SCM
OpenTSP Driver Description

Ed. 00
10. 2013.

SAMSUNG
ELECTRONICS

COPYRIGHT

TRADEMARKS

Product names mentioned in this document may be trademarks and/or registered trademarks of their respective companies.

This manual should be read before the installation and operation, and the operator should correctly install and operate the product by using this manual.

This manual may be changed for the system improvement, standardization and other technical reasons without prior notice.

For further information on the updated manual or have a question the content of manual, contact **Document Center** at the address below .

Address : Document & Training Center 3rd Floor Jeong-bo-tong-sin-dong. 129, Samsung-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do, Korea 443-742

Homepage: http://www.samsungdocs.com

INTRODUCTION

Purpose

The OpenTSP Driver Manual provides a brief description on the OpenTSP, the installation procedure, and the procedure on using the functions of the OpenTSP, for engineers who develop the TAPI service and telephony application programs.

Document Content and Organization

This manual includes six chapters and the ‘Acronyms’. The chapters are summarized as follows :

CHAPTER 1. Introduction

This chapter provides description on the TAPI, used on the TSP driver before using the OpenTSP, the list of supported functions, and the list of functions used only on the TSP driver.

CHAPTER 2. OpenTSP Driver Installation

This chapter describes items that must be checked before installing the OpenTSP driver and the procedure for installing the OpenTSP driver.

CHAPTER 3. TAPI Functions

This chapter describes the TAPI functions and expansion functions supported by the OpenTSP driver.

CHAPTER 4. Call Processing Flow

This chapter describes the life cycle of the TAPI, various call processing events of the OpenTSP driver, and the call processing procedure.

ABBREVIATION

The frequently used acronyms and their meanings in this guide are all collected.

Conventions

The following special paragraphs are used in this document to point out information that must be read. This information may be set-off from the surrounding text, but is always preceded by a bold title in capital letters.

WARNING

Indicates a potentially hazardous situation which if not avoided, could result in death or serious injury.

CAUTION

Indicates a potentially hazardous situation which if not avoided, may result in minor or moderate injury. It may also be used to alert against unsafe practices.

CHECKPOINT

Provides the operator with checkpoints for stable system operation.

NOTE

Indicates additional information as a reference.

OPERATION PROCEDURES

Indicates the operation procedures that should be executed in order.

Console Screen Output

- The lined box with '`Courier New`' font will be used to distinguish between the main content and console output screen text.
- '**`Bold Courier New`**' font will indicate the value entered by the operator on the console screen

Revision History

Edition No.	Date of Issue	Remark
00	10. 2013.	Original

This page is intentionally left blank.

TABLE OF CONTENTS

INTRODUCTION

Purpose I
Document Content and Organization I
Conventions II
Console Screen Output III
Revision History III

Introduction 1

1 Introduction to OpenTSP 1

2 OpenTSP Features 2

OpenTSP Driver Installation 1

1 Installation Environment and Conditions 2
1.1 Installation Environment 2
1.2 Installation Conditions 4
1.3 Checking the Telephony Service 5

2 OpenTSP Driver Installation Procedure 6

3 Checking Installation Data 9
3.1 Checking the OpenTSP Driver Files 9
3.2 Checking the OpenTSP Driver Registration 9
3.3 Changing the OpenTSP Driver Environment Settings 10

TAPI Functions 1

1 List of the TAPI Functions 1

2 List of the Expansion Functions 3
2.1 Station Information Request 4
2.2 Dispatch Conference – Setup with Member(s) 5
2.3 Dispatch Conference – Add Member(s) 6
2.4 Dispatch Conference – Delete Member(s) 7

Call Processing Flow 1

1 Life Cycle of the TAPI 1

2 Call Processing Events for the OpenTSP Driver 3

2.1 Major Events 3
2.2 Flow Chart of Call Status 4
2.3 Flow Chart of the Status of Calls in Progress 5
2.4 Details of Calls 6
2.5 Holding Calls in Progress 7
2.6 Procedure for Consult Transfer 9

ABBREVIATION

A ~ R Abbreviation-1
T ~ T Abbreviation-2

LIST OF FIGURES

Figure 1.1 TAPI Configuration Diagram 1
Figure 2.1 OpenTSP Configuration Window 7
Figure 2.2 Selecting Configure 10
Figure 2.3 Configuration Item 11
Figure 4.1 Life Cycle of the TAPI 1
Figure 4.2 Example of Flow Chart of Call Status 4
Figure 4.3 Flow Chart of the Status of Calls in Progress 5
Figure 4.4 Messages of Call Status 6
Figure 4.5 Messages of Call Status 6
Figure 4.6 Flow Chart of Calls on Hold 7
Figure 4.7 Messages of Call Status 7
Figure 4.8 Messages of Call Status 8
Figure 4.9 Flow Chart of Call Forwarding Status 9
Figure 4.10 Call Status Messages for Consult Transfer of Extension 201 10
Figure 4.11 Call Status Messages for Consult Transfer of Extension 202 10
Figure 4.12 Call Status Messages for Consult Transfer of Extension 203 11

LIST OF TABLES

Table 2.1 OpenTSP Driver Installation Procedure 1
Table 2.2 OpenTSP Driver Installation Environment 2
Table 2.4 OpenTSP Configuration Items 7
Table 3.1 OpenTSP Supported TAPI Functions 1

CHAPTER 1

Introduction

1 Introduction to OpenTSP

The SCM OpenTSP Telephony Service Provider Driver 3.x(referred to as ‘OpenTSP’ hereinafter) interfaces with the Samsung key telephone switch through the TCP/IP system, based on TAPI 2.x/TAPI 3.x, and enables call control and call processing of the Telephony Application Programming Interface(TAPI) service through the TSPI. The OpenTSP driver is installed on a PC using the Windows O/S. The TAPI 2.x/TAPI 3.x was designed by Microsoft based on the TAPI standard.

The Microsoft TAPI consists of the three modules shown below:

