UNISYS

BTOS
Voice Services
Programming Reference Manual

Relative to Release
Level 1.0

Priced Item

February 1987

U S America
5026016

UNISYS

BTOS Voice Services

Programming Reference Manual

Relative to Release
Level 1.0

Priced Item

February 1987
Distribution Code SA
Printed in U S America
5026016

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THIS DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed Program Product License or Agreement to purchase or lease equipment. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such License or Agreement. Unisys cannot accept any financial or other responsibility that may be the result of your use of the information or software material, including direct, indirect, special or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Comments or suggestions regarding this document should be submitted on a Field Communication Form (FCF) with the CLASS specified as 2 (S.S.W.:System Software), the Type specified as 1 (F.T.R.), and the product specified as the 7-digit form number of the manual (for example, 5026016).

Correspondence regarding this publication should be forwarded, using the User Comment sheet at the back of this manual, or remarks may be addressed directly to Unisys Corporation, Corporate Product Information,User Comment Coordinator, C1-NE19, P.O. Box 500, Blue Bell, PA 19422-9990 U S A.

About This Manual

This reference manual explains how to use BTOS Voice Services to interface between Unisys hardware and BTOS application software programs.

Who Should Use This Manual

This manual is intended for BTOS Voice Services programmers creating application software that uses the Telephone Server. To understand the information in this manual, you should be familiar with:

- the workstation operating system (BTOS)
- the BTOS applications you are using with Voice Services

How to Use This Manual

If you are using BTOS Voice Services for the first time, you should read sections 1 and 2. They provide an overview of the product and discuss Voice Services commands.

In any case, if you scan the contents and review the topics before you start, you may find this manual easier to use. To find definitions of unfamiliar words, use the glossary; to locate specific information, use the index.

How This Manual is Arranged

This manual is comprised of five sections, with related subjects grouped together. Section 1 describes the basic concepts involved in Voice operations. Thereafter, the general sequence of topics covers Voice commands, programming concepts, operations, and data structures.

Procedures

For BTOS Voice Services operations procedures, refer to section 4.

Conventions

The following conventions apply to all procedures:

- When you perform an operation by holding down one key and pressing another key, the key names are hyphenated (for example, **ACTION-GO**).
- The term character includes spaces.

Reference Material

This manual contains appendixes with reference information, a glossary, and an index.

For suggested error message responses, refer to appendix A.

For definitions of key terms used in this manual or related to this software, refer to the glossary.

Related Product Information

For an explanation of BTOS, refer to the *BTOS Reference Manual.*

For more information about BTOS commands, refer to the *BTOS Standard Software Operations Guide.*

Contents

Tables

Overview

What are Voice Services?

The Voice Services are a set of commands, software programs, and data files which provide an interface between the hardware and application software programs.

Telephone Server

The Telephone Server system service is an extension of your current operating system which provides the request and procedural interfaces application programs used to access the hardware for voice and telephone related operations.

Voice Management

The Voice Management facility of the Telephone Server system service makes it possible for an application program to use the analog to digital signal conversion (codec) feature of the hardware to encode or decode voice (such as a telephone conversation) to or from binary data stored on disk. Note that the hardware and software do not do voice recognition or speech synthesis.

Telephone Management

The telephone management feature of the Telephone Server system service allows an application program to take control of all the functions of the hardware, including automatic dialing, placing a call on hold, switching between calls on different lines, and automatic answering when a line is ringing.

Telephone Status Program

The Telephone Status program is a systems administration or software programming tool. It provides a simplified, visual picture of the hardware circuits that are connected when a telephone management function is executed. The Telephone Status utility has soft function keys that allow the user to select which lines to dial or answer, which

calls to place on hold, which lines to hang up or disconnect, and which lines to link for conference calling. As each function is executed, the screen visually connects the circuits that makes the operation take place.

The Telephone Status utility can be used to verify the proper operation of Voice Services, and as a debugging tool during program development.

Hardware Requirements for Voice

Voice messages and responses are recorded at either 32 KbS (8KHz) or 24KbS (6KHz), with optional compression of pauses.

The disk requirement without pause compression is 4096 bytes per second at 8 KHz and 3072 bytes per second at 6 KHz. The data rate with pause compression is about two/thirds of that, depending on the speaker. The following figures show these disk requirement values.

Pause Compression	No	No	Yes	Yes
Data Rate	8KHz	6KHz	8KHz	6KHz
Bytes per Second	4K	3K	~2.7K	~2K
Bytes per Minute	240K	180K	~160K	~120K
Bytes per Hour	14M	11M	~9M	~7M
Minutes/Megabyte	4.25	5.67	~6.33	~8.5

Cluster Utililization

Digital voice activity to or from a file located at the master workstation uses approximately 12% of the cluster bandwidth per workstation with a 1.8Mb line and 70% with 307Kb line.

CPU Utilization

The Telephone Server uses an average of less than 1% of CPU bandwidth for background, dialing, and modem activity.

During voice data transfers, the CPU utilization averages 15% (8MHz 80186 processor).

Comments

The Voice Services commands allow you to install the Telephone Server system service into memory, deinstall the system service, and access the Status Monitor for a visual map of the internal hardware state.

Install Telephone Server

You install the Telephone Server into active memory by using the INSTALL TELEPHONE SERVER command.

Command Form

INSTALL TELEPHONE SERVER
[Configuration file]

[Sys]<Sys>TmServer.Run
Command Case: Is

Parameter Fields
[Configuration file]

The configuration file contains information used by the Telephone Server system service for the generation of dial characters, ringing, etc. as described in tables 5–5 and 5–6.

If this field is left blank, then the default configuration file, [sys]<sys>TmConfig.sys, is used (if it exists). If a file is specified in this field, it must be a valid file.

The configuration file specified here is also used by the Operator program for storing information such as the phone numbers of each line. Because of this, each workstation in a cluster should be given its own configuration file.

Deinstall Telephone Server

The DEINSTALL TELEPHONE SERVER command is used to remove the Telephone Server system service from active memory.

Command Form

DEINSTALL TELEPHONE SERVER

[Sys]<Sys>TmServer.Run
Command Case: DD

Telephone Status

The TELEPHONE STATUS command is used to enter the Voice Services Status Monitor. The Status Monitor is a visual map of the analog crosspoint switch inside the hardware. This switch is used to connect incoming and outgoing calls with the rest of the hardware.

The Status Monitor displays several intersecting lines that originate and end in the upper right hand corner of the screen. These lines represent the telephone connection and external telephone lines attached to the back of the Voice Processor Module. By using the soft function keys shown at the bottom of the screen, you can perform several telephone functions and watch as they are carried out inside the Telephone Manager Module.

