE_ERROR	11	Activation key for Third Party CTI validation failed event

8.7 E_KXERROR

Outline	Represents the exception.	
Related section	All C# SDK Methods	
Members	Value (uint)	Description
KX_OK	0	The operation was invoked successfully.
KX_FAILED	1	The operation failed.
KX_REINIT	2	TAPI re-initialization has been requested.
KX_NODRIVER	3	Either Tapiaddr.dll could not be located or the telephone service provider for the specified device found that one of its components is missing or corrupt in a way that was not detected at initialization time. The user should be advised to use the Telephony Control Panel to correct the problem.
KX_NODEVICE	4	The specified device identifier, which was previously valid, is no longer accepted because the associated device has been removed from the system since TAPI was last initialized. Alternately, the line device has no associated device for the given device class.
KX_INVALLINEID	5	The specified line ID is invalid.
KX_INVALCALLSTATE	6	The current state of a call is not in a valid state for the requested operation.
KX_INVALCALLID	7	The specified call ID is not valid.
KX_NOTOWNER	8	The application is not the owner to the specified call.
KX_CALLUNAVAIL	9	All call appearances on the specified address are currently in use.
KX_INUSE	10	The line device is in use and cannot currently be configured, allow a party to be added, allow a call to be answered, allow a call to be placed, or allow a call to be transferred.
KX_NOLINE	11	No available line found.
KX_NOPROVIDER	12	No Panasonic 3rd Party TSP found.
KX_REQUESTTIMEOUT	13	C# SDK library failed to get reply message for the operation.
KX_SDKNOTINIT	14	The operation was invoked before any application initialized C# SDK library.
KX_FAILEDREINIT	15	User application attempted to initialize C# SDK library more than once.
KX_UNKNOWN	99	The operation failed for an unspecified or unknown reason.

8.8 E_CALLKIND

Outline	Represents the kind of call.	
Related section	**7.3 V_CALLACCOUNTINFO**	
Members	Value (int)	Description
KX_OUTGOING	1	Outgoing call
KX_INCOMING	2	Incoming call
KX_COtoCO	3	CO to CO call
KX_EXTtoEXT	4	Extension to extension call

8.9 E_CALLCONDITION

Outline	Represents the call condition.	
Related section	**7.3 V_CALLACCOUNTINFO**	
Members	Value (int)	Description
KX_TRANSFER	1	Transfer
KX_FORWARDtoCO	2	Forwarded to CO
KX_MODEM	3	Modem
KX_RECEIVEINCOMINGCALL	4	Receive incoming call
KX_ANSWERINCOMINGCALL	5	Answer incoming call
KX_NOANSWERINCOMINGCALL	6	No answer incoming call
KX_OUTGOINGCALLDISA	7	Outgoing call through DISA
KX_NODATA	100	No data

8.10 E_LINECALLINFOSTATE

Outline	Represents the call information elements that are meaningful for all calls on this address.	
Related section	**7.6 V_EVENTCALLINFO**	
Members	Value (int)	Description
KX_CALLID	0x00000040	The call ID field of the call-information record
KX_RELATEDCALLID	0x00000080	The related call ID field of the call information record
KX_ORIGIN	0x00000100	The origin field of the call information record
KX_REASON	0x00000200	The reason field of the call information record
KX_COMPLETIONID	0x00000400	The completion ID field of the call information record
KX_NUMOWNERINCR	0x00000800	The number of owner field in the call information record was increased.
KX_NUMOWNERDECR	0x00001000	The number of owner field in the call information record was decreased.
KX_NUMMONITORS	0x00002000	The number of monitor field in the call information record has changed.
KX_TRUNK	0x00004000	The trunk field of the call information record has changed.
KX_CALLERID	0x00008000	One of the callerID-related fields of the call information record has changed.
KX_CALLEDID	0x00010000	One of the calledID-related fields of the call information record has changed.
KX_CONNECTEDID	0x00020000	One of the connectedID-related fields of the call information record has changed.
KX_REDIRECTIONID	0x00040000	One of the redirectionID-related fields of the call information record has changed.
KX_REDIRECTINGID	0x00080000	One of the redirectingID-related fields of the call information record has changed.