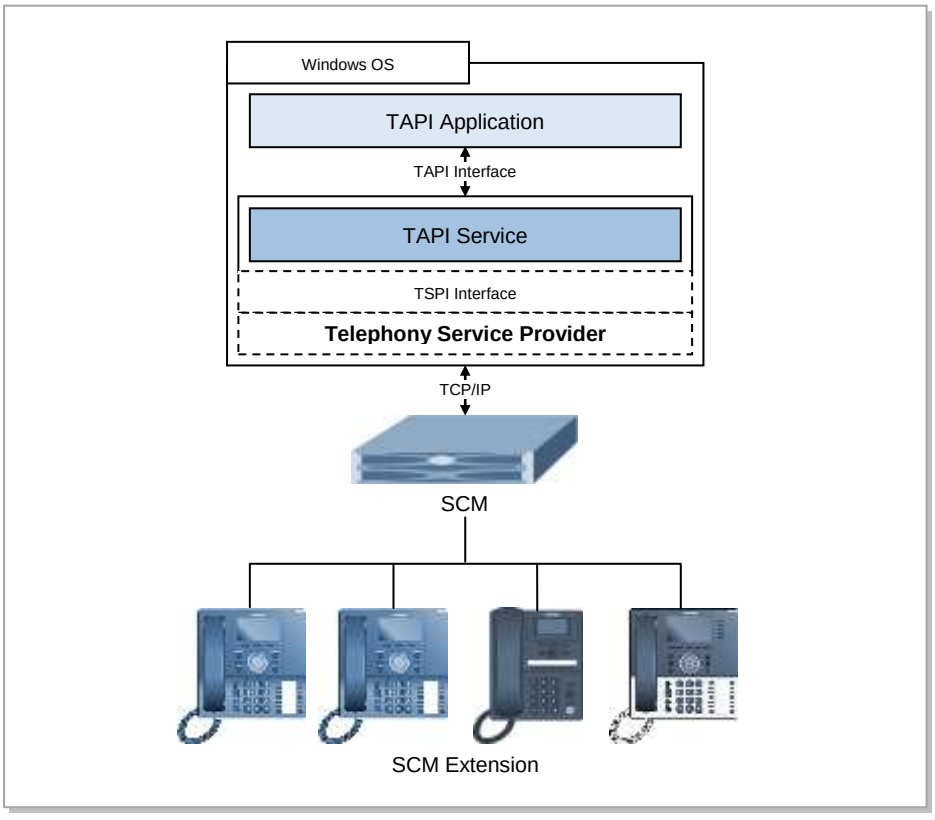


Figure 1.1 TAPI Configuration Diagram

Modules in Figure 1.1 are described below:

Telephony Application

Supplied by the application vender, the Telephony Application provides features such as call processing to the users through the TAPI, an API provided by the Microsoft TAPI Service.
Telephony applications include the ‘dialing’ program, basically embedded in the Windows OS, and the Outlook, the Contact Manager program of Microsoft.

TAPI Service

As a basic module of the Microsoft Windows OS, the TAPI Service directly uses the Telephony Service Provider installed on a PC upon the request of the application program.

Telephony Service Provider (TSP)

Provided by the switch vendor, the TSP is a service provider that communicates with the Microsoft TAPI. The TSP is executed when the application program requests the TAPI feature.

2 OpenTSP Features

Since the OpenTSP driver is a TSP composing the TAPI of the Windows OS, the Telephony Application program connects to the TAPI Service through a CTI to use the features of the Samsung key telephone system.
OpenTSP references the specification of the Microsoft TAPI, which supports the features introduced in ‘3.2. TAPI Function.’
Following features that are supplied only by the OpenTSP driver are described in ‘3. OpenTSP Driver Extended Function Feature List’.

- Station Information Request
- Dispatch Conference – Setup with Member(s)
- Dispatch Conference – Add Member(s)
- Dispatch Conference – Delete Member(s)

CHAPTER 2

OpenTSP Driver Installation

This chapter describes the environment and procedure required for the installation of the OpenTSP driver. For proper installation and operation of the OpenTSP driver, the installation environment and conditions should be checked before installation.
Refer to the table below, in which the installation procedure is summarized, when installing the OpenTSP driver.

Table 2.1 OpenTSP Driver Installation Procedure

Step	Procedure	Description
1	Installation Environment and Conditions	Check the following environment and condition before installing the OpenTSP driver. - Check the H/W and S/W environments. - Check for the OpenTSP driver license key.
2	OpenTSP Driver Installation Procedure	Install the OpenTSP driver according to the installation procedure. Read the Cautions and Notes carefully to prevent error during installation.
3	Installation Data Verification	After installing the OpenTSP driver, check if the installation is successful by verifying the driver file and the registration status.

1 Installation Environment and Conditions

The OpenTSP driver may be installed and executed for call processing on various Os of the Microsoft Windows. This section describes the environment and conditions that are required for proper installation of the OpenTSP driver.

1.1 Installation Environment

Check the installation environment below before installing the OpenTSP driver.

Table 2.2 OpenTSP Driver Installation Environment

Type	Category	Requirement
Hardware	Compatible Switch	The OpenTSP may only be used in the SCM system, which supports the TAPI 2.x interface.
	Switch Interface	The OpenTSP may only use the switch service through a CSTA Interface of SCM.
	Network Interface	A network card supporting TCP/IP protocol should be installed on the PC.
	Processor	* Client Mode - Dual Core 2 GHz or higher * Server Mode - Quad Core 2 GHz or higher
	Memory	* Client Mode - 1 GB or higher * Server Mode - 3 GB or higher
	HDD Capacity	500 MB or more free disk space(for Log files)
Software	TAPI version	TAPI 2.x or higher
	OS	- Windows 7 Professional or later (x86 only supported) * for x64, the separated OpenTSP x64 version will be released.

Table 2.3 OpenTSP Driver Installation Environment

License (Number of Lines Registered)	Requirements	Remarks
In case of 1	- Windows x86 OS - Intel CPU Dual Core 2 GHz or higher - RAM 1 GB or more higher	Windows OS x86
1,000, lower than 8 CPS	- Windows x86 OS - Intel CPU Dual Core 2 GHz or more higher - RAM 2 GB or more higher	Windows OS x86
3,000, lower than 18 CPS	- Windows x86	Windows OS x86

	- Intel CPU Quad Core 2 GHz or more higher - RAM 4 GB or higher	

1.2 Installation Conditions

Check the items below before installing the OpenTSP driver on the system.

Valid License

The license key is required for installing and using the OpenTSP driver.

- Client Mode: SCM -> External Application -> 3rd Party CSTA License should be registered.
- Server Mode: the valid OpenTSP License should be registered with the matched Computer MAC Address.

Procedure for checking the information on the TSP driver installed on the computer is as follows:

1) Select 'Start→Settings→Control Panel' on the computer.

2) Double click 'Phone and Modem Options' from the 'Control Panel' shown below :

3) Select [Edit(E)] from the 'Phone and Modem Options' window.

4) Enter the fields of the 'Edit Location' Window and click the [OK] button.

Select 'Country/Region' and enter your area code. Do not enter '0' of the area code. For example, if the area code is 031, enter '3
The 'Dialing Rules' option is used for making external calls through the TAPI. Enter the number to be used for making external calls. The number assigned for outside calls is usually '9'.

Dialing Rules

Since phones in offices usually connect to the trunk line through a private switch, consult the telephony manager of your company for information on the number assigned for outside calls.

5) Select the 'Advanced' tab from the 'Phone and Modem Options' screen to display the list of drivers(telephony service providers) installed on the system.

The TAPI compatible driver can be installed separately on each computer, and a newly added TAPI driver is displayed on the 'Advanced' tab of the 'Phone and Modem Options'.
The TAPI driver is registered as 'SCM Telephony Service Provider for Windows x86'.

1.3 Checking the Telephony Service

The procedure for checking if the Telephony service is normally operating is as follows:

1) Select 'Start→ Control Panel' from the computer.

2) Double click the 'Administrative Tools' from the control panel below.

3) Select 'Service' from the 'Administrative Tools' window.

4) Check the 'Telephony' service(TAPI service) status from the 'Service' window below. If the 'Telephony' service is displayed as 'started', the TAPI service is operating.

Terminating/Restarting the Telephony Service

Users can terminate or restart the telephony service, if necessary, through the above window.

5) The telephony service of the Windows OS is related to the following services. Thus, the services below should be checked for normal operation.