Command Form

TELEPHONE STATUS
[Run file]

[Sys]<Sys> TMSTATUS.RUN
Command Case: TS

Status Monitor Screen

The Status Monitor screen shows four vertical lines, marked J0 through J3, and eight horizontal lines, L0 through L7, that are used to connect the telephone, phone line 1, and phone line 2 to the hardware components of the Voice Processor Module. These components are the dialer, detector, encoder, and decoder. For detailed information about these components, see the *BTOS Systems Hardware Installation Guide.*

As circuits are completed and functions carried out, the Status Monitor lights up the lines on the screen to show a map of the internal circuitry of the module. All connections begin and end with the telephone, and take place through phone line 1, phone line 2, or both.

Phone, Lines 1 and 2

These three connections correspond to the plug-in jacks on the back of the Voice Processor Module. This is where the telephone and two external telephone lines plug into the module.

Lines 1 and 2 are for separate telephone lines from the local phone company. The telephone is the telephone in your office or home. For further information about connections to the Voice Processor Module, see the *BTOS Systems Hardware Installation Guide.*

Lines J0 through J3

Lines J0 and J1 are the connecting lines between the telephone and phone lines 1 and 2. They are lit when incoming calls pass through the internal hardware to make a connection with the telephone. Line J0 corresponds to phone line 1, and line J1 corresponds to phone line 2.

Lines J2 and J3 are auxiliary lines that act as jumpers to facilitate internal hardware switching. Telephone calls to the dialer or encoder/decoder, for example, are connected via lines J2 and J3. These lines also light up to show you how the internal switching takes place.

Lines L0 through L7

Lines L0, L4, and L7 are the internal connections to the telephone handset.

Line L0 is a straight through connection to the phone, without any modulation of the signal. Line L4 has an amplifier to boost voice signals for better clarity, and line L7 has an attenuator to reduce the sound of the high volume dialing signals generated inside the module. Each of these lines completes an internal circuit depending upon the function being performed.

Line L1 is reserved. L2 is the dialer, which generates the tones necessary to dial outside numbers from the workstation keyboard. Line L3 is the detector, which is used to detect activity on the incoming and outgoing lines, such as busy signals and dial tones.

Lines L5 and L6 are the encoder and decoder (codec). They make the conversion from analog (voice) to digital (binary) signals and back again, so that your workstation can process voice messages as file data. The answering machine functions of some application software take advantage of the encoder and decoder.

Status Monitor Function Keys

The function keys on the workstation keyboard allow you to perform several telephone operations that appear on the Status Monitor screen. These function keys are as follows:

F1	Line 1	Makes the connection between the telephone and phone line 1.
F2	Line 2	Makes the connection between the telephone and phone line 2.
F3	Hold	Places the currently active phone line on hold (pressing F1 or F2 takes the line off hold).
F4	Monitor	Allows you to screen incoming calls without callers knowing you are listening in. (Designed for use with an application software's answering machine function).
F5	Link	Allows conference calling by connecting the telephone, phone line 1 and phone line 2 together.
F6	Unused	
F7	DTMF	Toggles on or off the reading of touch-tone characters. The DTMF characters are displayed on the screen as they are detected.

F8	Redial	Allow you to type in a number on the keyboard's numeric keypad and dial it by pressing F9. (This replaces, but does not obstruct, dialing from the telephone handset).
F10	Hang up	Disconnects the phone line from the telephone and hangs it up.

To leave Status Monitor and return to the Executive, press **FINISH**.

Dialing Tips

Dialing can take place on the telephone attached to your workstation, or on the numeric keypad of the keyboard. Numbers entered from the keypad, including long distance calls, can be entered without spaces and parentheses.

Some numbers need special characters entered for the Telephone Server to complete the call. For example, an outside call placed from a PBX system needs a pause inserted between the PBX number and the outside number dialed, like this number below.

```
9=9462233
```

For a list of special characters used with the Telephone Server, refer to table 5–2.

Mistakes made while entering numbers can be corrected by using the **BACKSPACE** key and the Arrow direction and **DELETE** keys with **CODE** and **SHIFT**.

Programming Concepts

Voice Services provides access to hardware for two classes of applications; voice management and telephone management.

The Voice Management services allow applications (such as Word Processor and Mail) to do voice digitization.

The Telephone Management services allow application software (such as the Operator) to control the telephone functions of the hardware.

Naming Conventions

The following conventions are those which are used to express numbers and name variables in the following operations.

Numbers

Numbers are decimal except when suffixed with h for hexadecimal. Thus, 11 = 11 and 11h = 17.

Variable Names

The name given a variable indicates its characteristics, according to a formal naming convention. The parameters used in the procedure definitions, fields of request blocks, and other data structures of Voice Services are named according to this convention. For further information on these naming conventions, see the *BTOS Reference Manual.*

A variable name is composed of up to three parts: a prefix, a root, and a suffix.

Prefixes

The prefix identifies the data type of the variable. The following are some of the common prefixes.

a an absolute memory address (24 bits)

b byte (8 bit character or unsigned number)

c count (unsigned number)

f flag (TRUE = 0FFh or FALSE = 0)

i index (unsigned number)

p logical memory address (pointer) (32 bits consisting of the offset and the segment base address)

q quad (32–bit unsigned integer)

rg array of....

s size in bytes (unsigned number)

Prefixes can be composed. Common compound prefixes are as follows:

cb count of bytes (the number of bytes in a string of bytes)

pb pointer to (logical memory address of) a string of bytes

Roots

The root of a variable name can be unique to that variable, can be selected from the list below, or can be a compound of the two. The following common roots are:

erc status (error) code

exch exchange

fh file handle

lfa logical file address

lh line handle

ph partition handle

rq request block

th telephone handle

Suffixes

The suffix identifies the use of the variable. Suffixes are as follows:

Last	the largest allowable index of an array
Max	the maximum length of an array or buffer (thus one greater than the largest allowable index
Min	the minimum length of an array or buffer
Ret	identifies a variable whose value is to be set by the called process or procedure rather than specified by the calling process

Voice Management

The Voice Interface is a set of services that allows application programs to use the codec (voice encoder/decoder) features of the hardware for voice annotation, voice messaging, or as part of a software answering machine.

Using the Voice Services

There are two general classes of programs which access the Voice Services: ones which use the services in a simple fashion, and ones which are more advanced in their use of the voice services.

Voice Control Structure (VCS)

The TsVoiceRecordToFile and TsVoice PlaybackFromFile operations require the caller to pass a pointer to a structure called the Voice Control Structure (VCS), as shown in table 5-2. This structure contains information such as the file handle of the voice file, starting and ending logical file addresses, and data byte counts.

Simple Use of the Voice Services

The simple type of application is one in which the details of recording and playback connections, line quality, and so on are not important. An example of this type of application would be one where voice annotation of documents is desired.

