PHOENIX Audio Technologies

Phoenix TCP/IP Control Interface

TABLE OF CONTENTS

INTRODUCTION

Phoenix provides control on some of its devices using TCP/IP protocol via the LAN link. The protocol allows the user/installer to control some of the devices' functionality, including telephony functions like dialing, if and when applicable. Phoenix implemented this protocol in a few applications that are available on line for use with Windows, Mac OS, iOS, and Android operating systems. The user can use this protocol to develop their own controlling software on any third party controllers.

This document defines the interface for the control.

Throughout the document, the Phoenix Device will be referred to as 'the device', the controlling device/software will be referred to as 'the controller'.

COMMUNICATION WITH A DEVICE

General

The Controller will communicate with the Phoenix device via the LAN using TCP port 8888.
The messages sent on both directions, from the device and from the controller, comprises of printable ASCII characters terminated by a null character ('\0', or byte value 0).

All messages starts with upper case keyword ended with ':'. After the ':' keyword terminator one or more parameters may be added. The parameters and the keyword terminator are separated by a space character (0x20).

If the content to be conveyed is larger than the maximum data length (1000 bytes), which may happen with the 'CONTACTS:' and 'CALLS:' messages, the content will be split and sent in several consecutive messages, so that none of them exceeds the maximum length.

COMMUNICATION WITH A DEVICE

Starting a communication link with a device and logging in

A controller initiates a link with a device by sending it the 'LOGIN1: <IP address> [<password>]' message on the LAN.

The device will reply by sending '<LOGIN: <success> <device type> <device name>' message to the controller, where the value of the <success> is one of the following:
0: Login failed.
1: Login is successful.
2: password is needed but wasn't sent in the LOGIN1 request

Note: by default, the Phoenix devices do not require a password to communicate with the controller. However, the user has an option to set a password protection. This is done by login-in to the unit's control panel, and under 'SIP Client protocol -> Management -> Remote Control' selecting the password protection mechanism.

Once a controller is logged in to a device, it can send commands and will get updated with status and replies from the device.

In order to terminate the connection and stop the device from sending status updates, the controller should send 'STOP:' message to the device.

Note: By definition, TCP protocol is based on IP addresses; in other words the two communicating devices need to know each other's IP address, and the pairing is done using these addresses. This requires that the Device's IP address will be set to Static. The Phoenix protocol allows the controller to query the IP address of the device it wants to connect by using its name, hence enabling the establishment of connection with a device that has a dynamic IP address. However, this query has to be done using UDP protocol through port 8888. The controller will broadcast the 'WHERE_IS: <device name>' message on the LAN and receive back the required IP address. All the connected devices on the LAN receive the WHERE_IS: query, but only the one which its name is set in the broadcasted message <device name> parameter will reply by sending 'I_AM_HERE: <IP address> <device name>' message to the controller on UDP port 8888.

MESSAGES FROM THE CONTROLLER

Message from Controller	Description
WHERE_IS: <device name>	Used to query the IP address of a device with a <device name> name. **This message should be broadcasted using UDP port 8888.**
LOGIN1: <controller's MAC address> [<Password>]	Used to login to a device. The controller should send its MAC address as the first parameter. If the device expects a password for login, the password should be sent after the MAC address.
KEY: <key code>	Notify the device that a key was pressed. The key codes are: 0 to 9 – digit keys '0' to '9'. 10 – '*' 11 – '#' 17 – Speaker Volume up 18 – Speaker Volume down 19 – Send (hook-off) 20 – End (hook on) 23 – Contacts List 24 – Call List (all calls) 30 – Get device status 31 – Get call status
DIAL: <number>	A request to dial a number.
SET_VOLUME: <channel code><percent>	A request to set the volume of the channel identified by <channel code>* to <percent> (0 to 100).
GET_VOLUME: <channel code>	A request to get volume level of the channel identified by <channel code>*

MESSAGES FROM THE CONTROLLER

Message from Controller	Description
SET_MUTE: <channel code> <mute> Channel Codes 21 & 22 are only compatible with MT700	A request to mute (<mute> = 1) or unmute (<mute> = 0) the channel identified by <channel code>*.
GET_MUTE: <channel code> Channel Codes 21 & 22 are only compatible with MT700	A request to get the mute status of the channel identified by <channel code>*
SPLIT_ZONE: <code> *MT700 Only	A command to split the device chain into zones (<code> = 1) or to cancel the chain split (<code> = 0)
GET_LINE_TYPE: *MT700 Only	A request to get the type of the line inputs
GET_DIALED:	A request to get the dialed call list from the device.
GET_RECEIVED:	A request to get the received call list from the device.
GET_MISSED:	A request to get the missed call list from the device.
STOP:	Terminate the connection. Upon receiving this command, the device will disconnect its link with the controller and stop sending messages to it.