- Remote Access Auto Connection Manager
- Remote Access Connection Manager

Cases where the OpenTSP Driver is not Properly Loaded or Unloaded

If the two services above are operated 'Manually' the two services will act as a single module of the telephony service. If the two services were abnormally set during the system setup procedure, the two services and the telephony service may not operate normally and the OpenTSP driver may not be loaded or unloaded properly, disabling the use of related application programs. To avoid such incidents, it is recommended to set the start type as 'Not Used' to disable unnecessary services. If the two services are marked as 'Started' and the OpenTSP does not operate properly, change the 'start type' to 'Not Used' and reboot the system.

2 OpenTSP Driver Installation Procedure

The procedure for installing the OpenTSP driver is as follows:

> **OpenTSP Driver Installation Privilege**
>
> To install OpenTSP Driver, you have to login as Administrator.

1) Double click the OpenTSP installation file(SCM OpenTSP Driver-Vxxx-YYYYMMDD.exe)

2) Click the [Next>] button on the screen below.

3) Read the License Agreement below and select [Yes] to approve. Select [No] to abort the installation program.

4) The ‘Choose Destination Location’ window appears as shown below. Click the [Next>] button to use the default path(C:\Program Files\Samsung Call Manager OpenTSP Driver) or click the [Browse] button to change the installation folder.

> **OpenTSP Driver Installation Folder**
>
> The OpenTSP installation program installs two types of programs on the user’s computer.
> The SCMTSP32.TSP file(basic SCM Telephony Service Provider file) is copied to the C:\Windows\system32 folder and is registered to TAPI .
> Other Files are copied to the C:\Program Files\Samsung Electronics\Samsung Call Manager OpenTSP Driver folder.

5) Select ‘Typical’ from the ‘Setup Type’ window below and click the [Next>] button.

6) The ‘Select Program Folder’ window below appears. Click the [Next>] button to use the default name(SCM OpenTSP Driver). Enter a new name into the field to change the folder name.

7) Enter the entry items of the ‘Phone and Modem Options’ window below.

> **‘Dialing Rules’ Setup**
> Refer to the steps from 1) to 4) of the ‘Samsung TAPI Driver’‘ in ‘1.2 Installation Conditions’ for setting the ‘Dialing Rules’.

8) Enter the items of the ‘Configuration’ window below and click the [OK] button.

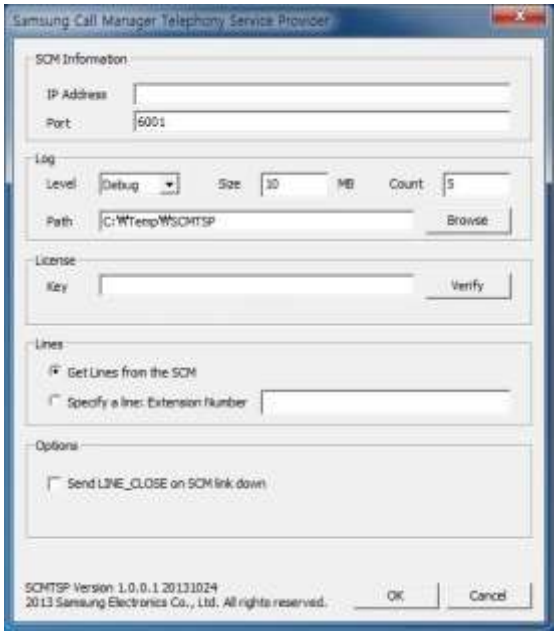


Figure 2.1 OpenTSP Configuration Window

Table 2.4 OpenTSP Configuration Items

Item	Function	Remarks
IP Address	IP Address of SCM	
Port	Default 6001	This value depends on the User Group Configuration of SCM.

Log Level	Represents the Log Level of OpenTSP Driver. (Default : Normal) None/Critical/Error/Warning/Normal /High/Debug	
Log Size	File Size for each log file	Default Value : 5 MB
Log Count	Log File Count	Default Value : 5
Path	Log File Path	Default Path: C:\Temp\SCMTSP
License Key	Register the valid License Key	Without the valid license, OpenTSP will be working as the client mode. With this mode, only 1 line can be opened.
Get Lines from SCM	When the OpenTSP driver is loaded, Telephone Numbers will be downloaded from SCM	
Specify a line	When the OpenTSP is loaded, the specified line will be only opened .	Be sure that the wanted line is registered within SCM.
Send Line Closed Event on the SCM Link Down	When the OpenTSP driver is disconnected from SCM unexpectedly, Line Close Event will be delivered to TAPI Application.	

9) Upon successful installation of the OpenTSP driver, the ‘OpenTSP Setup Complete’ window appears. Click the [Finish] button.

3 Checking Installation Data

3.1 Checking the OpenTSP Driver Files

During the installation of the OpenTSP driver, the OpenTSP driver and files should have been copied to the folders below:

Location of the OpenTSP driver file (TSP itself)

The OpenTSP driver is copied to the following folder.

- windows\system32\scmtsp32.tsp

Location of the OpenTSP driver files (Others)

The OpenTSP files should have been copied to the folder below if the default location in 'Selecting Installation Folder' was not changed.

- Program Files\Samsung Electronics\Samsung Call Manager OpenTSP Driver

3.2 Checking the OpenTSP Driver Registration

The registration of the OpenTSP driver file, which is registered as the TAPI driver of the Microsoft Windows OS, can be verified as follows. :

Check for the 'SCM Telephony Service Provider for Windows x86' on the 'Advanced' tab of the 'Start→Control Panel→Phone and Modem Options'.

3.3 Changing the OpenTSP Driver Environment Settings

The user may change the environment settings of the OpenTSP driver. The driver environment can be changed either through the 'Phone and Modem Options' screen or through the OpenTSP Config Tool program.

Changing from the 'Phone and Modem Options' screen

1) Click the [Configure] button from the 'Start→Control Panel→Phone and Modem Options→Advanced' screen.

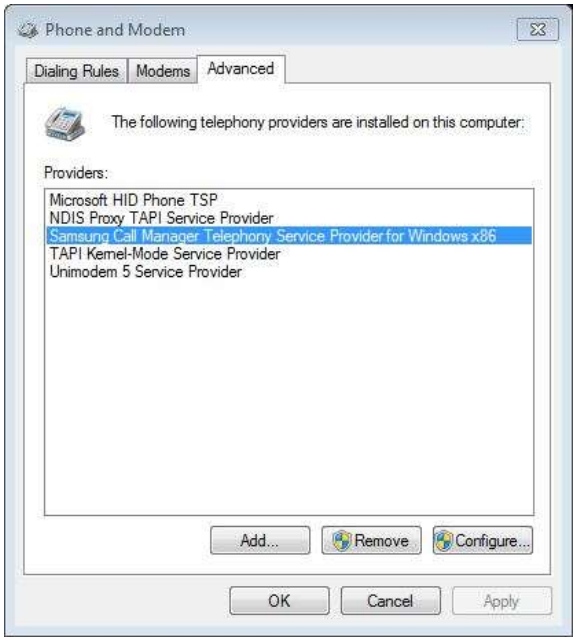


Figure 2.2 Selecting Configure

2) On the OpenTSP environment configuration screen below, change the settings and click the [OK] button.