Recording Voice

To record a voice message, the simple type of application needs to allocate a Voice Control Structure and set the following fields:

fh set to the file handle returned by OpenfFile operation.

lfaStart
lfaMax

the logical file addresses of starting and ending locations in the file where data are to be placed (normally 512 for lfaStart and the file size for lfaMax)

qSampleStart

the sample number to be assigned to the first data byte recorded (normally 0)

qSampleMax

the sample number that, when reached, will cause the recording to terminate (normally 0FFFFFFFFh)

fAutoStart

set this to true (0FFh). This causes the recording to be started as soon as the voice unit (aka phone or telephone set becomes off hook (picked up).

All other fields should be set to 0. The file should be created large enough for a large voice message.

After the operation completes, the first 512 bytes of the file should be used as a header to store information about the recording as described in table 5-3.

To conserve disk space, truncate the file (using ChangeFileLength) to the size returned in the variable pointed to by pLfaNext in the TsVoiceRecordToFile operation.

Playing Back Voice

To play back voice, the VCS should be initialized as above with the following exception: The lfaMax and qSampleMax fields in the VCS should be set to the values returned in the variables pointed to by pLfaNext and pqSampleNext in the TsVoiceRecordToFile operation. Store in the first 512 bytes of the file.

Advanced Use of the Voice Services

More advanced use of the voice services is possible for applications in which the details of controlling connections and so forth are important. An example of this type of application is an answering machine.

Such an application would set the fAutoStart field of th VCS to false (0), and set other fields as appropriate. The TsOffHook, TsVoiceConnect, and other operations would also be used.

An example of such a use may be found in the Telephone Management section in this chapter.

Termination of Recording or Playback

An application program controls the termination of a recording or playback by setting the control information in the VCS. A recording is terminated when any of the following conditions occur:

- an error occurs
- a TsVoiceStop command is issued
- the phone is hung up (recording from phone)
- a TsOnHook or TsHold command is issued (recording from line)
- a TsDoFunction (function hang up) command is issued
- the voice data fill the portion of the file specified (lfaStart and lfaMax)
- the maximum number of data bytes is reached (qSampleMax)
- a pause (silence) greater than the maximum occurs (cPauseMax)
- dial tone was detected (fStopOnDialTone)

Data format

The voice data are encoded (after hardware compression) at a rate of either 8KHz: 32K bits per second (4096 data bytes per second), or 6KHz: 24K bits per second (3072 data bytes per second). The Telephone Server system service store the data in 512–byte records as described in section 5, table 5-4.

Pause Compression

During recording, the hardware inserts escape bytes into the data stream. These escape bytes tell the system service when the voice levels fall below (7Fh) or rise above (0F7h) a certain threshhold. The data byte is replaced by 7Fh and 0F7h is added to the data.

If the fRawData flag is true in the VCS, then all of the data, including escape sequences, are written to disk during recording and escape sequences are ignored during playback. If the fRawData flag is false and the fNoPause flag is true in the VCS, then the escape sequences are not written to disk.

Setting fNoPause to true will result in slightly higher quality voice recordings, at the expense of disk space (approximately 50% more).

If both flags are false, then during voice recording the Telephone Server system service replaces a portion of the data which is below the threshhold with an escape sequence. The escape sequence consists of the byte 7Fh followed by a word containing the number of data bytes being discarded.

The field sPauseGap in the VCS is used to determine how much of the data are not discarded after a 7Fh escape is encountered and before the 0F7h escape is encountered.

During playback the discarded data bytes (number specified in the stored escape sequence) are replaced with value 08h, which is the encoded value of silence.

Memory Usage

The voice services require a 13312–byte or larger (1024 byte increments) work area. The work area is divided into an 8192–byte data queue used by the hardware, and the balance (5120 minimum) is used for two file system I/O buffers.

Additional memory improves performance. The disk performance is largely determined by disk seek time, which occurs once per I/O regardless of the number of bytes transferred. The larger the buffer sizes, the more bytes transferred per I/O, and fewer disk seeks needed.

There is no routing of requests: the application must run on the workstation where the hardware is located.

The hardware for digitizing is nonshareable. It is not possible to playback on one line while recording on the other.

Operations

The following are operational procedures:

TsVoiceConnect
TsVoicePlaybackFromFile
TsVoiceRecordToFile
TsVoiceStop

Telephone Management

The Telephone interface allows a program to directly control the functions of the hardware.

The module supports two telephone lines and a voice unit which can be connected to the dialing hardware, the detectors, and the voice encoder/decoder (codec).

Making Connections

The TsConnect, TsDoFunction, TsHold, TsOffHook, and TsOnHook services are used to make connections between the two telephone lines and the voice unit.

Dialing

The TsDial service is used to dial out flash, pulse, or DTMF (touch tone) characters from a telephone line or the voice unit. It is also used for call progress tone receiving (CPTR) to detect dial tone, or detect when the called number has answered.

Automatic Calling

Automatic dialing is accomplished by combining use of the Voice and Telephone Services. The following program fragment, writtem in BTOS Pascal, shows an example of dialing a number and playing a message (created with the Operator, for example), invoked from the Executive with the following command form:

Autodial
[Phone number]
[Voice file]

The variable and structure declarations are omitted. The AllocMemorySL, CheckErc; Exit, OpenFile, Read, and RgParam operations are described in the *BTOS Reference Manual.*

```
/*
** Allocate memory for voice.
*/
sWorkArea := 8000h;
CheckErc(AllocMemorySl(sWorkArea, ADS pWorkArea));

/*
** Get parameters and open voice file.  VCS is the Voice
** Control Structure (table 5-2) and VDFH is the data
** file header (table 5-3).
*/
CheckErc(RgParam(1,0,ADS sdNumber)):
CheckErc(RgParam(2,0,ADS sdFile));
CheckErc(OpenFile(ADS VCS.fh,sdFile.pb,sdFile.cb,0,0,'mr'));
CheckErc(Read(VCS.fh,ADS VDFH,size(VDFH),0,ADS sDataRet)):

/*
** Setup the fields of the Voice Control Structure from the
** voice data file header, set autostart to false.
*/
VCS.lfaStart := VDFH.lfaStart;
VCS.qSampleStart := VDFH.qSampleStart;
VCS.qSampleMax := VDFH.qSampleMax;
VCS.f6KHz := VDFH.f6KHz;
VCS.fAutoStart := false;

/*
** Let server use its defaults for the rest of the fields.
*/
VCS.cPauseMax :- 0;
VCS.cSampleOn := 0
VCS.SampleOff := false;
VCS.fStopOnDialTone := false;
VCSfAltConnection := false;
VCS.nSectorStatusUpdate := 0;
VCS.fRawData := false;

/*
** put line off hook, wait for dial tone, dial, wait for
** answer.
*/
CheckErc(TsOffBook (1,ADS th,0));
ch:= '=';
CheckErc(TsDial(1,0,ADS ch,1,100,ADS cbDialRet));
CheckErc(TsDial(1,0,sdNumber.pb,sdNumber.cb,100, ADS
cbDialRet));
ch := '?';
CheckErc(TsDial(1,0,ADS ch,1,100,ADS cbDialRet));

/*
** Lock the codec, specify connection, and play message.
*/
Check Erc(TsDoFunction(1,12));
CheckErc(TsVoiceConnect(1,false,true,false));
CheckErc(TsVoicePlaybackFromFile(1,pWorkArea,sWorkArea, ADS
VCS, size(VCS),ADS lfaNext,ADS qSampleNext,ADS status));

/*
** Hang up and exit.
*/
CheckErc(TsOnHook(1.0));
Exit;
```

Status

The TsGetStatus service reports changes in the state of the module such as a line ringing.