8.11 E_LINECALLSTATE

Outline	Represents the various call states that can possibly be reported for calls on this address.	
Related section	**6.2 KX_CALLSTATE** **7.4 V_CALLINFO**	
Members	Value (int)	Description
KX_IDLE	0x00000001	The call is idle−no call exists.
KX_OFFERING	0x00000002	The call is being offered to the station, signalling the arrival of a new call. In some environments, a call in the Offering state does not automatically alert the user; alerting is done by the switch instructing the line to ring. It does not affect any call states.
KX_ACCEPTED	0x00000004	The call was offering and has been accepted.
KX_DIALTONE	0x00000008	The call is receiving a Dial tone from the switch, which means that the switch is ready to receive a dialed number.
KX_DIALING	0x00000010	Destination address information (a phone number) is being sent to the switch on the call. Note that the function lineGenerateDigits does not place the line into the dialing state.
KX_RINGBACK	0x00000020	The call is receiving ring back from the called address. Ring back indicates that the other station has been reached and is being alerted.
KX_BUSY	0x00000040	The call is receiving a busy tone. Busy tone indicates that the call cannot be completed because either a circuit (trunk) or the remote party's station is in use.
KX_SPECIALINFO	0x00000080	The call is receiving a special information signal that precedes a pre-recorded announcement indicating why a call cannot be completed.
KX_CONNECTED	0x00000100	The call has been established and the connection is made. Information is able to flow on the call between the originating address and the destination address.
KX_PROCEEDING	0x00000200	Dialing has completed and the call is proceeding through the switch or telephone network.
KX_ONHOLD	0x00000400	The call is on hold by the switch.
KX_CONFERENCED	0x00000800	The call is currently a member of a multiparty conference call.
KX_ONHOLDPENDCONF	0x00001000	The call is currently on hold while it is being added to a conference call.
KX_ONHOLDPENDTRANSFER	0x00002000	The call is currently on hold awaiting transfer to another number.
KX_DISCONNECTED	0x00004000	The remote party has disconnected from the call.

KX_UNKNOWN	0x00008000	The state of the call is not known. This can be due to limitations of the call progress detection TSP Implementation.

8.12 E_LINECALLORIGIN

Outline	Represents where the call originated.	
Related section	**7.4 V_CALLINFO**	
Members	Value (int)	Description
KX_OUTBOUND	0x00000001	The call is an outgoing call.
KX_INTERNAL	0x00000002	The call is incoming and originated internally (on the same PBX, for example).
KX_EXTERNAL	0x00000004	The call originated as an incoming call on an external line.
KX_UNKNOWN	0x00000010	The call origin is currently unknown but may become known later.
KX_UNAVAIL	0x00000020	The call origin is not available and will never become known for this call.
KX_CONFERENCE	0x00000040	The call handle is for a conference call, that is, the application's connection to the conference bridge in the switch.
KX_INBOUND	0x00000080	The call originated as an incoming call, but the service provider is unable to determine whether it came from another station on the same switch or from an external line.

8.13 E_LINECALLREASON

Outline	Represents the reason for the call.	
Related section	**7.4 V_CALLINFO**	
Members	Value (int)	Description
KX_DIRECT	0x00000001	A direct call
KX_FWDBUSY	0x00000002	The call was forwarded from anther extension that was busy at the time of the call.
KX_FWDNOANSWER	0x00000004	The call was forwarded from another extension that didn’t answer the call after some number of rings.
KX_FWDUNCOND	0x00000008	The call was forwarded unconditionally from another number.
KX_PICKUP	0x00000010	The call was picked up from another extension.
KX_UNPARK	0x00000020	The call was retrieved as a parked call.
KX_REDIRECT	0x00000040	The call was redirected to this station.
KX_CALLCOMPLETION	0x00000080	The call was the result of a call completion request.
KX_TRANSFER	0x00000100	The call has been transferred from another number. Party identifier information may indicate who the caller is and where the call was transferred from.
KX_REMINDER	0x00000200	The call is a reminder (or "recall") that the user has a call parked or on hold for a potentially long time.
KX_UNKNOWN	0x00000400	The call reason is unknown.
KX_UNAVAIL	0x00000800	The call reason is unavailable.
KX_INTRUDE	0x00001000	The call intruded onto the line, either by a call completion action invoked by another station or by operator action. Depending on switch implementation, the call may appear either in the connected state, or conferences with an existing active call on the line.
KX_PARKED	0x00002000	The call was parked on the address. Usually, it appears initially in the on hold state.
KX_CAMPEDON	0x00004000	The call was camped on the address.
KX_ROUTEREQUEST	0x00008000	The call appears on the address because the switch needs routing instructions from the application.