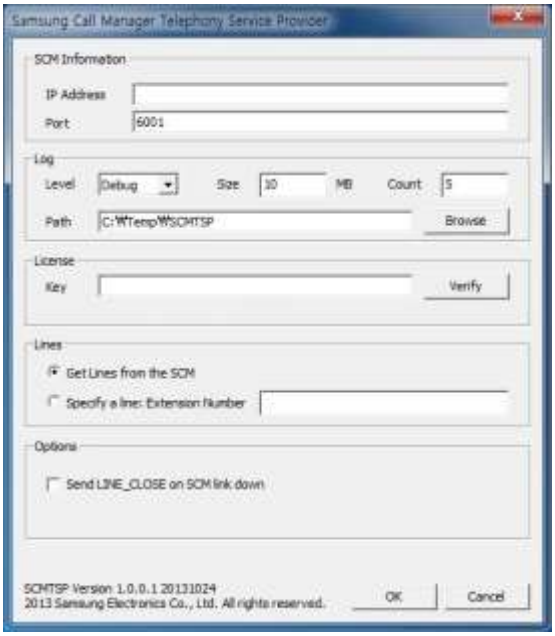


Figure 2.3 Configuration Item

3) The OpenTSP driver must be restarted to apply the changes except Log and License. In those cases close all CTI application programs and restart the OpenTSP driver.

Removing the OpenTSP Driver

Remove the OpenTSP driver installed on the system when the driver is no longer needed or when removing a previous version to install a new version of the driver.

Closing all TAPI compatible programs

All TAPI compatible application programs that are currently running must be closed before removing the OpenTSP driver. If the OpenTSP driver is being operated by a TAPI compatible application program, error may occur during the uninstallation process.

The procedure for removing the OpenTSP driver is as follows:

1) Select 'Start→Settings→Control Panel→Add/Delete Program' to display the screen below. Then, select the 'SCM OpenTSP Driver' and click the [Change/Remove] button.

2) Among the radio buttons, select the 'Remove' item and click the [Next>] button.

3) Click [OK] on the message window confirming the removal of the OpenTSP driver files.

4) Files related to the OpenTSP driver are removed from the system.

CHAPTER 3

TAPI Functions

This chapter describes the features of the TAPI functions and expansion functions that the OpenTSP driver supports.

1 List of the TAPI Functions

Restriction

The OpenTSP driver supports INTERACTIVEVOICE mode and only the Line Device function out of the list of the TAPI functions of Microsoft.

List of the TAPI Functions

The list of the TAPI functions that the OpenTSP driver enables is shown below :

Table 3.1 OpenTSP Supported TAPI Functions

TAPI Functions of Microsoft	Remarks
LineAnswer	Off-Hook
LineBlindTransfer	Consultation Call+Transfer
LineClose	
LineCompleteCall	Camp on+Msg Waiting+OHVA+Callback
LineCompleteTransfer	Transfer
LineDeallocateCall	Idle Call Remove
LineDevSpecific	Refer to 3.2 List of the OpenTSP Driver Expansion Functions.
LineDial	Make Call
LineDrop	
LineForward	Set/Reset Forward/DND
LineGetAddressCaps	
LineGetAddressID	
LineGetAddressStatus	
LineGetCallInfo	

TAPI Functions of Microsoft	Remarks
LineGetCallStatus	
LineGetDevCaps	
LineGetID	
LineGetLineDevStatus	
LineHold	Hold
lineMakeCall	Make Call
lineNegotiateExtVersion	
lineOpen	
linePickup	Direct Pickup+Group Pickup
lineRedirect	Redirect
lineSetAppSpecific	
LineSetCallData	dwSize should be less than or equal to 256
lineSetMediaMode	
lineSetStatusMessages	
lineSetupTransfer	Consultation Call
lineSwapHold	
lineUnhold	

The list above shows only the TAPI functions supported by the OpenTSP driver : Some functions from the list of the TAPI functions might be processed by the TAPI service itself. Also, some functions, which are used to add the Telephony Service Provider to the system, are not included in the list. If the functions that are not supported by the OpenTSP driver are called, an error message defined in the TAPI will appear.
The user can find the entire list of the Microsoft TAPI functions from the Microsoft site
(http://msdn.microsoft.com/en-us/library/windows/desktop/ms734273(v=vs.85).aspx)
The user can check the format of each function, how to use the functions, or the status values returned from the list and refer to the list to develop an application.

2 List of the Expansion Functions

Besides the functions defined by the TAPI, the OpenTSP driver provides a variety of expansion functions. Call the lineDevSpecific function to use the expansion functions.

Enter the syntax below to call the lineDevSpecific function :

```
LONG lineDevSpecific(HLINE hLine, DWORD dwAddressID, HCALL hCall,
LPVOID lpParams, DWORD dwSize);
```

The features available by the lineDevSpecific function in the Samsung key telephone system are as follows :

- Station Information Request
- Dispatch Conference – Setup with Member(s)
- Dispatch Conference – Add Member(s)
- Dispatch Conference – Delete Member(s)

2.1 Station Information Request

Application can get the current status of the station.

Enter the syntax below to call the lineDevSpecific function when the Station Lock is used in the TAPI application:

```
LONG lineDevSpecific(HLINE hLine, DWORD dwAddressID, HCALL hCall,
LPVOID lpParams, DWORD dwSize);
```

Input Parameter Values

- hLine : Processes the lines to be used.
- dwAddressID : 0
- hCall : Not used
- lpParams : Enters the command strings as shown below :

Value	Byte
'S' 'C' 'M'	3 Bytes
'F'	1 Byte

- dwSize : Buffer length

2.2 Dispatch Conference – Setup with Member(s)

Application can use the dispatch conference function

Enter the syntax below to call the lineDevSpecific function when the Vacant Station Message is used in the TAPI application:

```
LONG lineDevSpecific(HLINE hLine, DWORD dwAddressID, HCALL hCall,
LPVOID lpParams, DWORD dwSize);
```

Input Parameter Values

- hLine : Processes the lines to be used.
- dwAddressID : 0
- hCall : Not used
- lpParams : Enters the command strings as shown below :

Value	Byte
'S' 'C' 'M'	3 Bytes
'5'	1 Byte
Member Lists to be added(Separated by ',')	N Byte

- dwSize : Buffer length

2.3 Dispatch Conference – Add Member(s)

Application can add the member(s) into the current dispatch conference call.

Enter the syntax below to call the lineDevSpecific function when the Follow Me is used in the TAPI application :

```
LONG lineDevSpecific(HLINE hLine, DWORD dwAddressID, HCALL hCall,
LPVOID lpParams, DWORD dwSize);
```

Input Parameter Values

- hLine : Processes the lines to be used.
- dwAddressID : 0
- hCall : Not used
- lpParams : Enters the command strings as shown below :

Value	Byte
'S' 'C' 'M'	3 Bytes
'6'	1 Byte
Member List to be added(Separated by ',')	N Bytes

- dwSize : Buffer length

2.4 Dispatch Conference – Delete Member(s)

Application can delete the member(s) from the current dispatch conference call.

Enter the syntax below to call the lineDevSpecific function when the Make New Trunk Call is used in the TAPI application :

```
LONG lineDevSpecific(HLINE hLine, DWORD dwAddressID, HCALL hCall,
LPVOID lpParams, DWORD dwSize);
```

Input Parameter Values

- hLine : Processes the lines to be used.
- dwAddressID : 0
- hCall : Not used
- lpParams : Enters the command strings as shown below :

Value	Byte
'S' 'C' 'M'	3 Bytes
'7'	1 Byte
Member List to be deleted(Separated by ',')	N Byte(s)

- dwSize : Buffer length

This page is intentionally left blank.