Configuration

The TsQueryConfigParams, TsSetConfigParams, and TsRing services are used to query and update the configuration.

Operations

The procedures are as follows:
TsConnect
TsDeinstall
TsDial
TsDoFunction
TsGetStatus
TsHold
TsOffHook
TsOnHook
TsQueryConfigParams
TsReadTouchTone
TsRing
TsSetConfigParams

Operations

Introduction

This section lists all the Voice Services operations available to programmers working with the Telephone Manager Module.

Each of the listed operations is presented in the following form:

RoutineName(param-1, ..., param-n): Type

where:

Routine Name	is the name of the operation
param-n	is the parameter used in the call
Type	is the type of the returned alue (usually ercType or word).

The type of parameter used in an operation is identified by the name of the variable. For more information about the naming conventions used, refer to section 3, Programming Concepts, or see the *BTOS Reference Manual.*

TsConnect

Description

The TsConnect service is used to connect or disconnect the voice unit and telephone lines to each other.

Connecting a line which is on hold (see TsHold) terminates the hold condition.

Error code 11202 (invalid connection) will be returned if the connection is not allowed or if it requires the use of hardware already connected.

Error code 11205 (bad handle) will be returned if the handle specified is not valid.

Procedural Interface

TsConnect (iTmModule, th, device, fAdd) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
th	is a telephone handle returned by TsOffHook or TsGetStatus
device	specifies the line or phone mode, where: 0 means telephone line 1. 1 means telephone line 2. 2 means voice unit 3 means voice unit connected in monitor mode
fAdd	if true, the device is added to the connection, otherwise it is removed.

Request Block

TsConnect

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	8
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	802 Eh
12	iTmModule	2	
14	th	2	
16	device	2	
18	fAdd	2	

TsDeinstall

Description

The TsDeinstall request is issued by a program to deinstall the telephone server. The server will ignore all of its request codes, respond to any outstanding requests with the specified error code, cleanup its internal state, place the Voice Processor Module in power-off state, and then respond to the TsDeinstall request with its partition handle.

Procedural Interface

TsDeinstall (iTmModule, pPhRet, erc) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module.
pPhRet	points to a word where the telephone server partition handle will be returned.
erc	is the error code to be returned to any client programs of the telephone server.

Request Block

TsDeinstall

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	4
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	1
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8038h
12	iTmModule	2	
14	erc	2	
16	pPhRet	4	
20	sPhRet	2	2

TsDial

Description

The TsDial service causes the DTMF to generate the specified characters. The DTMF generator is automatically connected to the specified line.

Procedural Interface

TsDial (iTmModule, line, pbString, cbString, cErrorTimeout, pcbStringRet) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
line	is the phone line to be connected
pbString cbString	describe the number to be dialed. The string contains characters as defined in table 5–1
cErrorTimeout	is the maximum time that is allowed before returning an error when waiting for dial tone
pcbStringRet	points to a return status structure which contains the number of characters processed

Request Block

TsDial

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	6
1	RtCode	1	0
2	nReqPbCb	1	1
3	nRespPbCb	1	1
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	802F h
12	iTmModule	2	
14	line	2	
16	cErrorTimeOut	2	
18	pbString	4	
22	cbString	2	
24	pcbStringRet	4	
28	scbStringRet	2	2

TsDoFunction

Description

The TsDoFunction service causes the telephone server to perform functions similar to those available by pressing buttons on a typical two line telephone.

Procedural Interface

TsDoFunction (iTmModule, function) : ErcType

where:

iTmModule — is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module.

function — is the function to be done, where

0	selects line 1 and connects to it, placing other line on hold if it was connected
1	selects line 2 and connects to it, placing other line on hold if it was connected
2	selects line 1 and connects to it, hanging up other line if it was connected
3	selects line 2 and connects to it, hanging up other line if it was connected
4	places selected line on hold
5	hangs up selected line
6	turns on speaker phone and connects to selected line
7	turns off speaker phone and hangs up lines attached to it
8	links lines 1 and 2 together
9	unlinks lines and places unselected line on hold
10	switches the voice unit into monitor mode.

11	switches the voice unit into normal mode.
12	locks the voice encoder/decoder for the exclusive use of caller.
13	unlocks the voice encoder/decoder.
14	sets switch so that calling application is brought forward whenever the voice unit goes off hook.
15	resets Context Manager switch.
16	resets dialing
17	resets detector
18	resets modem.
19	resets voice.

Request Block

TsDoFunction

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	4
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8031 h
12	iTmModule	2	
12	function	2	

TsGetStatus

Description

The TsGetStatus service returns the state of the hardware and all users.

Procedural Interface

TsGetStatus (iTmModule, pStatusRet, sStatusRetMax, fNoWait) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
pStatusRet sStatusRetMax	describe the memory area where the status is to be returned
fNoWait	if true, the status will be returned immediately. If false the status will be returned the next time something changes, or immediately if the iEvent field in the status block pointed to by pStatusRet is not the same as the current value of iEvent.

Request Block

TsGetStatus

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	4
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	1
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	802D h
12	iTmModule	2	
14	fNoWait	2	
16	pStatusRet	4	
20	sStatusRetMax	2	

TsHold

Description

The TsHold service causes the specified line to be disconnected from all devices and placed on hold.

Procedural Interface

TsHold (iTmModule, line) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
line	is either 0 or 1 to specify the telephone line

Request Block

TsHold

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	4
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8034 h
12	iTmModule	2	
14	line	2	

TsOffHook

Description

The TsOffHook service makes the specified line go off hook. It does not change any connections of the cross point switch.

The service returns a handle which may be used for adding devices with the TsConnect operation.

Procedural Interface

TsOffHook (iTmModule, pThRet, line) : ErcType
where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
pThRet	points to the word where a telephone connection handle will be returned
line	is either 0 or 1 to specify telephone line 1 or 2

Request Block

TsOffHook

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	4
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8032 h
12	iTmModule	2	
14	line	2	
16	pThRet	4	
20	sThRet	2	2

TsOnHook

Description

The TsOnHook service causes the specified line to be disconnected from all devices and go on hook or hung up.