*Following is a list of the channels and their code used for the volume and mute commands:

Channels	Mute Control	Volume Control	Channel code
Podium lines	+	-	21
Microphones and non-podium lines	+	-	22
System output	+	-	23
Speakers	+	+	24

MESSAGES FROM THE DEVICE

I_AM_HERE: <IP address> <device name>

This is a response from a device to the 'WHERE_IS:' query. This reply will be sent using UDP port 8888 to the querying controller. The <IP address> and the <device name> are of the responding device.

'LOGIN: <success> <device type> <device name>'

After receiving 'LOGIN1' request from a controller, the device will respond with LOGIN reply with a <success> value: 1 – login is successful. 0 – login failed. 2 – Password is needed. The device will send its type in the <device type> field, and its name in the <device name> field. The <device type> will be 'stingray', 'condor' or other device type that may be added in the future, and will not include spaces. The device name may include spaces, and will be terminated by the NULL character at the end of the message.

'HOOK_STATE: <status>'

This message will be sent whenever the hook state of the SIP phone changes. The <status> value will be either 'ON' or 'OFF'.

'DEVICE_STATUS: <device status code>'

The device will send this message to indicate the following status: USB connection status, Phone registration status, Do Not Disturb (DND) and HD Quality. This message will be sent whenever the device status changes, and upon receiving the 'KEY: 30' message.

Device status code	Description
10	USB is not connected
11	USB is connected
20	Phone is not registered
21	Phone is registered and ready to use
30	Do not Disturb is canceled
31	Do not Disturb
40	High definition is canceled
41	In high definition call (HD)

MESSAGES FROM THE DEVICE

'CALL_STATUS: <call status code> [<string>]'

The device will send this message to indicate the call status. Depending on the <call status code>, the <string> parameter may be added with the name or number of the other side of a call. This message will be sent whenever the call status changes, and upon receiving the 'KEY: 31' message.

Call Status Code	Description
0	Idle
5	Ringing
6	Dialing
7	Incoming call
8	Outgoing call
9	Waiting call
15	Conference call

'VOLUME_LEVEL: <channel code> <percent>

This command will be sent as a response to a 'GET_VOLUME:' command sent by the controller. The <channel code> is the code identifying the channel (see table above). The <percent> is the volume level in the range 0 to 100.

'MUTE_STATUS: <channel code> <status>

This command will be sent as a response to a 'GET_MUTE:' command sent by the controller. The <channel code> is the code identifying the channel (see table above). The <status> is 1 if the channel is muted, and 0 if the channel is unmuted.

'LINE_TYPE: <line 1 type> <line 2 type> <line 3 type> <line 4 type>

This command will be sent as a response to a 'GET_LINE_TYPE:' command sent by the controller. The 4 values returned on the message are the line type code of each of the line inputs: 0 – Mixer 1 - Poduim 2 – Auxiliary 15 – Not set.

MESSAGES FROM THE DEVICE

'CONTACTS: <index> <number> <name> [;<index> <number> <name> [; …]]'

The device uses this message to send the items of its phone book to the controller. This message will be sent as a reply to the 'KEY: 23' message. Each entry in the phone book consists of index in the list (starting from 1), the phone number and the contact name. Many entries can be sent in a single message, where ';' is the separator between two entries. The <name> may contain spaces but not ';'. If the phone book is big, it may be sent in few messages. When the controller receives 'CONTACTS:' packet with entry index 1 in it, it should initialize its contact list. It may be needed few 'CONTACTS:' commands in order to load the entire phone book.

'CALLS: <index> <type> <number> <name> <time> <duration> [;<index> <type> <number> <name> <time> <duration> [; …]]'

The device uses this message to send its calls list entries to the controller. This message will be sent as a reply to the 'KEY: 24' message, and to the 'GET_DIALED:', 'GET_RECEIVED:' and 'GET_MISSED:' requests from the controller. If the message is a reply to the 'KEY: 24' command, all the calls in the device's log will be mixed and sent, ordered by the time of the call. If the message is sent as a reply to the other requests, only the requested type of calls will be sent.

Each entry in the calls log consists of:
- Index in the list (starting from 1)
- Type of call: '0 – Dialed call 1 – Received call 2 – Missed call.
- Phone number
- Contact name.
- Time of call start (text).
- Duration of call (text)

Many entries can be sent in a single message, where ';' is the separator between two entries. Spaces in the <name> will be sent as '^', the controller should turn them into spaces. ';' will not be embedded in the <name>. When the controller receives 'CALLS:' message with entry index 1 in it, it should initialize its call list. It may be needed few 'CALLS:' commands in order to load the entire log.