CHAPTER 4

Call Processing Flow

This chapter describes the life cycle of the TAPI, the call processing events of the OpenTSP driver, and call processing procedures.

1 Life Cycle of the TAPI

The knowledge of the TAPI life cycle shown below is needed to use the TAPI-based application to process calls :

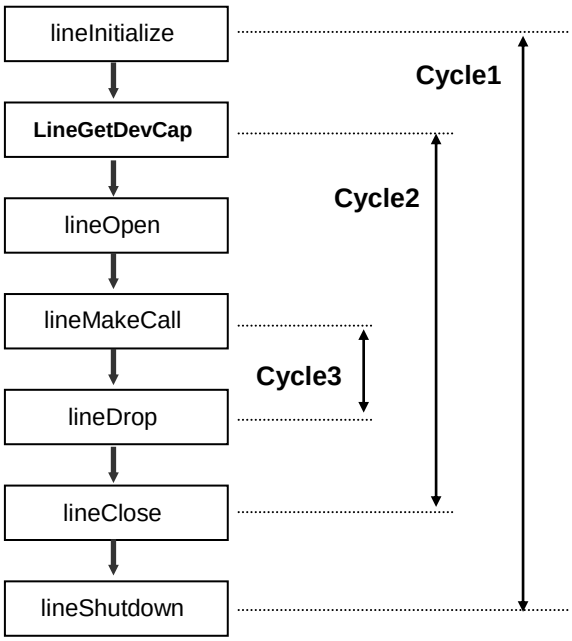


Figure 4.1 Life Cycle of the TAPI

Cycles 1 to 3 shown in Figure 6.1 are described below:

Cycle 1

The Phone Dialer and call center makes/connect/answers a call according to the steps of Cycle 1 as shown in Figure 4.1:
Each application can use the lineInitialize() function and then other TAPI functions.

Also, the lineShutdown() function should be called to prevent the TAPI-compatible application from using the TAPI function.

Each TAPI-compatible application can call the lineInitialize() function to check the number of the devices available by the TAPI service and register the processing modules for call processing events generated from each device. Also, the TAPI service loads the unloaded Telephony Service Provider(TSP) on the TAPI service by executing the lineInitialize() function to change each TSP driver to an operating state. Different TAPI-compatible applications can simultaneously call the lineInitalize() function. The information registered during each calling is automatically classified and sorted by the TAPI service.

The lineShutdown() function is used when each TAPI-compatible application does not use the TAPI functions any more. If this function is called, the TAPI service will delete the information registered during the lineInitialize() process in order not to report the call status events generated from each device.
Also, if the lineShutdown() function is called when any application does not use the TAPI service, the TAPI service will upload all the loaded TSPs.

Cycle 2

Call the lineInitialize() function to find out the number of devices available in the TAPI service of the system. Then, the TAPI-compatible application calls the lineOpen() function to make necessary line devices available to each application. The application, which has got permissions for the line after executing the lineOpen() function, can receive information on call processing in each line device and use functions on calls.
The TAPI-compatible application calls the lineClose() function when the application disables the line devices. If the lineClose() function is called, the call processing events generated from the line devices will not be reported and the functions for call processing cannot be used for the line devices.

Cycle 3

The TAPI-compatible application that has permissions for each line device through the lineOpen() function can use call processing functions for the line. Also, since the TAPI-compatible application receives the call processing events for the status of all calls, it can be defined to perform necessary operations according to the processing rule of the application. The call processing functions are available only if call objects exist in the line device.

How to Use Functions

For information about how to use the functions, refer to the Microsoft web sites about the TAPI.

2 Call Processing Events for the OpenTSP Driver

The TAPI service of the system offers the call processing events generated from the line device to the application after calling the lineOpen() function so that the TAPI-compatible application can use a specific line device as shown in the TAPI Life Cycle 2 of Figure 6.1.
This section describes the type of the call processing events to be reported while, the call processing events are generated from the Samsung key telephone system and forwarded to the TAPI service through the OpenTSP driver as well as processing procedures.

2.1 Major Events

LINE_CALSTATE and LINE_CALLINFO are the events that all the TAPI-compatible applications should process by default. These events are reported when the status of a specific call and the details of each call are changed in each line device.

LINE_CALLSTATE

The LINE_CALSTATE event is reported when the status of calls is possibly changed in each line device. Examples of the call status include IDLE, RINGBACK, OFFERING, CONNECT, HOLD, and DISCONNECT. The call status is reported in event of status transition. When the LINE_CALLSTATE event is generated, the TAPI-compatible application calls the lineGetCallState() function to read the details of call status.

LINE_CALLINFO

The LINE_CALLINFO event is reported when information on calls in each line device is changed. Information on calls needed during call processing includes caller ID/name, called party ID/name, the phone number/name of the person to whom a call is forwarded, and call status. The information can be changed. When the information is changed, the LINE_CALLINFO event is reported. The TAPI-compatible application calls the lineGetCallInfo() function to read the details when the LINE_CALLINFO event is generated.

Extension Name

SCM System allows ‘%’ in the Extension name string. ‘%’ may cause errors when formatting and printing a string in a C/C++ software. Be cautious when dealing whith ADDRESSNAME fields.

2.2 Flow Chart of Call Status

When call status is changed, the OpenTSP driver reports information on call status through the LINE_CALLSTATE and LINE_CALLINFO events. The LINE_CALLSTATE event reports the information on the status change of the generated calls and the LINE_CALLINFO event reports the information on the details of each call are changed.
The figure below briefly shows the change of call status from call generation to termination:

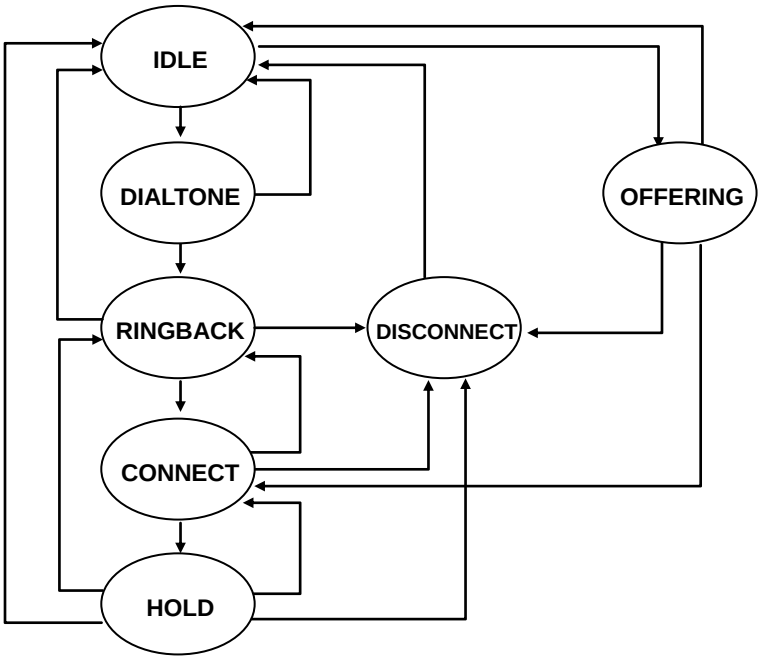


Figure 4.2 Example of Flow Chart of Call Status

Outgoing and incoming calls are exemplified below:

When Making a Call from Extension 2001 to Extension 2002

The procedure for making a call from Extension 2001 is as follows :

- IDLE→DIALTONE→RINGBACK→CONNECTED→DISCONNECTED→ IDLE

The procedure for connecting a call with Extension 2002 is as follows :

- IDLE→Offering→CONNECTED→DISCONNECT→IDLE

When call status is changed as described above, the OpenTSP driver forwards the LINE_CALLSTATE event to the TAPI-compatible application through the TAPI service. The TAPI-compatible application calls the lineGetCallInfo() function to obtain the details of call status.