Procedural Interface

TsOnHook (iTmModule, line) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
line	is either 0 or 1 to specify telephone line 1 or 2

Request Block

TsOnHook

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	4
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8033h
12	iTmModule	2	
14	line	2	

TsQueryConfigParams

Description

The TsQueryConfigParams service is used to get the Telephone Server configuration file name and the current configuration information.

Procedural Interface

TsQueryConfigParams (iTmModule,
pConfigRet,sConfigRetMax,
pbFileSpecRet, cbFileSpecRetMax,
pcbFileSpecRet) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
pConfigRet sConfigRetMax	describe the memory where the Telephone Server Configuration Block (refer to table 5–6) is returned.
pbFileSpecRet cbFileSpecRetMax	describe the memory where the file specification of the Telephone Server configuration file is returned
pcbFileSpecRet	points to a word where the actual size of the file specification is returned

Request Block

TsQueryConfigParams

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	2
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	3
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8035 h
12	iTmModule	2	
14	pConfigRet	4	
18	sConfigRetMax	2	
20	pbFileSpecRet	4	
24	cbFileSpecRetMax	2	
26	pcbFileSpecRet	4	
30	scbFileSpecRet	2	2

TsReadTouchTone

Description

The TsReadTouchTone service is used to read the DTMF signal from a telephone line or the voice unit.

Procedural Interface

TsReadTouchTone (iTmModule, device, pbDigits, cbDigitsMax, pcbDigitsRet, bStopDigit, cTimeout) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
device	specifies the line or phone, where: 0 means telephone line 1 1 means telephone line 2 2 means voice unit
pbDigits cbDigitsMax	describe the memory area where the digits are returned
pcbDigitsRet	is a pointer to a word where the actual number of digits read is returned
bStopDigit	is a character which will terminate the input. The character may be 0 to 9, *, #, or A to D. An illegal character is interpreted as meaning there is no stop digit
cTimeOut	is the time (in units of 100 ms) after which the read will be terminated if there are no characters received

Request Block

TsReadTouchTone

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	8
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	2
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8030h
12	iTmModule	2	
14	device	2	
16	bStopDigit	2	
18	cTimeOut	2	
20	pbDigits	4	
24	cbDigitsMax	2	
26	pcbDigitsRet	4	
30	scbDigitsRet	2	2

TsRing

Description

The TsRing service is used to turn on (or off) the monitor ringing, allowing an application and user to select a ring frequency interactively. If either line rings or if the voice unit becomes off hook the monitor ringing will return to normal.

Procedural Interface

TsRing (iTmModule, hz) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
hz	is the frequency to be used. A value of 0 means no ringing. The range is 0 to 255

Request Block

TsRing

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	3
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8037h
12	iTmModule	2	
14	hz	1	

TsSetConfigParams

Description

The TsSetConfigParams service is used to set the Telephone Server configuration information.

Procedural Interface

TsSetConfigParams (iTmModule, pConfig, sConfig) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
pConfig sConfig	describe the Telephone Server configuration block as described in table 5–6.

Request Block

TsSetConfigParams

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	2
1	RtCode	1	0
2	nReqPbCb	1	1
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8036h
12	iTmModule	2	
14	pConfig	4	
18	sConfig	2	

TsVoiceConnect

Description

The TsVoiceConnect service is used to specify the connection to be used with a TsVoicePlaybackFromFile or TsVoiceRecordToFile operation in which the fAutoStart flag in the Voice Control Structure is false. If the TsVoiceConnect operation is executed before the record or playback operation is completed, the voice must be locked first with the TsDoFunction operation.

Error code 11202 (invalid connection) will be returned if a connection is invalid or if the voice is not locked.

Procedural Interface

TsVoiceConnect (iTmModule, fVoiceUnit, fLine0, fLine1) : ErcType

where:

iTmModule	is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module
fVoiceUnit	if true then the codec will be connected to the voice unit
fLine0	if true then the codec will be connected to line 0
fLine1	if true then the codec will be connected to line 1

Request Block

TsVoiceConnect

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	5
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8023h
12	iTmModule	2	
14	fVoiceUnit	1	
15	fLine0	1	
16	fLine1	1	

TsVoicePlaybackFromFile

Description

The TsVoicePlaybackFromFile service plays back voice from the specified file.

Procedural Interface

TsVoicePlaybackFromFile (iTmModule, pWorkArea, sWorkArea, pVoiceControl, sVoiceControl, pLfaLast, pqSampleLast, pStatusRet) : ErcType

where:

iTmModule — is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module

pWorkArea
sWorkArea — describe a memory location to be used by the Telephone Service for communicating to the Voice Processor Module

pVoiceControl
sVoiceControl — describe the Voice Control Structure shown in table 5–2

pLfaLast — points to the returned actual lfa at the end of playback

pqSampleLast — points to the returned actual sample number at end of playback.

pStatusRet — points to the return status of the operation, where:

- 0 means termination because of an error.
- 1 means termination because the TsVoiceStop command was issued
- 2 means termination because line or voiceunit was placed on hook

3 means termination because the maximum lfa was reached

4 means termination because the maximum sample was reached

Request Block

TsVoicePlaybackFromFile

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	2
1	RtCode	1	0
2	nReqPbCb	1	2
3	nRespPbCb	1	3
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8021h
12	iTmModule	2	
14	pWorkArea	4	
18	sWorkArea	2	
20	pVoiceControl	4	
24	sVoiceControl	2	
26	pLfaLast	4	
30	sLfaLast	2	4
32	pqSampleLast	4	
36	sqSampleLast	2	4
38	pStatusRet	4	
42	sStatusRet	2	2

TsVoiceRecordToFile

Description

The TsVoiceRecordToFile service copies voice data to the specified file.

Procedural Interface

TsVoiceRecordToFile (iTmModule, pWorkArea, sWorkArea, pVoiceControl, sVoiceControl, pLfaNext, pqSampleNext, pStatusRet) : ErcType

where:

iTmModule — is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module

pWorkArea
sWorkArea — describes a memory location to be used by the Telephone Service for communicating to the Voice Processor Module

pVoiceControl
sVoiceControl — describe the Voice Control Structure shown in table 5–2

pLfaNext — points to the returned lfa of the next sector in the recording file at time of termination

pqSampleNext — points to the returned sample number which would have been given the next sample in the recording at time of termination

pStatusRet — points to the return status of the operation, where:

- 0 means termination because of an error
- 1 means termination because the TsVoiceStop command was issued

2 means termination because line or voice unit was placed on hook

3 means termination because the maximum lfa was reached

4 means termination because the maximum sample was reached

5 means termination because the maximum pause was detected

Request Block

TsVoiceRecordToFile

Offset	Field	Size (bytes)	Contents
0	sCntlnfo	1	2
1	RtCode	1	0
2	nReqPbCb	1	2
3	nRespPbCb	1	3
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8022 h
12	iTmModule	2	
14	pWorkArea	4	
18	sWorkArea	2	
20	pVoiceControl	4	
24	sVoiceControl	2	
26	pLfaNext	4	
30	sLfaNext	2	4
32	pqSampleNext	4	
36	sqSampleNext	2	4
38	pStatusRet		
42	sStatusRet	2	2

TsVoiceStop

Description

The TsVoiceStop service causes the digitization to be suspended.