8.14 E_LINEDEVSTATE

Outline	Represents the line device state.	
Related section	**6.4 KX_DEVSTATE**	
Members	Value (int)	Description
KX_RINGING	0x00000002	The switch tells the line to alert the user.
KX_CONNECTED	0x00000004	The line has been disconnected and is now connected to TAPI.
KX_DISCONNECTED	0x00000008	The line has been connected and is now disconnected from TAPI.
KX_MSGWAITON	0x00000010	The message waiting indicator is turned on.
KX_MSGWAITOFF	0x00000020	The message waiting indicator is turned off.
KX_INSERVICE	0x00000040	The line is connected to TAPI. This happens when TAPI is first activated or when the line wire is physically plugged in and in-service at the switch while TAPI is active.
KX_OUTOFSERVICE	0x00000080	The line is out of service at the switch or physically disconnected.
KX_OPEN	0x00000200	The line has been opened by another application.
KX_CLOSE	0x00000400	The line has been closed by another application.
KX_NUMCOMPLETIONS	0x00001000	The number of outstanding call completions on the line device has changed.
KX_DEVSPECIFIC	0x00020000	The line’s device-specific information has changed.
KX_REINIT	0x00040000	Items have changed in the configuration of line devices. To become aware of these changes, for example to become aware of the appearance of new line devices, the application should reinitialize its use of TAPI.
KX_TRANSLATECHANGE	0x00400000	Indicates that, due to configuration changes made by the user or other circumstances, one or more of the members in the LINETRANSLATECAPS structure have changed.
KX_REMOVED	0x01000000	Indicates that the device is being removed from the system by the service provider most likely through user action, through a control panel or similar utility. Subsequent attempts to access the device prior to TAPI being reinitialized will result in KX_NODEVICE being returned to the application.

9 Appendix A: Function-Device Support Chart

Legend:
🟩 Supported
▨ Not Supported

	Initialize	Shutdown	GetLineInfoList	GetPBXInfo	LineOpen	LineClose	AnswerCall	DropCall	MakeCall	SetSMDROn	SetSMDROff	GetSMDR	SetTwoWayRecording	ButtonPress	SetDisplay	ClearDisplay	TerminateDisplay	SetRingerStatus	GetLineStatus	GetCallInfo	GetLineInfo
Ext (PT) **Ext (IP-PT: non P2P)** **Ext (IP-PT: P2P)**	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	🟩	🟩 *2	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Ext (SLT)	🟩	🟩	🟩	🟩	🟩	🟩	▨	🟩	🟩	▨	▨	▨	🟩	▨	▨	▨	▨	🟩	🟩	🟩	🟩
Ext (ISDN)	🟩	🟩	🟩	🟩	🟩	🟩	▨	🟩	▨	▨	▨	▨	🟩	▨	▨	▨	▨	🟩	🟩	🟩	🟩
Ext (SIP)	🟩	🟩	🟩	🟩	🟩	🟩	▨	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩	🟩
Ext (PS)	🟩	🟩	🟩	🟩	🟩	🟩	▨	🟩	🟩	▨	▨	▨	🟩	▨	🟩	🟩	🟩	🟩	🟩	🟩	🟩
SYSTEM	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	🟩	▨	🟩
Virtual System (TR)	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
DoorPhone	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
DISA	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
Paging Group	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
Voice Mail Unit	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
Group	🟩	🟩	🟩	🟩	🟩	🟩	🟩 *1	🟩 *1	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
PS Group	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
VM Group	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
ParkDevice	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
External Pager	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩
Central Office (CO)	🟩	🟩	🟩	🟩	🟩	🟩	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	🟩	🟩	🟩

Condition
*1 ACD mode only
*2 DPT only

PNQX2716YA KK1109MH1010