2.3 Flow Chart of the Status of Calls in Progress

The events and messages generated while a call is being processed(i.e. from making a call to connecting a call) by the TAPI-compatible application are as follows :
The figure below shows the example of events on call status sent to the TAPI-compatible applications of both a caller and called party when a call is in progress:

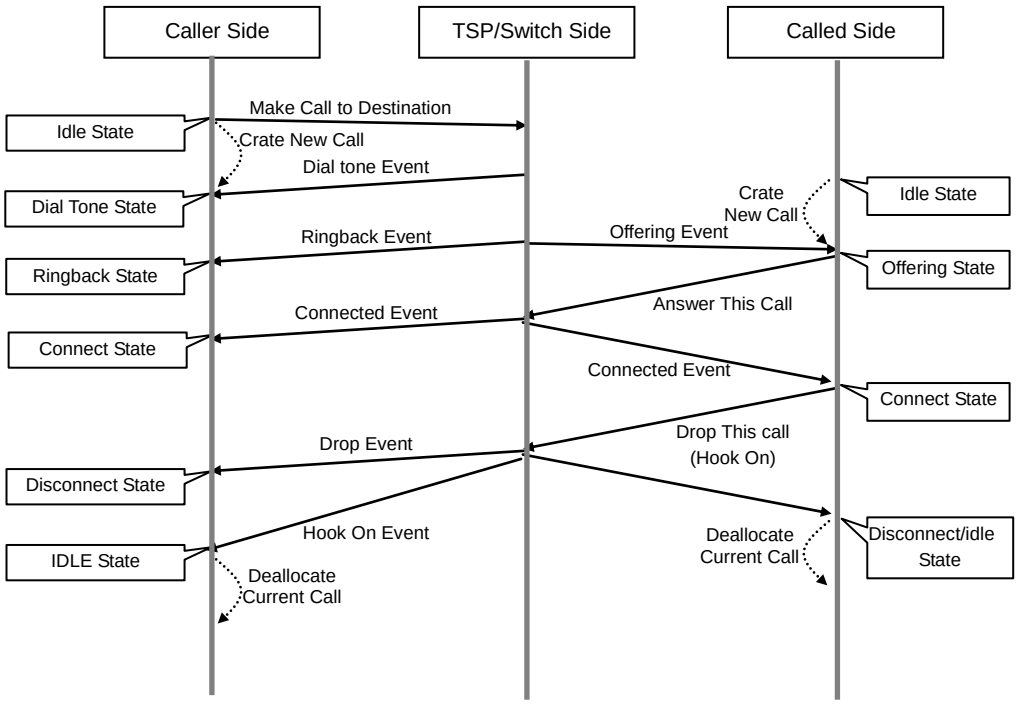


Figure 4.3 Flow Chart of the Status of Calls in Progress

2.4 Details of Calls

When call status is changed like a call is made from Extension 201 to Extension 202 by using the TAPI Sampler program, the status message of call is displayed for each occurred time. The messages of call status change include caller IDs, called IDs, trunk numbers, DNIS information, call directions, and reasons for call generation.

As shown in the displayed screen below, the messages of call status show both the status change of the call(LINE_CALLSTATE) and the change of the details of each call(LINE_CALLINFO). Once the details of calls are changed, the details are forwarded from the OpenTSP to the TAPI service through the LINE_CALLINFO event. Also, the TAPI-compatible application calls the lineGetCallInfo() function to read the changed information or one to be checked.

Caller (Extension 201)

Time	Event
09:24:29	201 opened.
09:24:54	LINECALLSTATE_DIALTONE received
09:24:55	LINECALLSTATE_RINGBACK received
09:24:55	LINECALLINFO on 0x000102ee received
09:24:55	LINECALLINFO: 0x000102ee
	dwTotalSize=0x000001a8 dwNeededSize=0x00000154 dwUsedSize=0x00000154
	hLine=0x00010355 dwLineDevieceID=0x00000005 dwAddressID=0x00000000
	dwTrunk=0xffffffff(-1)
	CalledID=202
	ConnectedID=202
09:24:57	LINECALLSTATE_CONNECTED received
09:24:57	LINECALLINFO on 0x000102ee received
09:24:57	LINECALLINFO: 0x000102ee
	dwTotalSize=0x000001a8 dwNeededSize=0x00000154 dwUsedSize=0x00000154
	hLine=0x00010355 dwLineDevieceID=0x00000005 dwAddressID=0x00000000
	dwTrunk=0xffffffff(-1)
	CalledID=202
	ConnectedID=202
09:25:02	LINECALLSTATE_DISCONNECTED received
09:25:04	LINECALLSTATE_IDLE received
09:25:04	IDLE Call Handle (0x000102ee) deallocated

Figure 4.4 Messages of Call Status

Called Party (Extension 202)

Time	Event
09:24:36	202 opened.
09:24:55	LINECALLSTATE_OFFERING received
09:24:55	LINECALLINFO on 0x000102cc received
09:24:55	LINECALLINFO: 0x000102cc
	dwTotalSize=0x000001a8 dwNeededSize=0x0000015c dwUsedSize=0x0000015c
	hLine=0x00010311 dwLineDevieceID=0x00000006 dwAddressID=0x00000000
	dwTrunk=0xffffffff(-1)
	CallerID=201
	CalledID=202
	ConnectedID=201
09:24:57	LINECALLSTATE_CONNECTED received
09:24:57	LINECALLINFO on 0x000102cc received
09:24:57	LINECALLINFO: 0x000102cc
	dwTotalSize=0x000001a8 dwNeededSize=0x0000015c dwUsedSize=0x0000015c
	hLine=0x00010311 dwLineDevieceID=0x00000006 dwAddressID=0x00000000
	dwTrunk=0xffffffff(-1)
	CallerID=201
	CalledID=202
	ConnectedID=201
09:25:02	LINECALLSTATE_IDLE received
09:25:03	IDLE Call Handle (0x000102cc) deallocated

Figure 4.5 Messages of Call Status

2.5 Holding Calls in Progress

If an extension asks for holding a call in progress, the TAPI-compatible applications of both the caller and called party will receive the call status events as shown in the figure below :