Procedural Interface

TsVoiceStop (iTmModule) : ErcType

where:

iTmModule — is the Voice Processor Module, 1 being the first Voice Processor Module to the right of the CPU module

Request Block

TsVoiceStop

Offset	Field	Size (bytes)	Contents
0	sCntInfo	1	2
1	RtCode	1	0
2	nReqPbCb	1	0
3	nRespPbCb	1	0
4	userNum	2	
6	exchResp	2	
8	ercRet	2	
10	rqCode	2	8024h
12	iTmModule	2	

Voice Services Structures

Introduction

The Voice Structures included here are an integral part of the operations given in section 4, Operations. They are also referred to frequently in section 3, Programming Concepts. The tables are as follows.

5-1 Dial Characters

5-2 Voice Control Structure

5-3 Voice Data File Header

5-4 Voice Data File Record

5-5 Configuration File Format

5-6 Telephone Server Configuration Block

5-7 Telephone Status Structure

5-8 Telephone Module Control Block (TMCB)

Table 5-1 **Dial Characters**

Characters	Function
0-9,A-Y,a-y,*,#	are the dial or DTMF characters normally found on the touch pad or rotary switch. Q, q, Z, and z are not mapped.
!A,!B,!C,!D	are the DTMF characters A–D which do not normally appear on the touch pad
%	switch to pulse dialing
&	switch to tone dialing
@	flash for the default flash time
!@x	flash for x units of 100 ms (x is a byte)
=	wait for dial tone
!=	wait for any tone
?	wait for the phone being called to answer
~	pause for the default pause time
!x	pause for x units of 100 ms
$tl	generate part of a dual-tone multifrequency (DTMF) tone t for time l (units of 10 ms). The primary use of this is to generate a single tone (for use in answering machines, for example).

Table 5-1 **Dial Characters** (continued)

Characters	Function
	The tone generated is derived from a combination of the two frequencies specified by the upper (column) and lower (row) 4 bit nibbles of t. The valid values of nibble are 0Eh, 0Dh, 0Bh, and 07h. If both upper and lower nibbles are valid, then the tone generated will be one of the 16 DTMF tones as specified in the following table:

	0E0h	0D0h	0B0h	070h
0Eh	1	2	3	A
0Dh	4	5	6	B
0Bh	7	8	9	C
07h	*	0	#	D

For example, a value of 0E7h will generate the tone for *.

If only one of the nibbles are valid (such as 07h or 0E0h) then a single tone will be generated. If neither nibble is valid then no tone is generated.

SP,(,),-,/ these characters are ignored (SP is a space, or ASCII 20h)

Table 5-2 **Voice Control Structure (VCS)**

Offset	Field	Size (bytes)	Description
0	fh	2	is an open file handle
2	lfaStart	4	is the position in the file where to start or playback the recording.
6	lfaMax	4	recording or playback will be terminated if this position in the file is reached
10	qSampleStart	4	the sample number to be assigned to the first sample recorded, or to be skipped to on playback
14	qSampleMax	4	recording or playback will be terminated if this sample number is reached
18	cPauseMax	2	on record, this is the maximum silence in units of 100 ms before the recording is terminated (the terminating pause will not be included in the recording) on playback, all pauses will be truncated to this amount. A value of OFFFFh means no pause termination or truncation
20	cSampleOn	2	

Table 5-2 **Voice Control Structure (VCS)** (continued)

Offset	Field	Size (bytes)	Description
22	cSampleOff	2	determine the playback fast forward rate. A zero value for cSampleOff means playback at normal speed. The cSampleOn value determines the number of pairs of samples which will be played back, and cSampleOff is the number of sample pairs which will be discarded. cSampleOn should be greater than 50. To playback at double speed, values of 100 and 100 could be used. To playback at triple speed, values of 100 and 200 could be used.
24	f6KHz	1	if true, the codec sampling rate will be 6KHz, otherwise 8KHz (8192 4 bit samples per second)
25	fAutoStart	1	if true, the recording or playback will start as soon as the voice unit is placed off hook, or immediately if it is already off hook
26	fNoPause	1	if true, the pause detection hardware will be disabled and no pause detection or compression will be done
27	fStopOnDialTone	1	if true, the recording will be terminated if dial tone is detected

Table 5-2 **Voice Control Structure (VCS)** (continued)

Offset	Field	Size (bytes)	Description
28	fAltConnection	1	if true, an alternate connection will be used (if possible) to increase gain levels.
29	nSectorStatusUpdate	2	if greater than zero, any TsGetStatus requests will be responded to each time the specified number of sectors have been processed.
31	sPauseGap	2	number of data bytes after start of pause and before end of pause where the actual data are used. If sPauseGap is 0 then the default value (250) will be used.
33	fRawData	1	if true, then the voice data will be recorded or played back with out examination of the data for escape sequences.

Table 5-3 **Voice Data File Header**

Offset	Field	Size (bytes)	Description
0	signature	2	must be VC (4356h)
2	version	2	version of file, currently 1.
4	reserved	2	
6	f6KHz	1	if true, the data were recorded at 6KHz
7	lfaStart	4	starting lfa in file of voice data
11	lfaMax	4	ending lfa in file of voice data
15	qSampleStart	4	starting data byte count of voice data
19	qSampleMax	4	ending data byte count of voice data
23	reserved	1	
24	dateTime	4	system date time when the recording was made
28	line	1	line over which recording was made, where 0 is the voice unit, and 1 or 2 is telephone line 1 or 2
29	reserved	227	

Table 5-4 **Voice Data File Record**

Offset	Field	Size (bytes)	Description
0	qSampleStart	4	the accumulated sample number of the first sample in this record
4	signature	1	always a V (56h)
5	version	1	version number of data file. Currently always 1.
6	rgbSample	506	voice data

The Telephone Server configuration file is shared with the Operator program. The first 256 bytes are used by the Telephone Server and the remainder of the file is used by the Operator.

If no configuration file is present, then the Telephone Server system service is initially configured so that ACTION–1 and ACTION–2 correspond to TsFunction values 2 and 3, respectively. Changing the values of rgAction and/or rgFunction will override this.

Table 5-5 **Configuration File Format**

Offset	Field	Size (bytes)	Description
0	signature	2	Always a tC.
2	version	2	Version number of configuration file. Currently not used.
4	TsConfig	18	Configuration information (see table 5-5).
22	reserved	106	
128	rgAction	64	Array of action key values (unencoded keyboard values) to be mapped to server functions. A value of 0 terminates the list.
192	rgFunction	64	Array of functions (see TsFunction) to be executed when the action key listed in rgAction is typed.
256	reserved	128	
512	OperatorData	512	The Operator program uses this area for storing data such as the phone number of each line or dialing prefix.