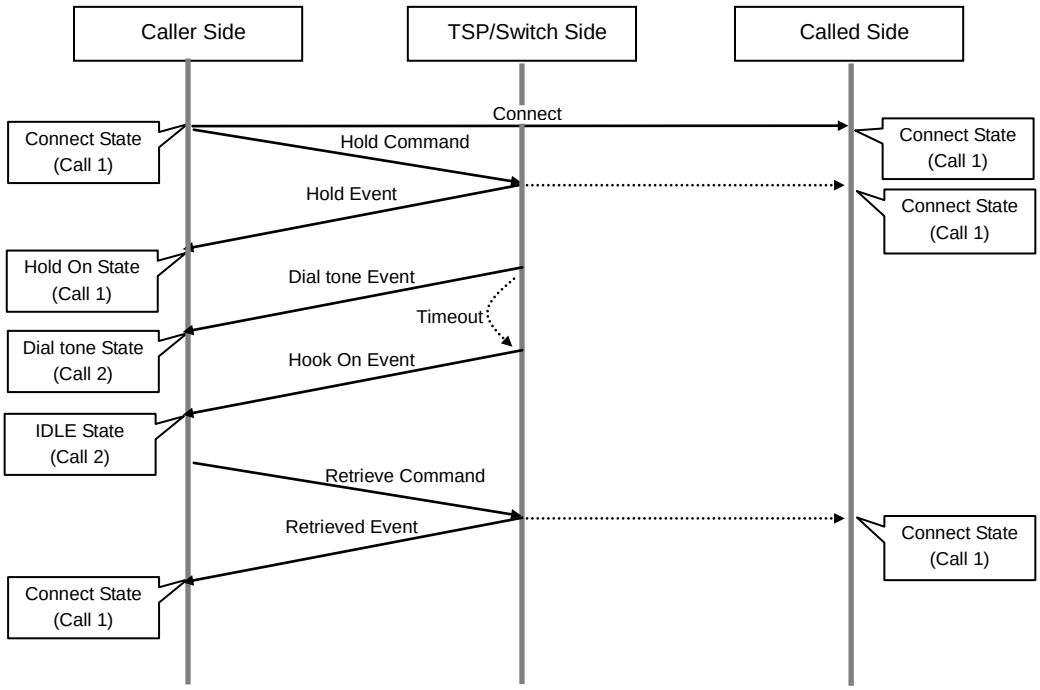


Figure 4.6 Flow Chart of Calls on Hold

The status message created when a call in progress is held is displayed on the TAPI Sampler program as shown below :

The Extension that has Asked for Holding a Call

Time	Event
10:10:03	LINECALLSTATE_ONHOLD received
10:10:03	LINECALLSTATE_DIALTONE received
10:10:05	LINECALLSTATE_IDLE received
10:10:05	IDLE Call Handle (0x000101ee) deallocated
10:10:05	LINECALLSTATE_CONNECTED received

Figure 4.7 Messages of Call Status

The Extension where a Call has been Held

Time	Event
10:10:03	LINECALLSTATE_CONNECTED received
10:10:03	LINECALLINFO on 0x00010222 received
10:10:03	LINECALLINFO: 0x00010222 dwTotalSize=0x000001a8 dwNeededSize=0x0000015c dwUsedSize=0x0000015c hLine=0x00010311 dwLineDevieceID=0x00000006 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CallerID=201 CalledID=202 ConnectedID=201
10:10:05	LINECALLSTATE_CONNECTED received
10:10:05	LINECALLINFO on 0x00010222 received
10:10:05	LINECALLINFO: 0x00010222 dwTotalSize=0x000001a8 dwNeededSize=0x0000015c dwUsedSize=0x0000015c hLine=0x00010311 dwLineDevieceID=0x00000006 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CallerID=201 CalledID=202 ConnectedID=201

Figure 4.8 Messages of Call Status

2.6 Procedure for Consult Transfer

If either Extension 201 or Extension 202 forwards an extension call to another extension(203) while Extension 201 or Extension 202 is making the call, the TAPI-compatible applications of the caller, called party, and forwarded party will receive the call status events as shown in the figure below :

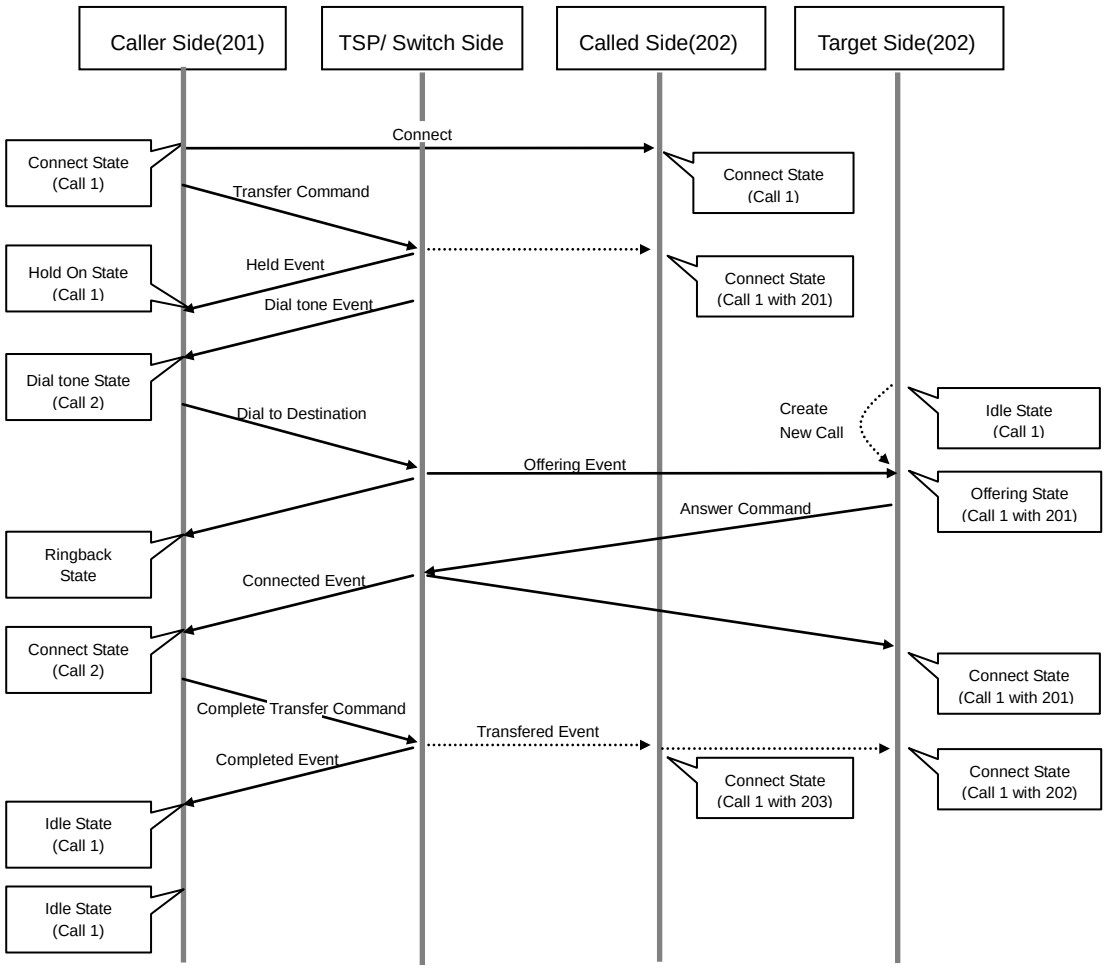


Figure 4.9 Flow Chart of Call Forwarding Status

The call status messages for the consult transfer are displayed on the TAPI Sampler program as shown below :