In the following structure, all fields except rgRingHz are two byte arrays, one byte for each telephone line. The default values given are used when no configuration file is present.

Table 5-6 **Telephone Server Configuration Block**

Offset	Field	Size (bytes)	Description
0	rgDtmfGenOff	2	
2	rgDtmfGenOn	2	Specifies the time between and during DTMF tones when dialing, in units of 10ms. Most PBXs and CBXs can handle values of 10 (100 ms), and some as little as 6 (60 ms). The default is 6 for the off time and 8 for the on time.
4	rgFlashTime	2	Specifies the length of a default flash (@ character in a dial string) in units of 100 ms. Most PBXs use a value of 10 (1 second). The default is 10.
6	rgPauseTime	2	Specifies the length of a default pause (character in a dial string) in units of 100 ms. The default is 20.
8	rgfPulseDial	2	If true, then the pulse dialing is used instead of DTMF generation. The default is false.

Table 5-6 **Telephone Server Configuration Block** (continued)

Offset	Field	Size (bytes)	Description
10	rgRingHz	4	This array is used to specify the frequency of the workstation monitor's beeper when ring signal is detected on: 0 neither line (default 0). 1 line 1 (default 40). 2 line 2 (default 90). 3 both lines 1 and 2 (default 150). A value of 0 means no tone is generated. The range of frequencies is 0 to 255.
14	rgiRingThrough	2	The ring number in which the ring current will be sent directly to the voice unit (phone). A value of OFFFFh means the current is never passed through. The default is 4.
16	rgRingMode	2	The ring mode for each line. When a line rings, some combination of ring the monitor or ringing the phone is done, where 0 does not ring either. 1 rings only the phone. 2 rings only the monitor. 3 always rings both.

Table 5-6 **Telephone Server Configuration Block** (continued)

Offset	Field	Size (bytes)	Description
			4 rings the monitor for the number of rings specified in rgiRingThrough above, then rings the phone. 5 rings the monitor for the number of rings specified in rgiRingThrough above, then rings both (default case).

Table 5-7 **Telephone Status Structure**

Offset	Field	Size (bytes)	Description
0	iEvent	2	A counter which is incremented every time the contents of the status structure changes.
2	defaultLine	1	The default line as selected by TsFunction.
3	fNeedCodecConnection	1	If true, then a voice operation is waiting for a connection before it can begin.
4	fCodecInUse	1	If true, then the codec is recording or playing back.
5	aTmcb	3	The 24 bit real address of the TMCB

Table 5-7 **Telephone Status Structure** (continued)

Offset	Field	Size (bytes)	Description
8	reserved	2	Reserved
10	baudRate	2	Reserved
12	originateMode	1	Reserved
13	parityMode	1	Reserved
14	reserved	1	Reserved
15	reserved	1	Reserved
16	vUnitState	16	The voice unit state, as described in vUnitState.
32	rgLineState(2)	32	The line 1 and line 2 states, as described in lineState.
64	codecState	16	The codec state as described in codecState.
0	fOff Hook	1	If true, then the voice unit is off hook.
1	rgfTLine(2)	2	An array of flags which, if true mean line 1 or line 2, or both is connected to the voice unit.

Table 5-7 **Telephone Status Structure** (continued)

Offset	Field	Size (bytes)	Description
3	rgfRingThrough(2)	2	An array of flags which, if true, means line 1 or line 2 is connected directly to the voice unit to allow ring voltage through.
5	fCodec	1	If true, the voice unit is connected to the codec.
6	fDtmfRec	1	If true, the voice unit is connected to the DTMF decoder.
7	fMonitor	1	If true, the voice unit is connected in monitor mode.
8	handle	2	The handle of the connection to which the voice unit is attached.
10	hookThroughMode	1	The mode for passing through on hook or off hook from the voice unit to the lines; where 0 means neither line, 1 means line 1, 2 means line 2, and 3 means both lines.
11	reserved	5	

Table 5-7 **Telephone Status Structure** (continued)

Offset	Field	Size (bytes)	Description
LineState			
0	status	1	The line status, where: 0 on hook 1 ring current detected 2 reserved 3 reserved 4 reserved 5 off hook
1	fDialing	1	if true, then a dial string is being processed on this line
2	dialState	1	the current state of dialing, where: 0 idle 1 generating DTMF 2 analyzing CPTR (waiting for dial tone) 3 generating pulse 4 generating flash 5 generating pause

Table 5-7 **Telephone Status Structure** (continued)

Offset	Field	Size (bytes)	Description
			6 analyzing CPTR (waiting for any tone)
			7 analyzing CPTR (waiting for answer)
			255 doing error recovery
3	fRinging	1	if true, the line is ringing
4	iRing	1	number of rings
5	fRingThrough	1	if true, the ringing is being passed through to the voice unit
6	fOff Hook	1	if true, the line is off hook
7	fHold	1	if true, the line is on hold
8	fCodec	1	if true, voice is being used
9	fDtmfRec	1	if true, then the DTMF detector is being used
10	handle	2	the connection handle
12	reserved	1	
13	reserved	1	
14	reserved	2	

Table 5-7 **Telephone Status Structure** (continued)

Offset	Field	Size (bytes)	Description
Codestate			
0	lfaCurrent	4	lfa of data record last processed
4	qSampleCurrent	4	data byte number last processed
8	reserved	8	

Table 5-8 **Telephone Module Control Block (TMCB)**

Offset	Field	Size (bytes)	Description
0	signature	2	Must be 08459h. It is checked by the Voice Processor firmware before generating each interrupt.
2	version	1	This is the firmware version number.
3	command	1	This is the command that the server gives to the Voice Processor.
4	vUnitStatus	1	The current state of the voice unit, where: 0 means on hook 1 means off hook

Table 5-8 **Telephone Module Control Block (TMCB)** (continued)

Offset	Field	Size (bytes)	Description
5	xpLine	1	The device to be connected in the XptSwitch, or for which data are to be started or stopped, where: 0 voice unit (with attenuation added) 1 reserved 2 DTMF generator 3 DTMF receiver 4 Call Progress Tone Receiver (CPTR) used for detecting dial tone and busy signals 5 CODEC voice digitization encoder 6 CODEC voice digitization decoder 7 voice unit
6	xpJunctorMap	1	The bit map of telephone lines and terminals in the XptSwitch for a connection, or the telephone line for hold and hook commands, where bits are assigned as follows: 0 (1h) line 1 1 (2h) line 2

Table 5-8 **Telephone Module Control Block (TMCB)** (continued)

Offset	Field	Size (bytes)	Description
			2 (4h) terminal 3 (8h) terminal with attenuation
7	param0	1	
8	param1	1	The param bytes are used in conjunction with various other commands as described.
9	t0status	1	
10	t1status	1	The current state of the PBX or CBX lines 1 and 2, where: 0 on hook 1 ring current detected 2 reserved 3 reserved 4 reserved 5 off hook
11	dialerCQCB	8	Control structure used for pulse dialing.
19	sDtmfGenBuf	1	Number of encoded characters to be dialed using DTMF generator.
20	aDtmfGenBuf	3	Address (24bit real address) of first encode character.