Extension 201

Time	Event
10:59:41	LINECALLSTATE_ONHOLDPENDTRANSFER received on 0x00010289
10:59:41	LINECALLSTATE_DIALTONE received on 0x000101ef
10:59:44	LINECALLSTATE_RINGBACK received on 0x000101ef
10:59:44	LINECALLINFO on 0x000101ef received
10:59:44	LINECALLINFO: 0x000101ef dwTotalSize=0x000001a8 dwNeededSize=0x00000154 dwUsedSize=0x00000154 hLine=0x00010044 dwLineDevieceID=0x00000005 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CalledID=203 ConnectedID=203
10:59:46	LINECALLSTATE_CONNECTED received on 0x000101ef
10:59:46	LINECALLINFO on 0x000101ef received
10:59:46	LINECALLINFO: 0x000101ef dwTotalSize=0x000001a8 dwNeededSize=0x00000154 dwUsedSize=0x00000154 hLine=0x00010044 dwLineDevieceID=0x00000005 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CalledID=203 ConnectedID=203
10:59:49	LINECALLSTATE_IDLE received on 0x000101ef
10:59:50	IDLE Call Handle (0x000101ef) deallocated
10:59:50	LINECALLSTATE_IDLE received on 0x00010289
10:59:50	IDLE Call Handle (0x00010289) deallocated

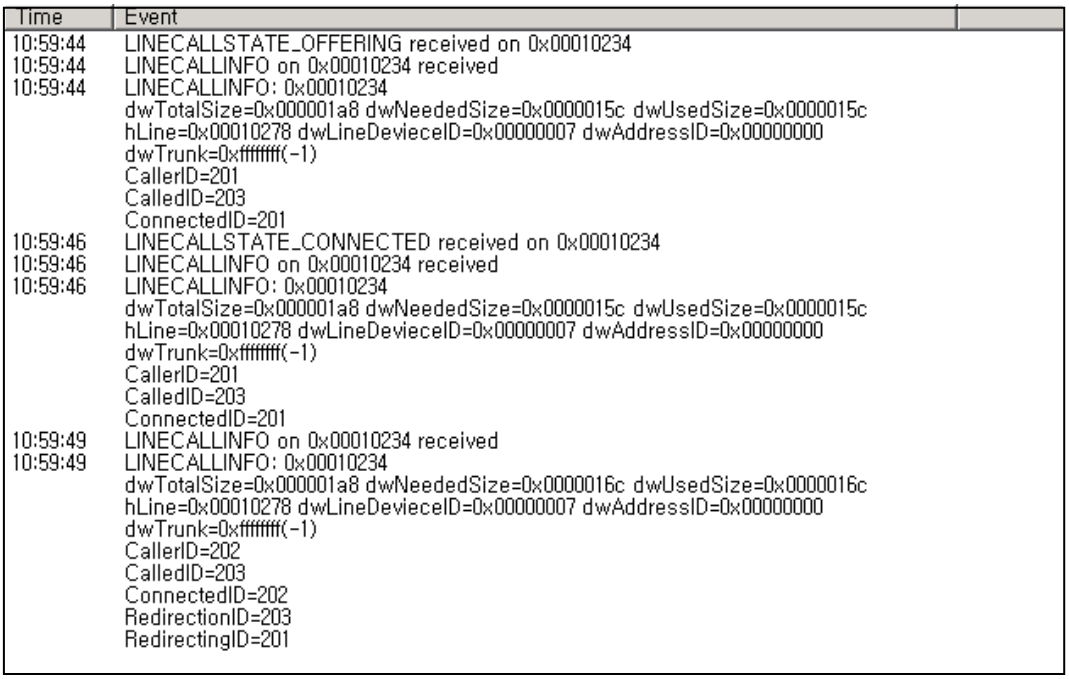
Figure 4.10 Call Status Messages for Consult Transfer of Extension 201

Extension 202

Time	Event
10:59:41	LINECALLSTATE_CONNECTED received on 0x00010212
10:59:41	LINECALLINFO on 0x00010212 received
10:59:41	LINECALLINFO: 0x00010212 dwTotalSize=0x000001a8 dwNeededSize=0x0000015c dwUsedSize=0x0000015c hLine=0x00010033 dwLineDevieceID=0x00000006 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CallerID=201 CalledID=202 ConnectedID=201
10:59:49	LINECALLSTATE_CONNECTED received on 0x00010212
10:59:49	LINECALLINFO on 0x00010212 received
10:59:49	LINECALLINFO: 0x00010212 dwTotalSize=0x000001a8 dwNeededSize=0x0000016c dwUsedSize=0x0000016c hLine=0x00010033 dwLineDevieceID=0x00000006 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CallerID=201 CalledID=202 ConnectedID=203 RedirectionID=203 RedirectingID=201

Figure 4.11 Call Status Messages for Consult Transfer of Extension 202

Extension 203



Time	Event
10:59:44	LINECALLSTATE_OFFERING received on 0x00010234
10:59:44	LINECALLINFO on 0x00010234 received
10:59:44	LINECALLINFO: 0x00010234 dwTotalSize=0x000001a8 dwNeededSize=0x0000015c dwUsedSize=0x0000015c hLine=0x00010278 dwLineDevieceID=0x00000007 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CallerID=201 CalledID=203 ConnectedID=201
10:59:46	LINECALLSTATE_CONNECTED received on 0x00010234
10:59:46	LINECALLINFO on 0x00010234 received
10:59:46	LINECALLINFO: 0x00010234 dwTotalSize=0x000001a8 dwNeededSize=0x0000015c dwUsedSize=0x0000015c hLine=0x00010278 dwLineDevieceID=0x00000007 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CallerID=201 CalledID=203 ConnectedID=201
10:59:49	LINECALLINFO on 0x00010234 received
10:59:49	LINECALLINFO: 0x00010234 dwTotalSize=0x000001a8 dwNeededSize=0x0000016c dwUsedSize=0x0000016c hLine=0x00010278 dwLineDevieceID=0x00000007 dwAddressID=0x00000000 dwTrunk=0xffffffff(-1) CallerID=202 CalledID=203 ConnectedID=202 RedirectionID=203 RedirectingID=201

Figure 4.12 Call Status Messages for Consult Transfer of Extension 203

This page is intentionally left blank.

ABBREVIATION

A

API	Application Program Interface
ASP	Abstract Service Primitive

C

CD	Compact Disk
CTI	Computer Telephony Interface
CPS	Character Per Second
CSTA	Computer Supported Telephony Application

D

DND	Do Not Disturb
DNIS	Dialed Number Identification Service
DTMF	Dual Tone Multi-Frequency

H

HTTP	Hypertext Transfer Protocol

I

ID	Identification
IP	Internet Protocol

O

OHVA	Off-hook Voice Announce

P

PIDs	Process Identifications

R

ROM	Read Only Memory

S

SCM	Samsung Communications Manager
SDK	Software Development Kit

T

TAPI	Telephony Application Programming Interface
TCP	Transmission Control Protocol
TSP	Telephony Service Provider
TSPI	Telephony Service Provider Interface

SCM

OpenTSP Driver Description

Visit us at

http://www.samsungdocs.com

SAMSUNG ELECTRONICS