Table 5-8 **Telephone Module Control Block (TMCB)** (continued)

Offset	Field	Size (bytes)	Description
23	bDtmfChar	1	Character decoded using DTMF receiver.
24	reserved	7	
31	reserved	8	
39	reserved	8	
47	codecBPCB	8	Control structure for CODEC encoding or decoding.
55	ercCodec	2	Error status from CODEC.
57	reserved	2	
59	cCptrLow	1	
60	cCptrHigh	1	High and low signal detected by CPTR. A low/high value of 0/255 means dial tone, 45–55/45–55 means busy, and 15–35/15–35 means fast busy. Other values are not defined and may mean no connection, voice, or noise on the line.
61	intStatus	1	The Voice Processor firmware places a status byte here before generating the XINT3 interrupt signal to wake up the server. The server puts a OFFh here after processing the interrupt.

Table 5-8 **Telephone Module Control Block (TMCB)** (continued)

Offset	Field	Size (bytes)	Description
62	xptMap	8	The state of the cross point switch is updated by the Voice ProcessorB.

Appendix A

The following are the error codes generated by the Voice Services Telephone Server:

Error Code	Error
11200	Invalid user. The user number of the program making the request is not an allowed user at this time for the specified operation.
11201	Reserved.
11202	Invalid connection. The connection resulting from the specified operation is not allowed.
11203	Invalid parameters. The parameters to the specified operation are not valid.
11204	Not off hook. The voice unit or telephone line must be off hook for the specified operation.
11205	Bad handle. The handle specified was invalid.
11206	Timeout. The operation was terminated because a specified time limit for completion expired.
11207	Invalid module. The module specfied is invalid.
11208	Not on hook. The line must be on hook for the operation specified.
11209	Voice unit on hook. The voice unit must be off hook for the operation specified.
11210	Invalid function. The function specified to the TsDoFunction operation is illegal.
11211	On hold. The operation was terminated due to the line being placed on hold as the result of command being issued.

Error Code	Error
11212	On hook. The operation was terminated due to the phone or line being placed on hook or to a TsOnHook or TsDoFunction (hang up) command being issued.
11213	Service reset. The service was reset with the TsDoFunction service.
11214- 11219	Reserved.
11220	Voice work area too small. The work area must be at least 13312 bytes and word alligned.
11221	Voice in use. The voice hardware is either in use or has been locked with TsDoFunction.
11222	Voice sample not found. The specified starting sample could not be found.
11223	Voice data lost (disk). The voice operation was terminated due to the disk and/or cluster not being able to keep up with the voice data rate.
11224	Voice data lost (cpu). The voice operation was terminated due to the Telephone Server process not being able to keep up with the voice data rate.
11225	Bad voice parameters. One or more field in the voice control structure are invalid.
11226	Invalid voice data file. The signature byte is invalid in a record of the file specified.
11227- 11249	Reserved.

Error Code	Error
11251	Initialization time out. The Voice Processor module did not respond due to a failure of the Voice Processor module, XBus DMA or XBus interrupt hardware or software.
11252	No request file. The Telephone Server request file did not get loaded by the Operating System during system initialization (boot).
11253	Bad request file. The Telephone Server request file is not of the same revision level as the Telephone Server.
11254	Telephone Server already installed.
11255	Voice Processor module failure. The Voice Processor module did not respond due to a failure of the Voice Processor module, XBus DMA or XBus interrupt hardware or software.
11256	Bad request. A request not recognized by the Telephone Server was received at the Telephone Server's service exchange.
11258	User swapped. The operation was terminated due to the swapping to disk of the program using the service.
11259	Invalid configuration file. The signature word in the configuration files specified is invalid.
11260	Duplicate request. The request is a duplicate of one currently being processed and is not allowed.
11261	Firmware invalid. The Voice Processor module firmware is invalid.
11262- 11269	Reserved.

Error Code	Error
11270	Bad dial character. A character specified in the dial string is not valid.
11271	Dialing in progress. A TsDial service is in progress using the dialing hardware.
11272	No dial tone: Busy. A busy signal was detected instead of dial tone when a wait for dial tone command (=) was processed.
11273	No dial tone: Fast busy. A fast busy signal was detected instead of dial tone when a wait for dial tone command (=) was processed.
11274	No dial tone. No dial tone was detected when a wait for dial tone command (= or !=) was processed.
11275	Bad dial pulse character. A character was processed which can not be dialed with pulse dialing (such as # or *).
11276	Too many characters to dial. No more than 64 characters can be processed at a time.
11277	Detector in use. The hardware required for detecting DTMF or dial tone is in use by another operation.
11278	No answer. The party being called did not answer.
11279- 11299	Reserved.

Glossary

Analog crosspoint switch. The analog crosspoint switch is the switching unit inside the Telephone Manager Module which makes connections between incoming and outgoing analog signals and the Telephone Manager's digital hardware.

CODEC. CODEC is the encoder/decoder hardware of the Telephone Manager Module which actually makes the analog to digital signal conversions.

CPTR. CPTR stands for Call Progress Tone Receiver. The receiver is used for detecting dial tone, busy, fast busy, and if the party called has answered.

DTMF. DTMF stands for Dual Tone Multifrequency. The double toned signal produced by the telephone, which when dialed, alerts the telephone company exchange to the placement or reception of a call. The hardware has both a DTMF generator for producing DTMF tones and a DTMF receiver for receiving them.

Flash. A flash is used with some PBXs to signal that a PBX feature is desired. A flash is generated by placing the line on hook for a short period of time, usually about 1 second.

Hertz (hz). Hertz is a cycle or period per second.

Off hook. Off hook is the electical state of a telephone line when a phone or other device is connected to the PBX or telephone company.

On hook. On hook is the electical state of telephone line when the line is not connected.

PBX. PBX stands for Private Branch Exchange in which the internal telephone system of a business or office processes telephone calls through its own switching system. This system is generally tied into, but independent of, the Telephone Company system.

Pulse. Pulse is an older form of dialing than DTMF in which the line is placed on and off hook rapidly.

Switchook. Switchook is the same as flash.

System service. System service is a program or subprogram which provides a service for other programs. It may include one or more processes, and may be part of the operating system, an installable program , or an application.

Telephone line. The telephone line is the connection to a PBX or telephone company.

Voice unit. A voice unit is any attachment to the workstation which can be used to send or receive voice messages: a phone, a telephone, a telephone set, or speaker phone.

Index

S

T

U

V

W

X

NOTES

NOTES